Reinforcement Learning for Uncoordinated Multiple Access

Benoît-Marie Robaglia

INSTITUT POLYTECHNIQUE DE PARIS

NNT : 2024IPPAT010

Thèse de doctorat

Reinforcement Learning for Uncoordinated Multiple Access

Thèse de doctorat de l'Institut Polytechnique de Paris
préparée à Télécom Paris

École doctorale n°626 École doctorale de l'Institut Polytechnique de Paris (ED IP Paris)
Spécialité de doctorat: Mathématiques et Informatique

Thèse présentée et soutenue à Paris, le 05/03/2024, par

BENOÎT-MARIE ROBAGLIA

Composition du Jury :

Salah-Eddine El AYOUBI Professeur, CentraleSupélec	Président/Examinateur
Inbar FIJALKOW Professeure, ENSEA / CY Cergy Paris Université	Rapporteuse
Marios KOUNTOURIS Professeur, EURECOM	Rapporteur
Cedric ADJIH Chargé de Recherche, INRIA	Examinateur
E. Véronica BELMEGA Professeure, ESIEE Paris, Université Gustave Eiffel	Examinatrice
Marceau COUPECHOUX Professeur, Télécom Paris	Directeur de thèse
Dimitrios TSILIMANTOS Ingénieur de Recherche, Huawei Technologies	Invité

A mes parents, Jean-Louis et Annick.

Remerciements

Je souhaite tout d'abord exprimer ma profonde gratitude à mes deux superviseurs, Marceau et Dimitrios. Au cours de ces trois dernières années, vous m'avez fait découvrir et apprécier ce domaine fascinant des télécommunications sans fil, à moi, un statisticien, n'ayant jamais eu d'appétence pour la physique. Votre capacité à me challenger scientifiquement, tout en valorisant et en respectant mes idées, a été un levier de croissance exceptionnel et est un cadeau que je chérirai tout au long de ma carrière. Enfin je vous remercie pour votre implication et perspicacité concernant l'organisation et la rédaction d'articles scientifiques. Je l'avoue, il m'est arrivé de me laisser aller sur certaines formulations, conforté par l'idée que votre regard expert viendrait polir mes ébauches.

Je tiens à remercier tout particulièrement Marceau pour ton investissement sans faille sur le plan académique, administratif, et humain. Si la transition avec Huawei s'est aussi bien déroulée c'est en partie grâce à toi. Je te remercie également pour ta disponibilité et tes conseils lors des moments d'isolement et de doute. Dimitrios, ta décision de continuer à apporter ton soutien à ma thèse, même après mon départ de Huawei et de façon plus informelle, a réellement compté pour moi. Tes commentaires, toujours pertinents et rigoureux, sont des qualités que je m'efforce d'incorporer dans mon travail futur.

Par ailleurs, il m'est essentiel de remercier les rapporteurs et les membres du jury pour leur précieux travail. Malgré des contraintes de temps, la soutenance s'est merveilleusement bien déroulée, et je vous suis infiniment reconnaissant pour votre engagement. Marios et Inbar, j'espère que la lecture de ce manuscrit aura été agréable, et je vous suis très reconnaissant d'avoir accepté cette tâche. Mes remerciements s'étendent également à Cédric, Véronica, et Salah, dont les questions pertinentes ont enrichi ma soutenance.

Je ne peux conclure sans exprimer ma profonde gratitude à l'égard de mes proches. Un immense merci à mes amis Bastien, Roland, Matthieu, Walid, Redwan, Alex, Axelle, Céline et Thami pour leur soutien indéfectible, leurs précieux conseils et encouragements. Enfin, je garde pour la fin une pensée toute particulière pour ma famille et ma chère Bettina, qui m'ont accompagné à chaque étape de ce parcours, offrant un soutien sans faille dans les moments de joie comme dans les épreuves.

Contents

List of Figures

List of Tables

List of Algorithms

List of Acronyms

Résumé en français

Les protocoles de contrôle d'accès au support (MAC) distribués sont fondamentaux pour les communications sans fil, mais les protocoles traditionnels basés sur l'accès aléatoire sont confrontés à des limitations importantes dans certains cas d'usage de l'internet des objets (IoT). En effet, ils ont du mal à garantir des critères de qualité de service exigents, ce qui les rend inadaptés aux communications ultra-fiables à faible latence (URLLC). Cette thèse aborde ces défis en exploitant le potentiel de l'apprentissage par renforcement profond (DRL), un paradigme dans lequel les agents optimisent leurs actions en interagissant avec un environnement.

En particulier, cette thèse aborde les principaux enjeux du problème de l'accès multiple (MA) pour les réseaux URLLC, incluant la latence excessive des protocoles centralisés, les collisions, les délais de retransmissions caractéristiques des protocoles sans allocation (GF) ainsi que la complexité de gérer l'hétérogénéité des appareils dans des environnements dynamiques. En outre, la thèse explore l'intégration de nouvelles techniques de couche physique comme l'accès multiple non orthogonal (NOMA) à des protocoles MAC dédiés aux communications URLLC. Notre méthodologie applique le DRL pour développer des protocoles intelligents, capables de s'adapter aux contraintes de l'URLLC.

Dans un premier temps, nous modélisons le problème de l'URLLC dans un paradigme centralisé, où la station de base (BS) orchestre les transmissions des appareils. Cette configuration présente l'avantage d'assurer une communication sans collision, mais introduit une observabilité partielle, car la station de base n'a pas accès à la mémoire et à l'état du canal des utilisateurs. Nous nous attaquons à ce problème en introduisant deux algorithmes : FilteredPPO et NOMA-PPO. Alors que le premier surpasse les algorithmes de référence dans les scénarios avec trafic quasi-périodique, le second démontre une performance supérieure à l'état de l'art également dans les scénarios avec trafic sporadique. Les troisième et quatrième contributions, SeqDQN et MCA-PPO, étudient l'application de l'apprentissage par renforcement multi-agents (MARL) pour l'URLLC où chaque appareil est équipé d'un algorithme DRL. Alors que SeqDQN explore une méthode pour aborder la non-stationnarité, améliorer le passage à l'échelle et l'apprentissage, MCA-PPO présente une solution théorique-

ment robuste pour le défi de l'accès dynamique multicanal (DMCA) permettant aux utilisateurs d'optimiser l'utilisation de la bande passante et donc d'améliorer les performances URLLC.

Les sections suivantes traitent du contenu de chaque chapitre.

Chapitre 1: Introduction

L'Internet des Objets (IoT) désigne un réseau interconnecté d'appareils physiques, de véhicules, de capteurs, de logiciels et d'autres technologies permettant de collecter, d'échanger et d'agir sur des données via Internet. Ces technologies, envisagées comme un réseau omniprésent de milliards d'appareils allant des smartphones aux drones et capteurs industriels, devraient révolutionner le paysage numérique. Toutefois, certaines applications de l'IoT présentent des contraintes strictes de latence caractérisées par le standard du Projet de Partenariat de Troisième Génération (3GPP) comme les Communications Ultra Fiables et à Faible Latence (URLLC). Par ailleurs, les protocoles traditionnels peinent souvent à répondre à ces exigences strictes, notamment en liaison montante, en raison de la charge et du délai de signalisation significatifs. Une alternative est l'accès aléatoire, mais cette méthode présente des inconvénients notables dans les scénarios URLLC, augmentant les latences dues aux collisions et retransmissions.

La liaison montante URLLC pose des défis uniques en raison de la nécessité de coordonner de manière décentralisée de multiples dispositifs IoT transmettant simultanément. Les approches peuvent être divisées en protocoles avec et sans allocation, chacun ayant ses propres avantages et inconvénients.

Les défis de l'accès multiple (MA) pour les réseaux URLLC se résument ainsi:

- **Protocoles Centralisés** : Ces protocoles facilitent la coordination des dispositifs et la prévention des collisions mais génèrent un surcoût de communication élevé, entraînant une latence substantielle. Le défi principal est de développer des protocoles centralisés pour les réseaux URLLC qui évitent la latence inhérente à l'établissement des liens.

- **Protocoles Sans Allocation (GF)** : Ces protocoles, potentiels candidats pour l'URLLC, évitent la latence liée au protocole de poignée de main à quatre temps. Cependant, ils introduisent une latence importante due aux

collisions et aux retransmissions propres aux protocoles décentralisés. Le défi est de concevoir des protocoles GF décentralisés qui atténuent efficacement l'effet des collisions.

- **Hétérogénéité des Dispositifs et Environnements Dynamiques** : La majorité des protocoles peinent à gérer l'hétérogénéité du trafic des dispositifs et les conditions de canal, ainsi qu'à gérer de manière optimale le trafic sporadique et les environnements dynamiques. Le défi est de concevoir des protocoles exploitant l'hétérogénéité des dispositifs et s'adaptant efficacement aux environnements dynamiques.

- **Nouvelles Techniques de Couche Physique** : Des techniques récentes comme la NOMA ou la technologie SIC, bien qu'offrant des solutions prometteuses pour répondre aux contraintes de l'URLLC, introduisent des défis tels que la sélection des utilisateurs et la gestion des interférences. Le défi est d'adapter les protocoles pour gérer la complexité accrue introduite par ces nouvelles techniques de la couche physique.

Dans le cadre de cette thèse, nous investiguons l'utilisation de l'apprentissage par renforcement afin de concevoir des protocoles de transmission sophistiqués, capables de répondre efficacement aux exigences rigoureuses de l'URLLC.

Chapitre 2: Fondements en Apprentissage par Renforcement Profond

Dans ce chapitre, nous introduisons le formalisme et les connaissances de base qui constituent les prérequis essentiels pour le reste de la thèse. En particulier, la section 2.1 présente les architectures de réseaux neuronaux employées tout au long de la thèse, la section 2.2 offre un aperçu de l'apprentissage par renforcement pour un agent unique et, enfin, la section 2.3 examine le cadre du multi-agent et ses défis.

Chapitre 3: FilteredPPO: une Approche Centralisée d'Apprentissage par Renforcement Profond pour le Trafic en Liaison Montante IoT

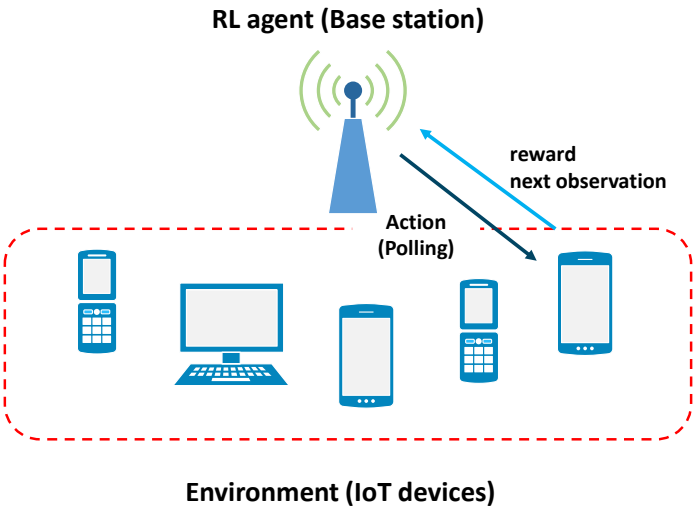


Figure 1: Modèle de système

Dans ce chapitre, nous présentons une solution centralisée pour aborder le problème d'accès multiple (MA) en liaison montante spécifique aux dispositifs IoT avec des exigences strictes de latence. Dans ce cadre, les dispositifs peuvent accéder à un médium de communication commun, mais uniquement lorsque la station de base (BS) choisit de les programmer. Cette orchestration assure l'absence de collisions, un problème persistant avec les protocoles d'accès aléatoire conventionnels. Néanmoins, afin de minimiser les ressources allouées à la coordination des appareils, la BS dispose d'une connaissance restreinte de la mémoire des appareils. Par conséquent, la BS doit apprendre efficacement les modèles de trafic des dispositifs pour maximiser le débit, étant donné son manque de connaissances en temps réel sur le moment où un dispositif souhaite transmettre. Notre modèle se classe dans la catégorie des protocoles centralisés avec communication limitée et la complexité du modèle est linéaire avec la taille du réseau. Dans ce chapitre, nous considérons un modèle de trafic probabiliste et périodique et des paquets avec des échéances strictes.

Nous pouvons observer que notre méthode parvient avec succès à apprendre les modèles de trafic des appareils malgré l'observabilité partielle de notre

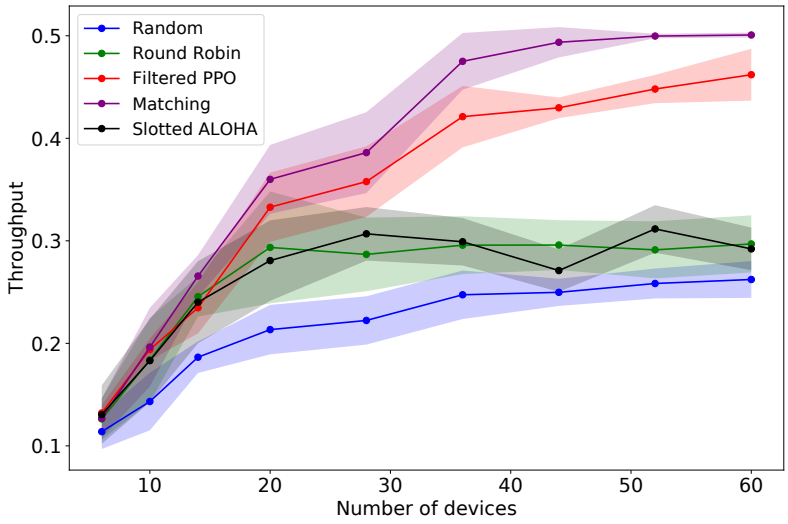


Figure 2: Evolution du débit en fonction du nombre de dispositifs. Les dispositifs sont synchrones. Les résultats sont calculés avec 5 graines.

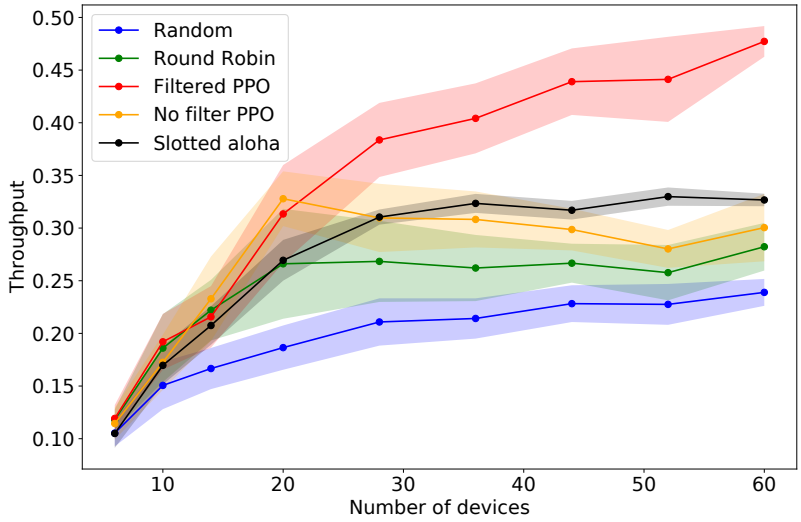


Figure 3: Evolution du débit en fonction du nombre d'appareils. Ces appareils sont asynchrones. Les résultats sont calculés avec 10 graines.

problème, et ce, à la fois lorsque les émetteurs sont synchrones (Fig. 2) et asynchrones (Fig. 3). Les résultats numériques montrent que notre solution surpasse les protocoles traditionnels d'accès multiple et atteint une performance compa-

rable à celle d'un algorithme optimal conscient des caractéristiques du trafic et des échéances strictes.

Chapitre 4: NOMA-PPO: une Approche Centralisée d'Apprentissage par Renforcement Profond pour les réseaux NOMA-URLLC

Dans ce chapitre, nous étendons le cadre d'étude de la liaison montante avec observabilité partielle introduit dans le Chapitre 3 aux réseaux NOMA-URLLC. Nos investigations précédentes ont mis en évidence une limitation centrale des approches de planification centralisée pour l'URLLC : la visibilité limitée sur l'état des dispositifs. Cette contrainte entrave considérablement la capacité de ce cadre à répondre aux exigences strictes de fiabilité de l'URLLC. Pour améliorer la fiabilité, nous proposons une stratégie innovante permettant à la station de base (BS) de sonder plusieurs utilisateurs dans le même intervalle temporel. Elle est également équipée de la technologie NOMA pour atténuer efficacement les interférences provenant des transmissions de paquets simultanées.

Cependant, cette approche introduit de nouvelles complexités. En effet, permettre à plusieurs utilisateurs d'accéder au canal simultanément augmente exponentiellement l'espace d'action de l'agent de RL, transformant le problème de sélection d'action en un problème combinatoire à chaque étape temporelle. De plus, l'agent RL doit non seulement gérer l'observabilité partielle sur l'état de la mémoire des utilisateurs, mais aussi faire face à une visibilité limitée sur l' état des canaux. Ces états de canal sont cruciaux pour optimiser le processus de décodage SIC dans l'implémentation NOMA.

Les contributions de ce chapitre sont les suivantes:

- Nous formulons un problème général d'accès multiple (MA) avec la contrainte URLLC, en considérant des paquets ayant des délais stricts et des communications en liaison montante NOMA comme un POMDP.

- Nous introduisons la notion d'"*agent state*" afin d'aborder théoriquement la formulation du POMDP. Nous démontrons que l'agent state est une statistique suffisante pour l'historique des observations-actions passées, ce qui

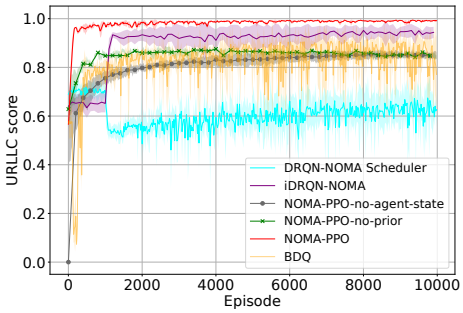
nous permet 1) d'exprimer les actions et observations passées de manière compacte, et 2) de convertir le problème de POMDP en MDP et de bénéficier des propriétés de convergence des algorithmes DRL. Cette transformation peut être étendue à d'autres contextes sans fil où l'observabilité partielle concernant le tampon ou l'évolution du canal doit être adressée.

- Nous proposons un algorithme DRL, NOMA-PPO, qui améliore l'algorithme de pointe PPO avec deux composants : 1) une architecture de réseau de politiques de branchement afin de gérer de manière linéaire les espaces d'actions combinatoires. Cette idée est inspirée de l'architecture BDQ et étendue aux méthodes PG. 2) Des politiques bayésiennes, qui incorporent des informations préalables sur le problème MA dans l'agent DRL.

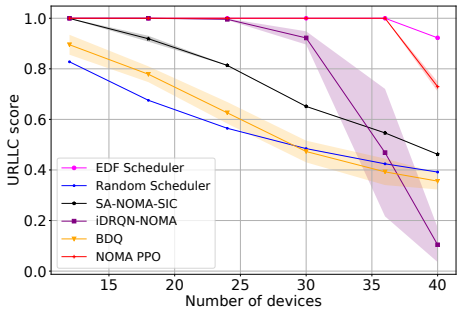
- Nous fournissons des preuves numériques que notre approche surpasse les références traditionnelles MA et DRL dans des scénarios 3GPP en termes de score URLLC (Fig. 4b and Fig. 4c), de vitesse de convergence (Fig. 4a), et d'équité (Fig. 4d). De plus, nous montrons que notre algorithme est capable de gérer différents modèles de trafic, notamment un modèle de trafic périodique déterministe et un modèle de trafic apériodique probabiliste. Enfin, notre algorithme est très robuste à des changements de configuration du canal. On montre qu'il sait exploiter les informations sur les canaux qu'ils reçoit au fil du temps (Fig. 5).

Nos expériences ont été conduites avec un modèle de réseau sans fil réaliste basé sur l'évaluation du SINR et du régime de longueur de bloc finie avec des paramètres tirés des recommendations 3GPP. Chaque appareil transmet un signal subissant une atténuation liée à la distance, de l'évanouissement rapide et du bruit thermique.

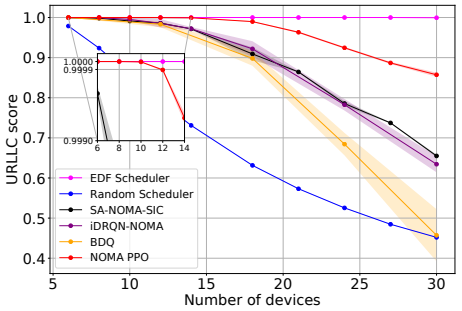
Nous considérons également deux types de modèles de trafic également décrits dans les standards 3GPP: le modèle probabiliste périodique et le modèle probabiliste apériodique.



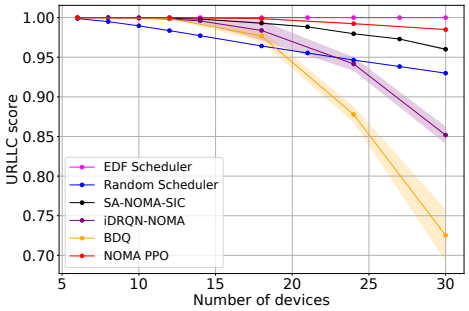
(a) Evolution du score URLLC pendant l'entraînement pour 18 utilisateurs.



(b) Evolution du score URLLC dans le scénario 3GPP déterministe périodique.

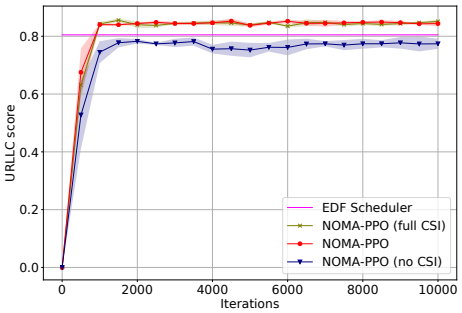


(c) Evolution du score URLLC dans le scénario 3GPP probabiliste apériodique.

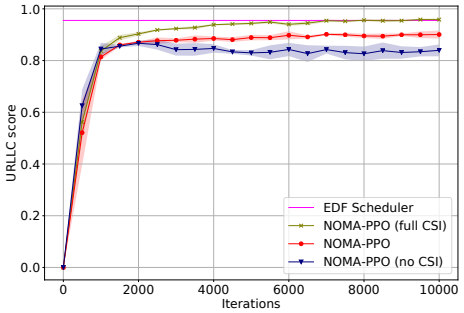


(d) Indice d'équité de Jain dans le scénario 3GPP probabiliste apériodique.

Figure 4: Métriques de performance dans le scénario 3GPP.



(a) $T_c = 1.4$ms, 10 appareils.



(b) $T_c = 0.34$ms, 10 appareils.

Figure 5: Évolution du score URLLC pendant l'entraînement sous différentes conditions de canal.

Chapitre 5: SeqDQN: une Approche Décentralisée d'Apprentissage par Renforcement Profond Multi-Agents pour les réseaux URLLC

Dans ce chapitre, nous proposons une approche entièrement décentralisée pour résoudre le problème d'accès multiple en liaison montante avec des contraintes de latence strictes. Nous équipons chaque appareil d'un algorithme d'apprentissage multi-agents profond (MARL) afin d'apprendre un protocole de transmission par interaction avec les autres dispositifs et l'environnement. Tout d'abord, nous réalisons une analyse approfondie de la performance des algorithmes MARL traditionnels, tels que iDQN et QMIX, appliqués à notre problème spécifique. Cette analyse nous permet d'évaluer les forces et les faiblesses des algorithmes les plus utilisés dans la littérature et de les utiliser comme références pour notre problème. Ensuite, nous introduisons SeqDQN, une solution MARL distribuée inspirée du paradigme d'entraînement à deux échelles temporelles où les agents ne mettent pas à jour leurs Q-fonctions simultanément. Dans SeqDQN, les dispositifs mettent à jour leur Q-fonction séquentiellement, en commençant par les dispositifs ayant l'exigence de latence la plus contraignante. Cette stratégie d'entraînement vise à atténuer les défis posés par la non-stationnarité qui provient de l'apprentissage indépendant lors de la formation d'agents décentralisés.

Nos contributions dans ce chapitre sont les suivantes:

- Nous formulons rigoureusement le problème multi-agents en liaison montante URLLC comme un Dec-POMDP, intégrant les caractéristiques et contraintes du scénario URLLC.

- Nous implémentons et évaluons les algorithmes d'apprentissage multi-agents renforcé (MARL) les plus couramment utilisés dans la littérature, à savoir iDQN pour l'approche apprentissage indépendant (IL) et QMIX pour l'approche d'apprentissage centralisé avec execution décentralisée (CTDE), dans le contexte de notre problème multi-agents en liaison montante URLLC.

- Nous introduisons SeqDQN, un algorithme MARL distribué où les agents ne mettent pas à jour leurs Q-fonctions simultanément. Au lieu de cela, ils mettent à jour leur Q-fonction séquentiellement, en commençant par les

dispositifs ayant l'exigence de latence la plus contraignante. Les avantages de cette méthode sont : 1) Nous réduisons la non-stationnarité causée par l'apprentissage simultané de plusieurs agents, qui est un inconvénient majeur de l'IL, 2) notre méthode proposée est plus évolutive pour un grand nombre d'agents que le CTDE et 3) l'entraînement est beaucoup plus rapide que les algorithmes MARL existants (iDQN et QMIX).

- Nous évaluons notre solution en comparaison avec des algorithmes références traditionnels dans plusieurs scénarios au sein de notre modèle de système, sous un trafic périodique probabiliste d'une part et déterministe d'autre part. Nous montrons que non seulement notre méthode surpasse les références d'accès multiple, mais elle atteint également ou dépasse les performances URLLC des algorithmes MARL de référence.

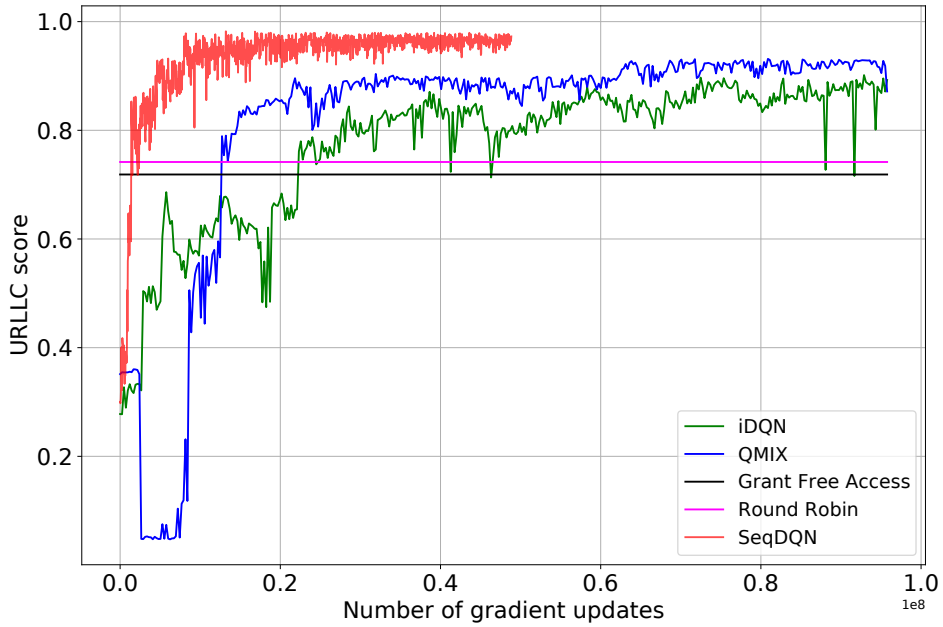
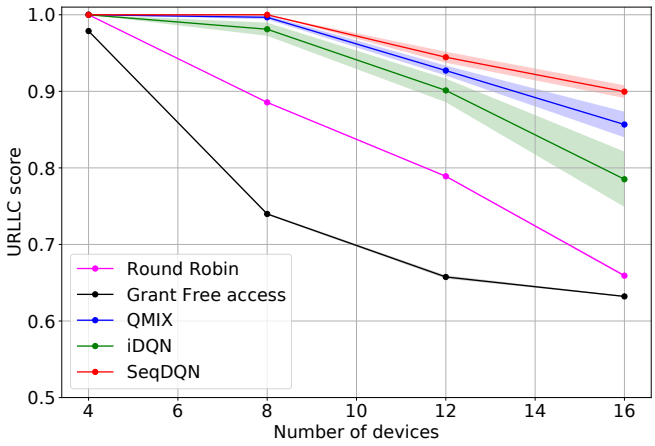
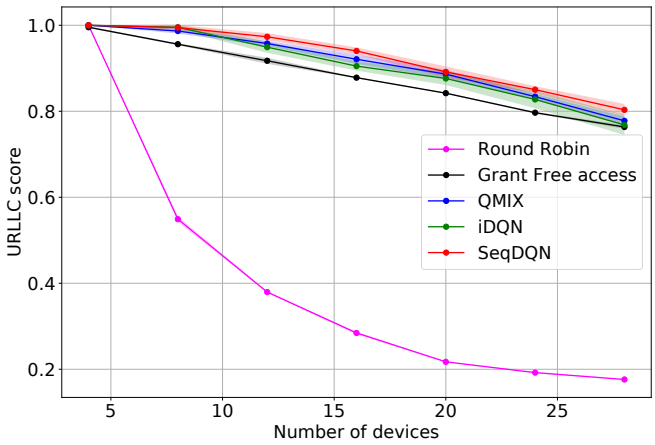


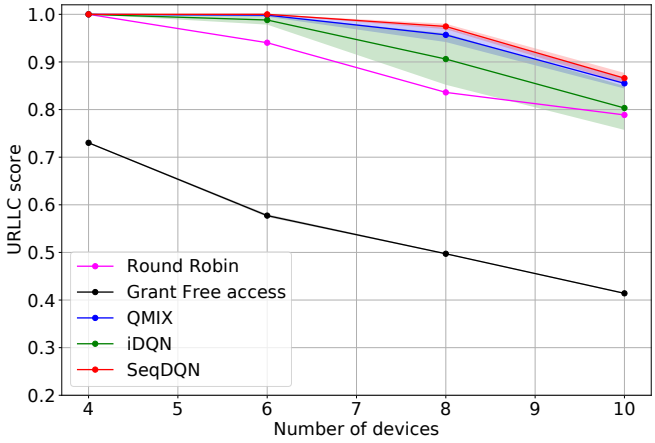
Figure 6: Évolution du score URLLC au cours de l'entraînement pour un trafic périodique probabiliste dense avec 12 utilisateurs.



(a) Trafic probabiliste périodique dense.



(b) Trafic probabiliste périodique clairesemé.



(c) Trafic déterministe périodique.

Figure 7: Évolution du score URLLC en fonction du nombre d'appareils pour différents modèles de trafic.

Chapitre 6: Multi-Agent PPO pour l'Accès Dynamique Multi-Canal URLLC

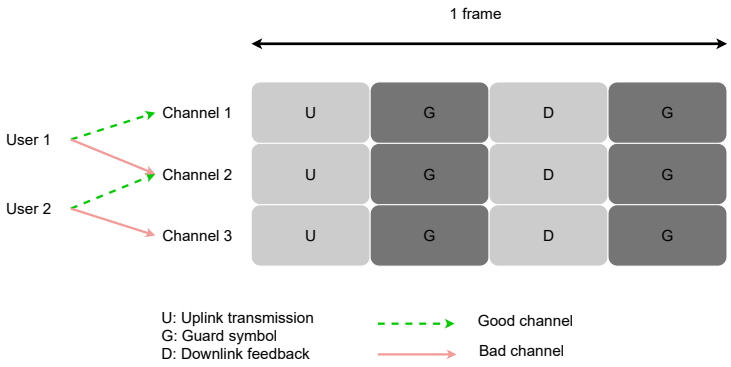


Figure 8: Modèle de système et structure de trame

Dans ce chapitre, nous développons l'étude de la liaison montante URLLC introduite dans le chapitre 5, en étendant son application à l'Accès Dynamique Multi-Canal (DMCA). Reconnaissant les défis posés par les modèles de trafic apériodiques aux algorithmes d'apprentissage multi-agents renforcé (MARL), nous exploitons la large bande passante de 40 MHz disponible dans les scénarios IoT industriels, tout en concervant une probabilité d'erreur URLLC (Fig. 8). En divisant cette bande passante en plusieurs sous-canaux de fréquence orthogonaux, nous enrichissons le modèle de système, fournissant un environnement plus complexe et diversifié pour que les algorithmes MARL expriment leur potentiel. Après avoir passé en revue la littérature sur le DMCA, nous avons constaté que les recherches existantes n'ont pas encore pleinement résolu les complexités du DMCA dans les réseaux URLLC, en particulier dans des conditions de canaux hétérogènes variables dans le temps et de profils de trafic divers. Pour combler cette lacune, nous introduisons une approche innovante basée sur le MARL. Notre méthodologie tire parti du cadre théorique de TRPO dans un contexte multi-agents pour répondre aux défis et exigences spécifiques du problème URLLC-DMCA. Dans ce chapitre, nos contributions sont les suivantes :

- Nous formulons un problème de DMCA dans un réseau URLLC avec des

utilisateurs hétérogènes qui doivent livrer un court paquet dans un délai strict en liaison montante, comme un Dec-POMDP.

- Nous considérons un cadre général où les paquets sont générés selon un trafic soit probabiliste apériodique, soit probabiliste périodique. Dans ce contexte, les utilisateurs observent plusieurs sous-canaux orthogonaux variant dans le temps selon le modèle Gilbert-Eliott.

- Nous introduisons deux solutions PPO pour résoudre le Dec-POMDP. La première, MCA-PPO, est justifiée théoriquement et bénéficie de la garantie d'amélioration monotone. La seconde, MCA-iPPO, est une approche entièrement décentralisée qui, bien qu'elle manque de garanties théoriques rigoureuses, montre de bons résultats empiriques et offre une procédure d'entraînement simplifiée.

- Enfin, nous validons la supériorité des méthodes proposées sur différents scénarios. Nos résultats surpassent systématiquement les références traditionnelles d'accès multiple et de DRL.

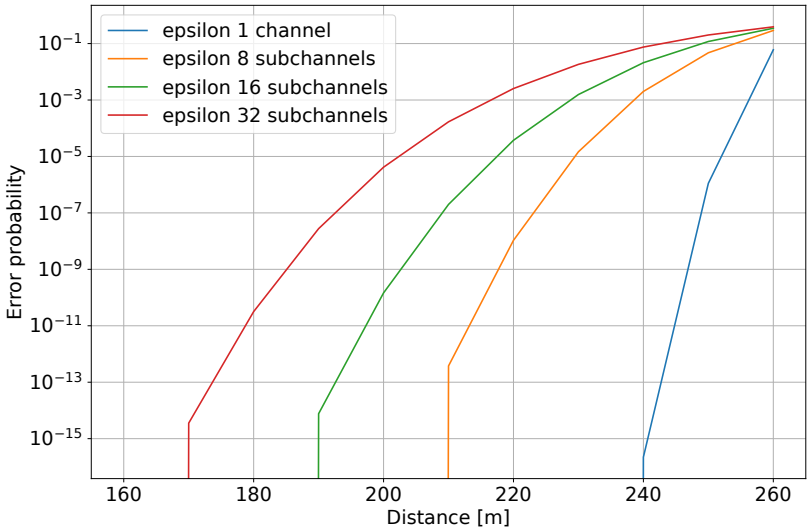
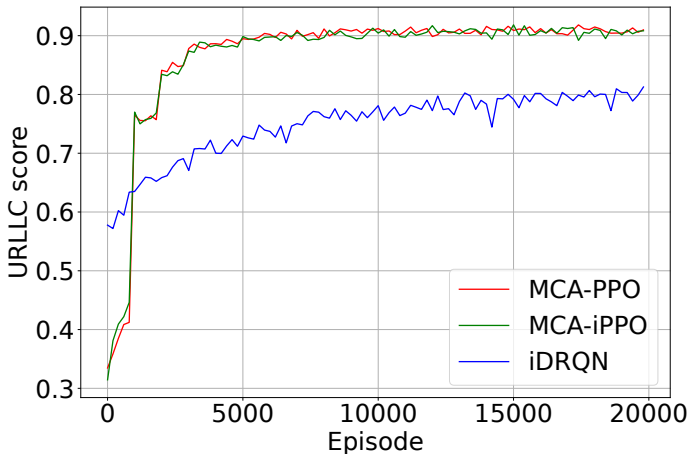
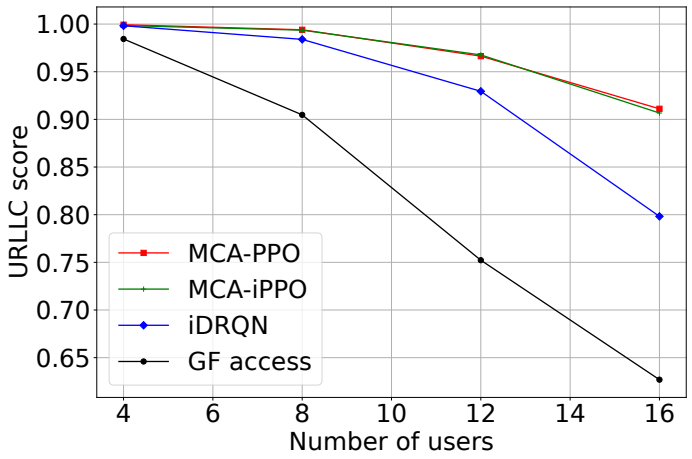


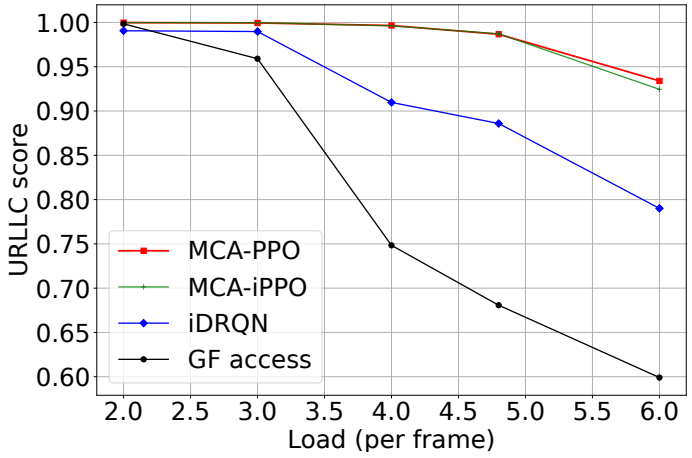
Figure 9: Probabilité d'erreur de paquet en fonction de la distance à la BS.



(a) Entrînement de 16 appareils homogènes.



(b) Scénario homogène



(c) Scénario hétérogène.

Figure 10: Evolution du score URLLC en fonction (a) du nombre d'épisodes d'entraînement; (b) du nombre d'utilisateurs; (c) de la charge par trame.

Chapitre 7: Conclusion et Perspectives Futures

Nous concluons la thèse avec ce chapitre et présentons des pistes pour étendre ce travail. Ces dernières peuvent être de court ou de long terme.

Perspectives Court Terme

- Tester et appliquer les algorithmes de cette thèse dans des scénarios plus réalistes, pour augmenter le réalisme et l'applicabilité de la recherche.
- Intégrer des mécanismes de contrôle de puissance pour les dispositifs, alignés avec l'approche décentralisée de notre travail.
- Appliquer les algorithmes MCA-PPO au problème de l'Accès Aléatoire Moderne, avec un focus sur la résolution de collisions.
- Étendre l'application de NOMA-PPO à un cadre multi-agents, explorant l'interaction et la coordination entre plusieurs BSs pour améliorer la capacité URLLC.

Perspectives Long Terme

- Explorer le paradigme des agents en réseau permettant une communication locale entre dispositifs IoT grâce à la technologie de communication de dispositif à dispositif (D2D), prometteuse pour répondre aux exigences des réseaux 5G, notamment dans les contextes mMTC et URLLC.
- Étudier le régime de champ moyen, où l'impact sur chaque agent provient d'une mesure globale basée sur l'ensemble des autres agents. Ce principe est particulièrement pertinent pour les environnements peuplés d'un grand nombre d'appareils.
- S'orienter vers le cadre de la multi-connectivité, bénéfique pour améliorer la couverture et la fiabilité en URLLC en permettant aux dispositifs de transmettre des données dupliquées à plusieurs BS, tout en relevant les défis de la coordination des transmissions et de gestion des interférences.

Chapter 1

Introduction

1.1 Context: New Challenges of the Internet-of-Things

The Internet-of-Things (IoT) refers to the interconnected network of physical devices, vehicles, sensors, software, and other technologies to collect, exchange, and act on data over the Internet. They are expected to revolutionize the digital landscape, envisaged as an omnipresent network of billions of devices, ranging from smartphones to drones and industrial sensors [Bockelmann et al., 2018]. This massive interconnected ecosystem promises to bring in a myriad of disruptive applications that will transform many aspects of our daily lives, such as drone-based services, smart grids, healthcare, home automation, industrial monitoring, and the development of smart cities [Chen et al., 2018, Mozaffari et al., 2016, Dileep, 2020, Vishwakarma et al., 2019]. Many of these applications have stringent latency constraints. For example, an IoT sensor might monitor indoor temperatures and also need to report extreme events, such as fires, necessitating both reliable and rapid transmission. Another illustrative scenario is in industrial automation contexts [Brown et al., 2018], where IoT sensors in factories must communicate emergencies swiftly and reliably to ensure immediate responses. Such communication requirements are characterized by the Third Generation Partnership Project (3GPP) standard [3GPP, 2016] as Ultra Reliable Low Latency Communications (URLLC). A classical URLLC reliability requirement is for example to transmit a 32-byte packet with success probability $1 - 10^{-5}$ and with a latency deadline of 1 ms [3GPP, 2016]. Furthermore, a deadline is said to be *strict* if the packet is lost beyond this delay. Additional URLLC use cases are illustrated in Figure 1.1. In addition, IoT devices are expected to have very limited wireless communication resources because of their limited battery

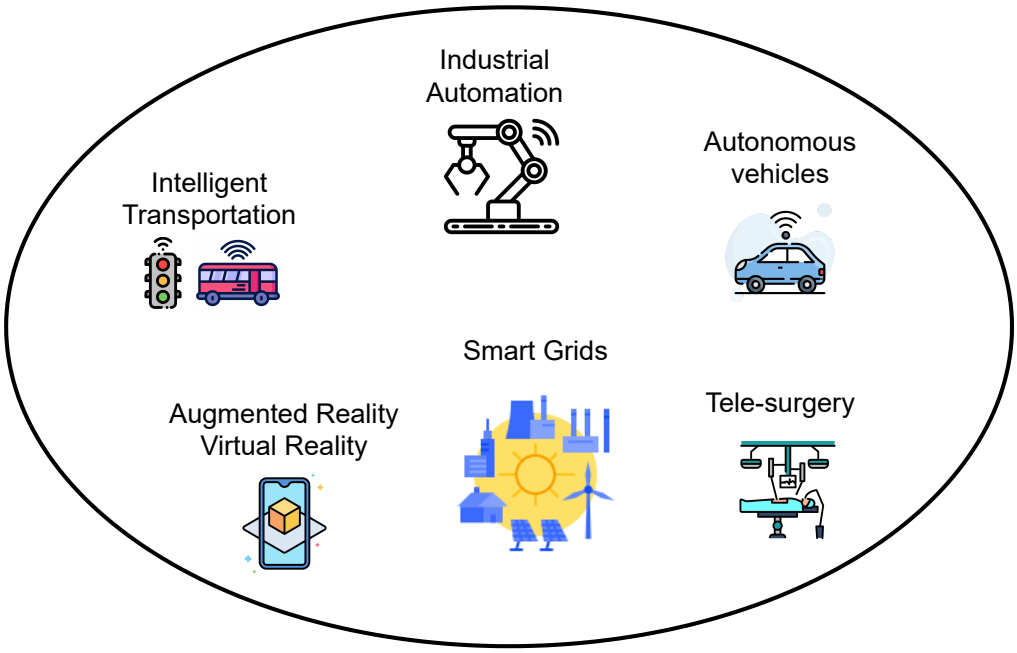


Figure 1.1: Potential URLLC use cases

life [Hägerling et al., 2014] and thus need to have minimal interaction with a Base Station (BS) to operate. Therefore, Multiple Access (MA) protocols need to operate autonomously with minimal control signalling in this type of systems, commonly known as massive Machine Type Communications (mMTC) systems.

Nevertheless, traditional protocols often fall short in meeting these stringent requirements, particularly on the uplink, i.e., from IoT devices to a central BS, because the BS can acquire traffic and channel information only at the cost of a significant signalling load and delay. As highlighted by the authors in [Chen et al., 2018], one of the primary sources of latency in cellular networks is the link establishment, including grant acquisition or random access, between a user and the BS. This process can account for delays of up to a notable 10 ms. Deterministic access protocols, such as Time Division Multiple Access (TDMA) [Miao et al., 2016], Code Division Multiple Access (CDMA) [Simon et al., 1994], and Orthogonal Frequency Division Multiple Access (OFDMA) [Hanzo et al., 2005], are particularly susceptible to this limitation. Specifically, TDMA assigns distinct time slots to each user, while CDMA allocates unique codes for each user, enabling concurrent transmissions within the same frequency band. OFDMA, a core component of Fourth Generation (4G) and Fith Generation (5G) technolo-

gies, divides the frequency spectrum into individual sub-carriers for each user. In addition to the latency used to allocate the time, frequency, or code resources, these methods struggle with scalability and fail to accommodate a vast number of devices. Moreover, their deterministic nature makes them suboptimal for IoT traffic, which is often characterized by its sporadic and unpredictable patterns.

An alternative to deterministic access protocols is random access methods, based on ALOHA [Roberts, 1975] or Carrier Sense Multiple Access (CSMA) [Bianchi, 2000]. A distinct advantage of these random access techniques is that they circumvent the handshake protocol, thereby potentially reducing initial communication delays. However, these methods have notable drawbacks when applied to URLLC scenarios. As the traffic load intensifies, the likelihood of collisions — where multiple devices attempt to transmit simultaneously — significantly increases, leading to retransmissions and, consequently, additional latency. Over the years, significant research has been invested in refining these methods by optimizing parameters like access probabilities and backoff timers with the objective to minimize collisions. The primary goal has been to either maximize throughput [Rajagopalan et al., 2009] or optimize a specific utility function, a notable example being the alpha-fairness function [Mo and Walrand, 2000]. Yet, despite these enhancements, these methods have notable drawbacks when applied to URLLC scenarios. The CSMA protocol is less effective in expansive environments with many obstacles and distant users. This limitation arises because CSMA depends on sensing the medium, a process that becomes unreliable in these conditions due to signal attenuation and propagation delays. Given these limitations, ALOHA and CSMA, still struggle with latency issues caused by collisions and retransmissions and, consequently, fail to meet the rigorous latency and reliability standards set by the URLLC constraint.

1.2 Uplink URLLC Access Solutions

Uplink communication presents unique challenges compared to downlink, especially in meeting URLLC requirements. In uplink scenarios, the traffic dynamics are characterized by multiple devices, often IoT-based, attempting to transmit data to a BS concurrently. This simultaneous transmission can lead to potential congestion and requires devices to coordinate in a decentralized manner, ensur-

ing minimal communication overhead. Conversely, in downlink scenarios, the BS acts as a central controller, broadcasting data to individual devices. Given its centralized nature, the BS can adeptly manage and allocate resources. Another difference lies in the power dynamics. While the BS in downlink operations benefits from a consistent and robust power source, uplink devices, especially IoT devices, struggle with stringent power constraints, making reliable transmissions a challenge. These varying constraints are intrinsically linked to the specific applications of each communication type. On the one hand, uplink communications often serve industrial automation systems, where sensors relay critical data to central controllers. On the other hand, downlink communications are used for applications like remote surgery and augmented reality. Therefore, given these distinctions, the approaches to tackle downlink and uplink URLLC communications differ significantly [Bennis et al., 2018]. In this thesis, we exclusively study uncoordinated MA for URLLC in the uplink context.

Uplink access schemes for URLLC can be divided in two main groups, namely grant-based and grant-free protocols.

In the first set, the scheduling of the devices is performed by the BS, see e.g. [Cuozzo et al., 2022, Nomeir et al., 2021]. Devices with a packet to transmit first send a scheduling request on the uplink. The BS then allocates uplink resources for the packet transmission. Uplink packets may include in their header some scheduling information (like the buffer status) to avoid the scheduling request step. In this case, a scheduling algorithm is required at the BS to meet the delay and reliability constraints without losing resources when a polled device has no packet to transmit. This is the baseline protocol adopted in 5G New Radio (NR) [3GPP, 2017b]. The main drawback of the approach lies in the duration of the four-way handshake that may be incompatible with URLLC constraints as highlighted in the previous section. The advantage of the scheduling, though, is to avoid interference between device transmissions and reduce the latency resulting from retransmissions. However, one can leverage the scheduling benefits while avoiding the handshake delay. In this framework, the BS can "intuitively" schedule devices, without having any real time information regarding their buffer status or channel conditions. Thus the polled devices transmit if they have a packet ready for transmission. This approach eliminates the need for users to seek transmission permission, and the BS can effectively coordinate

resource allocation and manage interference. Nevertheless, the limitation of this method, is the BS's lack of insight into buffer and channel specifics, potentially leading to inefficient resource use by scheduling users with either an empty buffer or an unfavorable channel condition. We address this challenge by proposing two Deep Reinforcement Learning (DRL) algorithms in Chapters 3 and 4.

In the second set of access schemes, the handshaking is removed by allowing uplink transmissions to be Grant-Free (GF). This means that devices can transmit without an explicit command from the BS. We can further distinguish contention-free and contention-based GF access. In contention-free GF (also called semi-persistent scheduling), the BS pre-allocates periodic orthogonal uplink resources to the devices, so that there are no collisions [Feng et al., 2019]. When a device has a packet to send, it waits for the next opportunity. This access scheme has been also adopted by 5G NR [3GPP, 2017b]. Contention-free GF is however mostly adapted to periodic deterministic traffic, and becomes inefficient when the traffic is sporadic or probabilistic because resources may be lost, if there is no packet to be sent, or deadlines violated, when the packet arrival rate is suddenly higher.

Several papers have studied contention-based GF, a family of protocols that are versions of Slotted ALOHA (SA) enriched with smart retransmission schemes. Contrary to other approaches, uplink transmissions are indeed here subject to collisions. A typical example of this literature is the work presented in [Elayoubi et al., 2019], where authors adapt SA to URLLC and industrial IoT use cases by introducing retransmission schemes that depend on the traffic profile of the devices. In [Mahmood et al., 2019a], authors summarize the classical retransmission schemes: the K-repetition GF scheme, in which a pre-determined number of copies of the same packet are transmitted; the reactive GF scheme, in which devices receive a feedback from the BS for every transmission; and the proactive GF scheme, in which a packet is repeatedly sent until a positive acknowledgement is received. A blind retransmission scheme has also been proposed [Abreu et al., 2018] where devices retransmit on a shared resource without waiting for a feedback. Yet, these techniques still struggle to simultaneously satisfy the reliability and latency requirements under random traffics as the optimal parameter settings may vary over time. To enhance these techniques, the study by [Mahmood et al., 2019a] suggests using repetitions with hybrid allocations. In this

approach, the initial transmission occurs over a dedicated resource, while retransmissions use a shared resource pool. Another popular strategy is the use of the Non-Orthogonal Multiple Access (NOMA) technology [Saito et al., 2013]. This innovative transmission technique is based on Successive Interference Cancellation (SIC) and allows the scheduling of multiple users on the same time-frequency resource and therefore improves the spectral efficiency. Thus, NOMA has been considered to enhance the GF access protocols with the goal of better using the available resources and reduce the number of collisions for a given traffic load, see e.g. [Mahmood et al., 2019a, Tegos et al., 2020, Shahab et al., 2020] and references therein. However, with or without NOMA, all SA-based approaches suffer from high collision rates when the load or the number of devices increases [Liu et al., 2020] and fail to take advantage of the various traffic patterns or channel conditions across the devices. In Chapters 5 and 6, we introduce two innovative DRL algorithms tailored to enhance collision management in GF schemes. Additionally, in Chapter 4, we present an algorithm that incorporates the NOMA in a DRL-based protocol.

Another family of protocols that leverage the SIC decoding technique is modern random access [Berioli et al., 2016]. The idea is to allow devices to transmit several replicas of their packet within a single frame to the BS. The BS then addresses collisions using an iterative decoding process based on SIC. The main challenge of this approach is to derive the optimal number of copies each user should transmit within a frame. The two most studied approaches are Contention Resolution Diversity Slotted Aloha (CRDSA) [Casini et al., 2007], that produces two replicas of the same packet at random slots within a single radio frame; and Irregular Repetition Slotted Aloha (IRSA), a protocol that generates multiple replicas based on a pre-defined probability distribution, which is the same for all users. The transmission slots for these copies are selected uniformly at random. Nonetheless, a significant limitation in adapting modern random access protocols for URLLC is the extended frame length. This issue arises, as replicas are sent in the time domain, necessitating frames composed of numerous slots. Therefore, this frame structure is not adapted to the stringent latency requirements of URLLC environments, where rapid communication is crucial.

A promising approach to enhance the URLLC capacity is the multi-frequency channel access, that divides the bandwidth into multiple orthogonal subchannels.

This method is particularly advantageous in uplink URLLC scenarios, where data packets are assumed to be typically small [3GPP, 2018b], and the available bandwidth quite large [3GPP, 2017a]. A user is thus able to transmit packet replicas over the different subchannels without introducing delay costs, effectively leveraging the wide bandwidth. However, a trade-off exists: while this approach improves resource utilization, the allocation of less bandwidth per transmission might reduce robustness and decrease the decoding probability. Each packet transmission, having a smaller bandwidth portion, may become more susceptible to errors and interference, underscoring the need for careful balance in resource allocation to maintain reliability in URLLC systems. This technology is studied in Chapter 6, where it is combined with DRL in a GF access framework.

An additional key feature in modern wireless communication systems with potential to satisfy URLLC requirements is *multi-connectivity*. It enables a device to maintain simultaneous connections to multiple network nodes, typically BSs. Its integration into URLLC networks could significantly enhance reliability by allowing devices to transmit packet replicas to various BSs and thus improve reliability [Mahmood et al., 2019b, Segura et al., 2022, Kesava and Mehta, 2022].

The last technology that could significantly improve the development of URLLC solutions is the massive Multiple-Input Multiple-Output (MIMO) technology. This technique uses multiple antennas on both transmission and reception ends to enhance reliability and spectral efficiency [Biglieri et al., 2007]. Moreover, massive MIMO has been instrumental in augmenting GF access protocols, optimizing them for the stringent requirements of URLLC networks [Ding et al., 2021]. This solution will not be included in the scope of the thesis.

The challenges of the MA problem for URLLC networks can be summarized as follows:

- **Centralized Protocols**: These protocols are attractive for their capability to coordinate devices, especially in avoiding collisions. However, they introduce high communication overhead, leading to substantial latency due to the information exchange required for device coordination. The primary challenge here is: *How can we develop centralized protocols for URLLC networks that bypass the latency inherent in link establishment?*

- **GF Protocols**: Emerging as potential contenders for URLLC, these protocols circumvent the latency associated with the four-way handshake pro-

tocol. Yet, they introduce an important latency, inherent to decentralized protocols, due to collisions and retransmissions. The question then arises: *How can we design decentralized, GF protocols that effectively mitigate collision?*

- **Device Heterogeneity and Dynamic Environments**: The vast majority of protocols struggle to handle the heterogeneity in device traffic and channel conditions. They also often fall short in optimally managing sporadic traffic and dynamic environments. This leads to the following challenge: *How can we design protocols tailored to leverage device heterogeneity and adapt to dynamic environments effectively?*

- **New Physical Layer Techniques**: Recent techniques such as NOMA or the SIC technology have emerged as promising solutions to address the URLLC constraint. However, they bring their own unique challenges such as the user selection problem and managing interference in this new paradigm. This leads to the problematic: *How can protocols be adapted to navigate the increased complexity introduced by these new physical layer techniques?*

In this thesis, we will tackle these challenges by exploring a new generation of "intelligent algorithms", leveraging the latest advancements in Machine Learning, specifically DRL. Such algorithms have already shown significant promise for IoT applications [Al-Garadi et al., 2020] and are promising to deal with the URLLC constraint in the MA problem.

1.3 Reinforcement Learning for Uplink Access

DRL has emerged as a powerful tool in addressing complex problems in wireless communications, particularly in the context of uplink multiple access challenges. These applications can be broadly categorized into two different approaches: Single-Agent Reinforcement Learning (SARL) and Multi-Agent Reinforcement Learning (MARL).

Single-Agent Reinforcement Learning

SARL can be used in the literature to optimize the parameters of the MA protocols when the optimization problem is intractable. One of the main parameters that researchers have tried to optimize is the transmit power. [Neto et al., 2021] propose a power control framework for 5G networks, using SARL at the BS. This algorithm aims to enhance total data rate and reduces neighbor cell interference in the uplink channel by adjusting the power level of the user equipment. Regarding NOMA systems, [Ahsan et al., 2022] develop an intelligent resource allocation scheme for uplink, utilizing SARL. This solution dynamically allocates users and balances resources in order to maximize the average long-term sum rate. Moreover, in massive URLLC networks, [Liu et al., 2021] optimizes GF Access parameters such as the number of repetitions or the number of resources in order to maximize the number of successfully decoded users. Finally, [Ayoub et al., 2021] learn an optimal degree distribution thanks to DRL in IoT networks.

While the optimization of parameters through SARL has shown promise in enhancing network performance, it does not fully address the fundamental challenges inherent in the underlying MA protocols. Specifically, issues such as collisions in GF access and the latency of handshake protocols in centralized schedulers remain unresolved. An innovative solution of SARL is to model the BS as a partially observable scheduler, a method that we are among the first to present.

Multi-Agent Reinforcement Learning

The other DRL framework that has been wildly used in wireless communication networks is the MARL framework. A notable application of MARL is seen in Dynamic Spectrum Access (DSA) where secondary users employ learning techniques to derive transmission protocols. The effectiveness of MARL algorithms in DSA has been proved by the work of [Chang et al., 2018, Xu et al., 2020, Xu et al., 2018, Tan et al., 2021, Kassab et al., 2020], showing the ability of agents to first coexist with established protocols like TDMA and ALOHA, and to learn throughput-optimal strategies in such environments. Additionally, the work of [Yang et al., 2020] transforms the URLLC constraint into a data rate constraint, thus enabling the learning of transmission strategies that aim to maximize the network energy efficiency. Finally, the study by [Lin et al., 2020] introduces a

multi-connectivity application for their MARL algorithm, enabling users to collaborate with neighboring devices. This collaborative approach is designed to optimize the selection of a BS, with the objective of maximizing the number of successful transmissions in the network.

Even if several studies have employed MARL to address the MA problem, they often overlook key scientific constraints inherent to this problem and directly apply the DRL algorithms. Specifically, they tend to disregard the non-stationarity resulting from the concurrent learning of the multiple agents and the partial observability issues arising from users only having access to their own buffer and channel states.

In conclusion, only a small number of previous studies have explored the application of both single-agent and multi-agent DRL in this field due to the particular challenges involved in URLLC communications.

1.4 Contributions and Structure of the Thesis

In this thesis, we explore the application of SARL and MARL for learning efficient transmission protocols in an industrial IoT scenario requiring URLLC.

We consider the MA problem where several heterogeneous devices have to transmit to a BS on the uplink. In this study, we model the stringent latency requirement with strict deadlines. Packets not delivered within this time constraint are discarded. Yet, they can be re-transmitted until their deadline is reached. We examine both periodic [Hou and Kumar, 2013] and aperiodic traffic [3GPP, 2018b] for packet generation in order to cover the majority of the factory automation use cases of the 3GPP standards. For the channel model, we first study a perfect channel model with collision, meaning that when two or more devices try to transmit a packet, a collision occurs and none of the packets are received. In a second step, we consider a more realistic interference model in line with the 3GPP recommendations [3GPP, 2018b]. Here, packet decoding relies on the Signal-to-Interference-plus-Noise Ratio (SINR) [Salaün et al., 2020] and the finite block length regime [Ren et al., 2020].

The first contribution of this thesis is the development of a framework modeling the uplink MA problem as a centralized problem with limited communication between the BS and the devices. In this setup, a device can only transmit when

scheduled by the BS. The main advantage of this method is that it guarantees no collisions, which are inherent to random access protocols. Nonetheless, the BS lacks full observability into the devices' buffers introducing partial observability. As a consequence, it needs to learn efficiently the traffic patterns of the devices in order to maximize the throughput, as it does not know when a device has a packet ready for transmission. In order to tackle this problem, we propose FilteredPPO in Chapter 3, a scheduling algorithm where the access point is modelled as a DRL agent: it combines the Proximal Policy Optimization (PPO) algorithm [Schulman et al., 2017] with deep Recurrent Neural Networks (RNN) to handle partial observability. In addition, we incorporate invalid action masking [Huang and Ontañón, 2020] to speed up the training process and make our algorithm more efficient with large number of devices. We demonstrate the superiority of our method over traditional benchmarks in heterogeneous periodic traffics. However, the main limitation of our approach is that discernible traffic patterns must exist for the learning algorithm to effectively leverage them. In contrast, it struggles when it faces aperiodic traffic patterns.

Our second contribution extends this partially observable scheduling framework to URLLC networks by incorporating the NOMA technology in Chapter 4. This enhancement makes the framework more general, allowing it to accommodate various traffic models and a realistic interference model. In this setting, the BS has the capability to schedule any subset of users within a single time frame. The decoding of the active users then follows the SIC procedure. This formulation introduces a combinatorial scheduling challenge as the potential subsets that can be selected in each frame grow exponentially in the number of devices. We address this complex problem by introducing the notion of *agent state*. We demonstrate both theoretically and experimentally that this statistic is an efficient way to cope with partial observability. We then propose a DRL algorithm, *NOMA-PPO*, that enhances the state-of-the-art algorithm PPO [Schulman et al., 2017] with a branching policy network architecture in order to linearly manage combinatorial action spaces. The combination of the agent state and this DRL algorithm allows us to define Bayesian policies in order to incorporate prior information about the MA problem into the DRL agent [Titsias and Nikoloutsopoulos, 2018]. We numerically show that our solution outperforms traditional benchmarks in terms of URLLC performance.

The third contribution of this thesis, presented in Chapter 5, is in the application of deep MARL to our URLLC problem. We equip each device with a deep MARL algorithm in order to learn a transmission protocol by interacting with the other devices and the environment. We first assess the performance of traditional Deep Q-learning frameworks [Tan et al., 2021, Rashid et al., 2018] and then propose SeqDQN, a distributed MARL algorithm where agents update their Q-function sequentially, starting with the devices with the most stringent latency requirement. The advantages of this training method are: 1) We reduce the non-stationarity caused by multiple agents learning concurrently, which is a major drawback of Independent Learning (IL), 2) our proposed method is more scalable to a large number of agents than Centralized Training Decentralized Execution (CTDE) and 3) training is much faster than the existing MARL algorithms.

The fourth and last contribution of this thesis is about the Dynamic Multiple Channel Access (DMCA) problem in a URLLC network with heterogeneous devices that we study in Chapter 6. In this context, users monitor multiple time-varying orthogonal sub-channels. Indeed, dividing the channel in multiple orthogonal channels can enhance the URLLC capacity as the bandwidth is usually much larger than what is required to send a packet in IoT scenarios [3GPP, 2018b]. In each time frame, users select a subset of sub-channels to transmit a replica of their ready-to-send packet. The complexity of this problem arises from the need to learn distributed transmission protocols in a dynamic channel environment. We present two PPO solutions to address this problem. MCA-PPO, the first solution, is theoretically-justified and benefits from the monotonic improvement guarantee. The second one, MCA-iPPO is a fully decentralized approach. While it may not have strong theoretical foundations, it displays a good empirical performance and offers a simplified training procedure.

The thesis is organized as follows. In Chapter 2, we delve into the foundational concepts of DRL, covering both single-agent and multi-agent frameworks, providing the necessary algorithms and concepts for later chapters. Chapter 3 presents FilteredPPO, a centralized algorithm for scheduling uplink IoT traffic, that bypasses coordination latency at the cost of introducing partial observability. Chapter 4 expands the centralized approach from Chapter 3, adopting a more realistic system model in line with 3GPP guidelines for factory automation.

This chapter also tackles the URLLC constraint within this context, proposing NOMA-PPO, a DRL algorithm that is able to deal with the NOMA technology and outperform existing benchmarks in terms of URLLC performance. Chapter 5 evaluates conventional Q-learning based multi-agent state-of-the-art solutions on our MA problem, and introduces SeqDQN, a distributed method where agents sequentially update their Q-functions, reducing non-stationarity issues. Chapter 6 studies distributed protocols tailored for URLLC networks in a dynamic multi-channel setting, offering a decentralized solution with theoretical guarantees. Finally, Chapter 7 concludes the thesis and sheds light on potential directions for future research.

Notations: For a finite set X, $\Delta(X)$ denotes the set of all probability distributions over X. The indicator function is denoted $\mathbb{1}\{\cdot\}$, $diag(\cdot)$ is the diagonal operator that transforms a vector in a diagonal matrix and $\odot$ is the Hadamard product. The matrices are written in bold upper case and the vectors in bold lower case. $\langle\cdot\rangle$ refers to a tuple, $[\cdot]$ to the modulo operator and $\arg_B \min(S)$ returns a set of B elements in S having the lowest value (ties are broken at random). $k_{1:m}$ denotes an ordered subset $k_1, \ldots, k_m$ of $[\![1, K]\!]$ and $-k_{1:m}$ refers to its complement.

1.5 List of Publications

Journal Paper

- Benoît-Marie Robaglia, Marceau Coupechoux and Dimitrios Tsilimantos, *Deep Reinforcement Learning for Uplink Scheduling in NOMA-URLLC Networks*, submitted to IEEE Trans. on Machine Learning in Communications and Networking.

Conference Papers

- Benoît-Marie Robaglia, Apostolos Destounis, Marceau Coupechoux and Dimitrios Tsilimantos, *Deep Reinforcement Learning for Scheduling Uplink IoT Traffic with Strict Deadlines*, IEEE Globecom, Dec. 2021.

- Benoît-Marie Robaglia, Marceau Coupechoux, Dimitrios Tsilimantos, and Apostolos Destounis, *SeqDQN: Multi-Agent Deep Reinforcement Learning for Uplink URLLC with Strict Deadlines*, EuCNC & 6G Summit, June 2023.

- Benoît-Marie Robaglia, Marceau Coupechoux and Dimitrios Tsilimantos, *Multi-Agent Proximal Policy Optimization for Dynamic Multi-Channel URLLC Access*, submitted to ICC 2024.

Chapter 2

Background: Deep Reinforcement Learning

Contents

In this chapter we provide the formalism and background knowledge that are essential prerequisites for the rest of the thesis. In particular, Section 2.1 introduces the neural network architectures employed throughout the thesis, Section 2.2 provides an overview about single-agent reinforcement learning and finally, Section 2.3 examines the multi-agent framework and its challenges.

2.1 Deep Learning

In this thesis, we use state-of-the-art Deep Neural Networks (DNN) to model both policies and value functions. DNNs are a family of function approximators, with a large number of parameters commonly referred to as θ. DNNs are trained using backpropagation and stochastic gradient descent on mini-batches of data [Amari, 1993]. In this section, we introduce the two main network architectures that will be used in the thesis: feedforward networks [Goodfellow et al., 2016, Chapter 6] and recurrent networks [Goodfellow et al., 2016, Chapter 10].

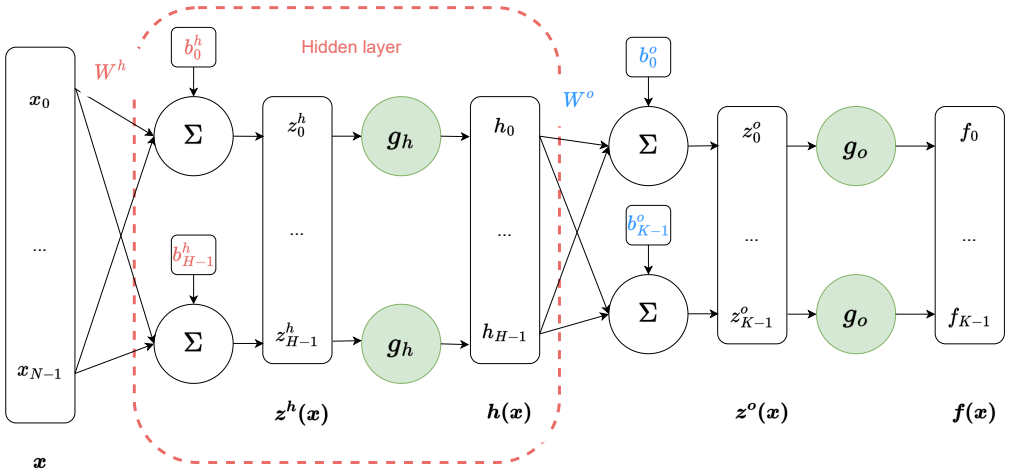


Figure 2.1: MLP with one hidden layer

2.1.1 Feed-Forward Neural Networks

Feedforward neural networks, or Multi-Layer Perceptron (MLP) are the most common neural network architecture. An MLP can be represented by a vector function $f^{MLP}(\cdot;\theta)$ parameterized by a vector of parameters θ that maps an input $\boldsymbol{x} \in \mathbb{R}^N$ to an output $\boldsymbol{y} \in \mathbb{R}^K$. An example of a MLP architecture with one hidden layer $\boldsymbol{h}$ is given in Figure 2.1. In this figure, the output vector $\boldsymbol{f}$ is obtained as follows:

- $\boldsymbol{z^h(x)} = \boldsymbol{W^h x} + \boldsymbol{b^h}$: the first operation is an affine combination of the weight matrix $\boldsymbol{W^h} \in \mathbb{R}^{N \times H}$ and bias $\boldsymbol{b^h} \in \mathbb{R}^H$ with the input vector $\boldsymbol{x}$.
- $\boldsymbol{h(x)} = g^h(\boldsymbol{z^h(x)})$: the activation function g^h transforms the latent vector $\boldsymbol{z^h(x)}$ to introduce non-linearity.
- $\boldsymbol{z^o(x)} = \boldsymbol{W^o h(x)} + \boldsymbol{b^o}$: this affine transformation combines the weight matrix $\boldsymbol{W^o} \in \mathbb{R}^{H \times K}$ and bias $\boldsymbol{b^o} \in \mathbb{R}^K$ with the hidden vector $\boldsymbol{h}$.
- $\boldsymbol{f(x)} = g^o(\boldsymbol{z^o(x)})$: the activation function g^o transforms the latent vector $\boldsymbol{z^o(x)}$ to introduce non-linearity and form the output of the neural network.

Activation functions are formally defined as element-wise operators, meaning

they are applied individually to each element rather than to the entire vector, as illustrated in Figure 2.1.

The parameters of this neural network are: (W^h, b^h, W^o, b^o). Activation functions such as ReLU or sigmoids [Karlik and Olgac, 2011] enable the MLP to capture and model complex non-linear relationships.

2.1.2 Recurrent Neural Networks

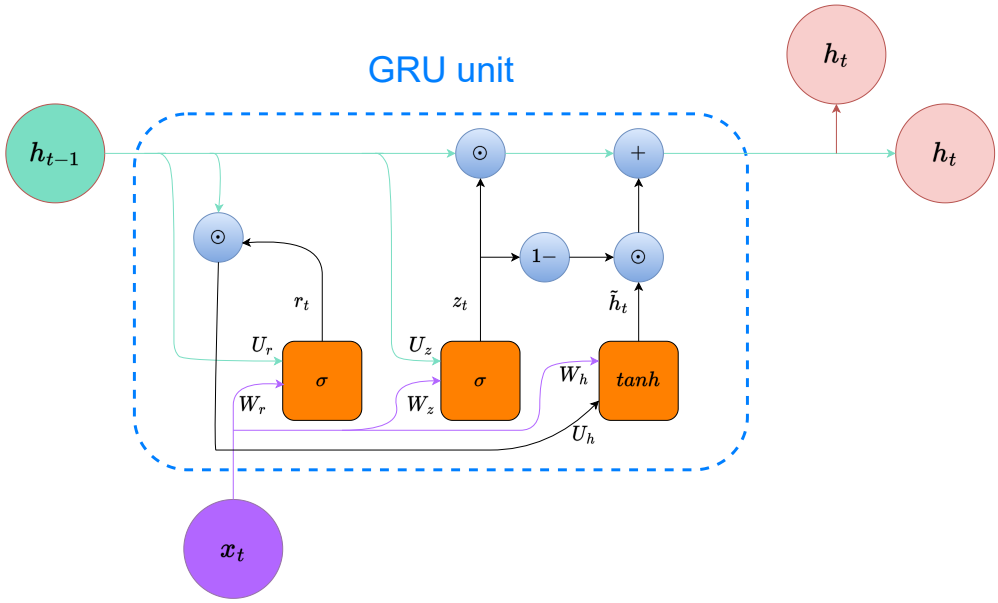


Figure 2.2: GRU cell

Another popular category of neural networks for processing sequential data is RNN [Rumelhart et al., 1986]. Unlike MLPs, RNNs use shared parameters across the sequence of inputs. This sharing mechanism, often referred to as "memory cells", enables RNNs to make connections across various parts of the input sequence.

The two main architectures of RNNs that will be used in this thesis are Long Short-Term Memory (LSTM) [Hochreiter and Schmidhuber, 1997] and Gated Recurrent Unit (GRU) [Chung et al., 2014].

The architecture of a GRU cell is illustrated in Figure 2.2. We consider a sequence of inputs of length T, $(x_1, x_2, \ldots, x_T)$. At each time t, the GRU cell produces an "activation vector" h_t based on the current input x_t and the previous activation h_{t-1}. In particular, h_t is computed as follows.

The *reset gate* r_t and the *update gate* z_t are computed using the current input x_t and the previous hidden state h_{t-1}:

$$r_t = \sigma(W_r x_t + U_r h_{t-1}) \tag{2.1}$$

$$z_t = \sigma(W_z x_t + U_z h_{t-1}) \tag{2.2}$$

where σ is the sigmoid activation function. The *candidate activation* $\tilde{h}_t$ is then computed:

$$\tilde{h}_t = tanh(W_h x_t + U_h(r_t \odot h_{t-1})) \tag{2.3}$$

Finally, the activation h_t is obtained by linear interpolation between the previous activation h_{t-1} and the candidate activation $\tilde{h}_t$:

$$h_t = (1 - z_t)h_{t-1} + z_t\tilde{h}_t \tag{2.4}$$

To summarize, while the update gate defined in equation (2.2) represents how much of the past information needs to be transmitted to future states, the reset gate (equation (2.1)) determines how much of the past information to forget. Equation (2.3) creates a candidate activation vector by combining the current input with the past hidden state, modulated by the reset gate. Ultimately, equation (2.4) defines the actual next activation by combining the previous hidden state and the candidate activation, weighted by the update gate.

Overall, the update gate chooses how much of the candidate activation vector to include in the new activation, whereas the reset gate determines how much of the previous activation to remember or forget.

The weights of the GRU cell are $(W_r, W_z, W_h, U_r, U_z, U_h)$.

While LSTM and GRU units are very similar in their ability to selectively retain or disregard information throughout time, they have a few differences [Chung et al., 2014]. The major one is their respective approach to manage the memory cell state. LSTM units maintain a memory cell state distinct from the hidden state, which is regulated by three distinct gates, called: input, output, and forget. GRU units, on the other hand, use a "candidate activation vector" instead, which is updated using only two gates: the reset gate and the update gate.

As a consequence, the GRU architecture is simpler than the LSTM, with fewer parameters, making it more computationally efficient and easier to train. Its performance is yet inferior to the LSTM, especially when the task requires modeling very long-term dependencies.

2.2 Single-Agent Reinforcement Learning

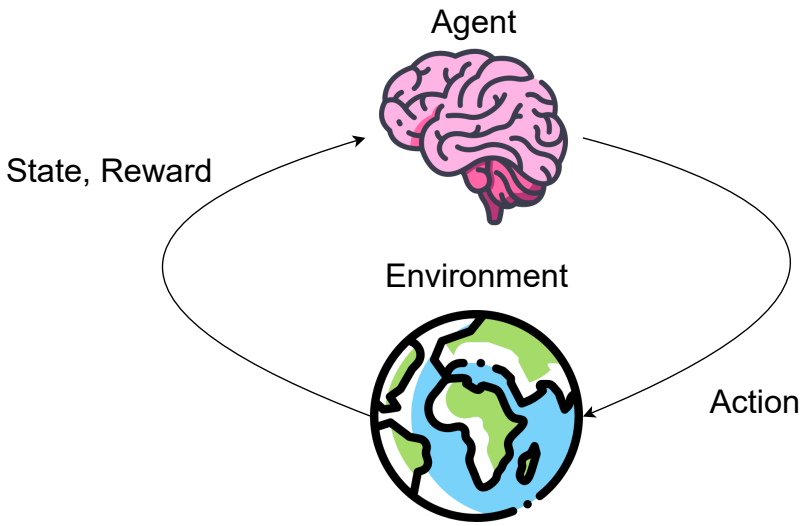


Figure 2.3: Diagram of a single-agent MDP

In SARL, an agent interacts with an environment in order to learn efficient strategies maximizing the long-term reward [Sutton and Barto, 2018].

This problem is usually formulated as a Markov Decision Process (MDP).

2.2.1 Mathematical Framework

Markov Decision Process

Definition 2.1 ([Puterman, 1990]). *(Markov Decision Process) A MDP is defined by a tuple* $(\mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{R}, \gamma)$ *where:*

- $\mathcal{S}$ *is the set of states.*
- $\mathcal{A}$ *the set of actions.*

- $\mathcal{T} : \mathcal{S} \times \mathcal{A} \to \Delta(\mathcal{S})$ *the transition probability of reaching a new state* $s' \in \mathcal{S}$ *given the previous action* $a \in \mathcal{A}$ *and state* $s \in \mathcal{S}$*. It verifies the Markov property:* $s' \sim \mathcal{T}(\cdot|s(t) = s, a(t) = a)$*,*

- $\mathcal{R} : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to \mathbb{R}$ *the reward function that an agent gets from the environment for reaching a new state* s' *after taking an action* a *in a state* s*.*

- $\gamma \in [0, 1)$ *the discount factor that models the agent's preference for short-term rewards over long-term ones.*

Definition 2.2. *We define a trajectory* $\tau = (s_0, a_0, s_1, a_1, \dots s_{T-1}, a_{T-1}, s_T)$ *such that at each time* $t \in [0, T-1]$*, the agent obverses the state* s_t *and makes the action* a_t*. The environment then transitions to the next state* $s_{t+1} \sim \mathcal{T}(\cdot|s_t, a_t)$ *and returns the instantaneous reward* $\mathcal{R}(s_t, a_t, s_{t+1})$*.*

In this thesis, we consider finite horizon MDP ($T < \infty$) with finite state and action sets.

The goal of the Reinforcement Learning (RL) agent is to find a policy $\pi \in \Pi : \mathcal{S} \to \Delta(\mathcal{A})$ that maximizes the expected sum of future discounted rewards:

$$\mathbb{E}_{s_{t+1} \sim \mathcal{T}(\cdot|s_t, a_t), a_t \sim \pi(\cdot|s_t)} \left[\sum_{t \geq 0} \gamma^t \mathcal{R}(s_t, a_t, s_{t+1}) | a_t \sim \pi(\cdot|s_t), s_0) \right] \tag{2.5}$$

For convenience, let $R(\tau) = \sum_{t=1}^{T} \gamma^t \mathcal{R}(s_t, a_t, s_{t+1})$ the sum of discounted rewards obtained on a trajectory τ. $R(\tau)$ is also called the return.

Definition 2.3. *(Value and advantage functions) [Puterman, 1990] Given a MDP* $(\mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{R}, \gamma)$ *and a policy* π*, we define the state-action function* $Q^\pi : \mathcal{S} \times \mathcal{A} \to R$ *(also called the Q-function), the value function* $V^\pi : \mathcal{S} \to R$ *and the advantage function* A^π*:* $\mathcal{S} \times \mathcal{A} \to R$ *as:*

$$Q^\pi(s, a) = \mathbb{E}_{\tau \sim (\pi, \mathcal{T})}[R(\tau)|s_0 = s, a_0 = a], \forall s \in \mathcal{S}, a \in \mathcal{A} \tag{2.6}$$

$$V^\pi(s) = \mathbb{E}_{\tau \sim (\pi, \mathcal{T})}[R(\tau)|s_0 = s], \forall s \in \mathcal{S} \tag{2.7}$$

$$A^\pi(s, a) = Q^\pi(s, a) - V^\pi(s) \tag{2.8}$$

where $\mathbb{E}_{\tau\sim(\pi,\mathcal{T})}$ is the expectation under the distribution of the probability measure over the set of trajectories and over the policy.

The Q-function $Q^\pi(s,a)$ represents the expected return from taking action a in the state s and the value function represents the expected return of being in a state s and following the policy π. The advantage function describes the relative value of an action with respect to the value of the state (how much better or worse it is to take this action).

An optimal policy π^* exists in a finite MDP with finite horizon [Puterman, 1990] and can be obtained by dynamic programming [Bertsekas, 2012], typically with value iteration or policy iteration algorithms. However, this family of algorithms assumes that the model of the environment (transition probability and reward function) is known. RL is an alternative solution that is able to learn an optimal policy without knowing the model, only based on the experiences collected by interacting with the environment. Furthermore, when facing high-dimensional state and action spaces, RL famously suffers from the "curse of dimensionality" [Sutton and Barto, 2018]. DNNs can be used to overcome this problem and can be combined with RL to handle high-dimensional MDPs [Mnih et al., 2015].

Partially Observable Markov Decision Process

In certain scenarios, an agent may only have access to a partial observation of the environment, rather than the full environmental state. This situation is modeled through a framework known as Partially Observable Markov Decision Process (POMDP) e.g. [Sondik, 1971, Kaelbling et al., 1998].

Definition 2.4 (POMDP). *A POMDP can be described by a tuple ($\mathcal{S}$, $\mathcal{A}$, $\mathcal{T}$, $\mathcal{R}$, Ω, $\mathcal{O}$, γ), where*

- *$(\mathcal{S},\mathcal{A},\mathcal{T},\mathcal{R},\gamma)$ is an MDP,*
- *Ω is the observation space, i.e., a finite set of observations,*
- *$\mathcal{O}:\mathcal{S}\times\mathcal{A}\mapsto\Delta(\Omega)$ is the probability distribution of the observation o when the environment is in state s' and the agent has taken action a: $o\sim\mathcal{O}(\cdot|s_{t+1}=s',a_t=a)$.*

The *history* $\hbar_t$ at time t is defined as the sequence of actions taken by the agent and observations from the environment $\hbar_t = \{o_0, a_0, o_1, a_1, ..., a_{t-1}, o_t\}$, where $a_t \in \mathcal{A}$ and $o_t \in \Omega$ for all t. The agent makes decisions using a stochastic *policy* π that is a distribution over the actions knowing the history.

RL approaches can be classified in two categories: *value-based* and *policy-based* methods.

2.2.2 Value-Based Solutions

Value-based methods are a family of algorithms that focus on finding the optimal Q-function, maximizing (2.6). Given the optimal Q-function Q^*, we can derive the optimal policy, taking the *greedy action* $\pi^*(s) = \arg\max_{a \in \mathcal{A}} Q^*(s, a)$.

The most popular value-based algorithm is Q-learning [Watkins and Dayan, 1992] where the agent updates an estimate of the optimal Q-function $\hat{Q}(s, a)$ based on the Bellman equation. More concretely, let s' be the next state an agent reaches after taking an action a in a state s. The Q-learning update reads:

$$\hat{Q}(s, a) \leftarrow \hat{Q}(s, a) + \alpha \left(r + \gamma \max_{a' \in \mathcal{A}} \hat{Q}(s', a') - \hat{Q}(s, a) \right) \tag{2.9}$$

where $r = \mathcal{R}(s, a, s')$ is the immediate reward and $\alpha > 0$ the learning rate. The Q-learning converges almost surely to the optimal Q-value with finite state and action spaces [Szepesvári and Littman, 1999].

Deep Q-Networks (DQN)

With the advancement of DNNs, the authors of [Mnih et al., 2015] introduce Deep Q-Networks (DQN), where the Q-function is approximated by a DNN, called Q-network. In DQN, the decision maker selects an action a in the state s according to an ε-*greedy policy*: it selects the action that maximizes the Q-value $Q(s, a)$ with probability $1 - \varepsilon$ and chooses uniformly a random action with probability ε so that the agent keeps exploring the environment. After each step, the agent stores the system transitions (s, a, r, s') in a so called replay buffer $\mathcal{B}$ where the agent observes the next state s' after taking the action a in the state s and receiving the reward r. The Q-network's parameters θ are learnt by sampling batches of b transitions from the replay buffer and minimizing the following loss called TD-error:

$$\mathcal{L}(\theta) = \sum_{i=1}^{b} \left[\left(r_i + \gamma \max_{a'} Q(s'_i, a'_i; \theta^-) - Q(s_i, a_i; \theta) \right)^2 \right] \tag{2.10}$$

where θ^- are the parameters of what the authors call the *target network* that are used to stabilize the training procedure. These parameters are an old version of the parameters θ and are periodically updated. The pseudo-code of DQN can be found in Algorithm 1. In this pseudo-code, the function $\texttt{Uniform}(\mathcal{A})$ draws an action uniformly in the action space.

Algorithm 1: Deep Q-Networks

Initialize the replay buffer $\mathcal{B}$ to capacity C, the number of episodes M, the trajectory length T, the time at which training is starting T_s, the period of updates of the target network T_u, the timestep $t' = 0$
Initialize the Q-network with random weights $\theta \leftarrow \theta_0$, $\theta^- \leftarrow \theta_0$
for *episode* $= 1, 2 \ldots, M$ **do**
    Reset the environment and receive the first state s_1.
    for $t = 1, 2, \ldots, T$ **do**
        Increment $t' = t' + 1$
        Select an action

$$a_t = \begin{cases} \arg\max_{a \in \mathcal{A}} Q(s_t, a; \theta) & \text{with probability } 1 - \varepsilon \\ a \sim \texttt{Uniform}(\mathcal{A}) & \text{with probability } \varepsilon \end{cases}$$

        Execute a_t in the environment; observe reward r_t and the next state s_{t+1}
        Store transition (s_t, a_t, r_t, s_{t+1}) in $\mathcal{B}$ and remove the oldest value if the buffer is full.
        Set the state to $s_t = s_{t+1}$
        if $t' > T_s$ **then**
            Sample a minibatch of B transitions $\{(s_j, a_j, r_j, s'_j)\}_{j=\{1,\ldots,B\}}$ from $\mathcal{B}$
            Set $y_j =$

$$\begin{cases} r_j + \gamma \max_{a' \in \mathcal{A}} Q(s_{j+1}, a'; \theta^-) & \text{for non-terminal } s'_j \\ r_j & \text{for terminal } s'_j \end{cases} \quad \forall j$$

            Compute the loss in (2.10) and perform gradient descent to update θ
            if $(t' \mod T_u) == 0$ **then**
                Set $\theta^- = \theta$

2.2.3 Policy-Based Solutions

The second family of RL algorithms is called "policy-based". Instead of finding the optimal policy by maximizing a Q-value, these algorithms directly search over the policy space in order to find an optimal policy π^* that maximizes (2.5). In practice, π^* is approximated by a parameterized policy $\pi_\theta(\cdot|s)$ where θ are the parameters of a DNN. Thus, the Policy Gradient (PG) algorithms aim at maximizing the expected return $J(\theta) = \mathbb{E}_{\tau\sim(\pi_\theta,\mathcal{T})}[R(\tau)]$. This optimization problem is usually solved using gradient ascent: $\theta \leftarrow \theta + \alpha\nabla_\theta V^{\pi_\theta}(s_0)$. The gradient of the cumulative return is derived by the famous PG theorem [Sutton et al., 1999]:

$$\nabla_\theta V^{\pi_\theta}(s_0) = \mathbb{E}_{\tau\sim(\pi_\theta,\mathcal{T})}\left[\sum_{t=0}^{T} \nabla_\theta \log \pi_\theta(a_t|s_t) R(\tau)\right] \tag{2.11}$$

where T is the length of the trajectory τ. This theorem is derived thanks to the log derivative trick and the proof can be found in [Sutton and Barto, 2018]. The term $\sum_{t=0}^{T} \nabla_\theta \log \pi_\theta(a_t|s_t)$ can be interpreted as the maximum log likelihood, indicating how likely a given trajectory is under the current policy. Multiplying it with the rewards increases the probability of a policy that yields trajectories with high positive rewards.

In addition, one can observe that the transition function does not appear in the PG theorem, suggesting that the Markov property is not required to compute the gradient. This suggests that PG methods can be applied to a POMDP without the need for adaptation. This can be simply achieved by substituting the state s_t by the observation o_t.

In most cases, the explicit computation of this expectation is impractical due to its computational intractability. To get around this problem, we usually approximate the expectation with Monte Carlo sampling, by generating N trajectories $\{\tau_1, \tau_2, \ldots, \tau_N\}$ and then estimate the expectation by calculating the average of these trajectories as follows:

$$\nabla_\theta V^{\pi_\theta}(s_0) \approx \frac{1}{N}\sum_{i=1}^{N}\left[\sum_{t=0}^{T} \nabla_\theta \log \pi_\theta(a_{i,t}|s_{i,t}) R(\tau_i)\right] \tag{2.12}$$

This vanilla PG algorithm is called REINFORCE [Williams, 1992]. In prac-

tice, we do not compute the gradient explicitly, but we minimize the following loss function with backpropagation: $L(\theta) = -\frac{1}{N}\sum_{i=1}^{N}\left[\sum_{t=0}^{T}\log\pi_\theta(a_{i,t}|s_{i,t})R(\tau_i)\right]$.

However, vanilla PG algorithms often struggle with sample inefficiency, which can lead to convergence to local optima and large policy steps. Consequently, this creates estimators with high variance and, occasionally, a collapse in performance. Indeed, the rewards obtained in a trajectory can be very inconsistent, particularly in the early stages of training. Since the PG algorithm is directly impacted by the return $R(\tau)$, this inconsistency leads to updates characterized by high variance.

One initial improvement to enhance the efficiency of the gradient calculation involves applying the principle of causality. This principle states that actions taken at a future time t' cannot affect the rewards at an earlier time t i.e. when $t < t'$. In the context of our calculation in (2.12), this means reformulating the gradient as follows:

$$\nabla_\theta V^{\pi_\theta}(s_0) \approx \frac{1}{N}\sum_{i=1}^{N}\left[\sum_{t=0}^{T}\nabla_\theta\log\pi_\theta(a_{i,t}|s_{i,t})\sum_{t'=t}^{T}\gamma^{t'}r_{t'}\right] \tag{2.13}$$

We continue our exploration of the PG method, until the derivation of the PPO algorithm, a necessary background for the rest of the thesis.

Baselines

A popular approach to mitigate variance in the REINFORCE algorithm involves the use of baselines. This technique aims to modify the trajectory probabilities, focusing not only on trajectories with high returns, but also on those that outperform the average.

Our gradient with a baseline $b(s_t)$ becomes:

$$\nabla_\theta V^{\pi_\theta}(s_0) \approx \frac{1}{N}\sum_{i=1}^{N}\left[\sum_{t=0}^{T}\nabla_\theta\log\pi_\theta(a_{i,t}|s_{i,t})\left(\sum_{t'=t}^{T}\gamma^{t'}r_{t'} - b(s_t)\right)\right] \tag{2.14}$$

The integration of a baseline b into the policy gradient equation can be justified by showing that its inclusion does not alter the expected value of the gradient i.e. $\mathbb{E}_{\tau\sim(\pi_\theta,\mathcal{T})}\left[\nabla_\theta\log\pi_\theta(a_t|s_t)b(s_t)\right] = 0$, by linearity of the expectation.

$$\begin{aligned}\mathbb{E}_{\tau\sim(\pi_\theta,\mathcal{T})}\left[\nabla_\theta \log \pi_\theta(a_t|s_t)b(s_t)\right] &= \mathbb{E}_{s_{1:t},a_{1:t-1}}\mathbb{E}_{s_{t+1:T+1},a_{t:T}}\left[\nabla_\theta \log \pi_\theta(a_t|s_t)b(s_t)\right] \\ &= \mathbb{E}_{s_{1:t},a_{1:t-1}}\left[b(s_t)\mathbb{E}_{a_t}\left[\nabla_\theta \log \pi_\theta(a_t|s_t)\right]\right]\end{aligned}$$

Besides, as:

$$\begin{aligned}\mathbb{E}_{a_t}\left[\nabla_\theta \log \pi_\theta(a_t|s_t)\right] &= \int \frac{\nabla_\theta \pi_\theta(a_t|s_t)}{\pi_\theta(a_t|s_t)}\pi_\theta(a_t|s_t)da_t \\ &= \nabla_\theta \int \pi_\theta(a_t|s_t)da_t \\ &= \nabla_\theta 1 \\ &= 0\end{aligned}$$

we deduce that the gradient estimator with a baseline is unbiased:

$$\mathbb{E}_{\tau\sim(\pi_\theta,\mathcal{T})}\left[\nabla_\theta \log \pi_\theta(a_t|s_t)b(s_t)\right] = 0 \tag{2.15}$$

An intuitive good baseline is the expected return of being in a state s_t, which is the value function at s_t: $b(s_t) = V(s_t)$. This baseline allows us to use the advantage in the gradient ascent update as we can notice that $\mathbb{E}_{\tau\sim(\pi_\theta,\mathcal{T})}[\sum_{t'=t}^{T}\gamma^{t'}r_{t'}] = Q^\pi(s_t,a_t)$.

$$\nabla_\theta V^{\pi_\theta}(s_0) \approx \frac{1}{N}\sum_{i=1}^{N}\left[\sum_{t=0}^{T}\nabla_\theta \log \pi_\theta(a_{i,t}|s_{i,t})A^\pi(s_{i,t},a_{i,t})\right] \tag{2.16}$$

On-policy and Off-policy learning

So far, we have described PG as an on-policy algorithm, meaning that the policy being learned is also the one used to make decisions and interact with the environment. However, on-policy learning suffers from low sample efficiency because each update requires collecting new samples using the updated policy to compute the policy gradient, making previously collected samples obsolete and non-reusable.

A way to alleviate this problem is to use *importance sampling*. This method

allows us to compute the expected reward using samples collected with another policy. Let's assume that we have samples from a policy π_{old} instead of π_θ. We can write the PG objective as follows:

Let $P_\theta(\tau) = P(s_1)\prod_{t=1^T}\pi_\theta(a_T|s_t)p(s_t{+}1|s_t,a_t)$ the probability of a trajectory associated with the policy π_θ, and Q the trajectory distribution associated with the policy π_{old}. We can write:

$$J(\theta) = \mathbb{E}_{\tau\sim P_\theta(\tau)}[R(\tau)] \tag{2.17}$$
$$= \int R(\tau)P_\theta(\tau)d\tau \tag{2.18}$$
$$= \int Q(\tau)\frac{P_\theta(\tau)}{Q(\tau)}R(\tau)d\tau \tag{2.19}$$
$$= \mathbb{E}_{\tau\sim Q(\tau)}\left[\frac{P_\theta(\tau)}{Q(\tau)}R(\tau)\right] \tag{2.20}$$

Therefore, we can derive the PG with importance sampling:

$$\nabla_\theta J(\theta) = \mathbb{E}_{\tau\sim Q(\tau)}\left[\nabla_\theta\frac{P_\theta(\tau)}{Q(\tau)}R(\tau)\right] \tag{2.21}$$
$$= \mathbb{E}_{\tau\sim Q(\tau)}\left[\frac{P_\theta(\tau)\nabla_\theta\log P_\theta(\tau)}{Q(\tau)}R(\tau)\right] \tag{2.22}$$
$$= \mathbb{E}_{\tau\sim(\pi_{\text{old}},\mathcal{T})}\left[\prod_{t=1}^{T}\frac{\pi_\theta(a_t|s_t)}{\pi_{\text{old}}(a_t|s_t)}\sum_{t=0}^{T}\nabla_\theta\log\pi_\theta(a_t|s_t)R(\tau)\right] \tag{2.23}$$

Trust Region Policy Optimization (TRPO)

[Schulman et al., 2015a] introduced Trust Region Policy Optimization (TRPO), a PG algorithm that not only integrates baselines and importance sampling but also incorporates a Kullback-Leibler (KL) divergence constraint on each iteration's policy update. The idea is to update the policy by taking the largest step possible to maximize the performance, maintaining training stability through a KL divergence constraint that regulates the magnitude of policy updates. Formally, TRPO's optimization problem reads:

$$\max_\theta \mathbb{E}_{s,a\sim(\pi_{\text{old}},\mathcal{T})}\left[\frac{\pi_\theta(a|s)}{\pi_{\text{old}}(a|s)}A^{\pi_{\text{old}}}(s,a)\right] \tag{2.24}$$
$$\text{s.t. } \mathbb{E}_{s\sim\mathcal{T}}\big[KL[\pi_\theta(\cdot|s)||\pi_{\pi_{\text{old}}}(\cdot|s)]\big] \leq \delta \tag{2.25}$$

where π_{old} is the policy before the update and KL is the Kullback–Leibler divergence [Kullback and Leibler, 1951]. The advantage function $A^{\pi_{\text{old}}}$ is estimated as usual and can be computed according to several methods that can be found in [Schulman et al., 2015b]. They require a value function, that is represented by a DNN V_φ, parameterized by φ. The main benefit of TRPO is its ability to guarantee a monotonic improvement throughout policy iterations, meaning each gradient update guarantees an enhanced policy. The theoretical proof of this result is detailed in [Schulman et al., 2015a]. However, TRPO is a second order method as it requires the computation of a second order matrix when approximating the KL term which makes it computationally expensive.

Proximal Policy Optimization (PPO) is a family of PG methods introduced by [Schulman et al., 2017] that benefits from the stability and reliability of trust region methods, but with better sample complexity and a significantly simpler implementation. PPO is primarily divided into two variants: *PPO-Penalty* and *PPO-Clip*. PPO-Penalty uses a soft constraint on the KL-divergence in the surrogate objective, penalizing large deviations in terms of KL-divergence between the new policy and the old one. On the other hand, PPO-Clip does not have any constraint and applies policy ratio clipping to keep the new policy from deviating too much from the old one.

For the remainder of the thesis, we will focus on the PPO-Clip variant of the algorithm and refer to it as PPO, since it is the most often used version in the literature due to its performance stability and ease of implementation.

The PPO objective reads:

$$\mathbb{E}_{\boldsymbol{s},\boldsymbol{a}\sim(\pi_{\text{old}},\mathcal{T})}\left[\min\left(\frac{\pi_\theta(\boldsymbol{a}|\boldsymbol{s})}{\pi_{\text{old}}(\boldsymbol{a}|\boldsymbol{s})}A^{\pi_{\text{old}}}(\boldsymbol{s},\boldsymbol{a}), g(\nu)A^{\pi_{\text{old}}}(\boldsymbol{s},\boldsymbol{a})\right)\right] \tag{2.26}$$

with $g(\nu) = \text{clip}\left(\frac{\pi_\theta(\boldsymbol{a}|\boldsymbol{s})}{\pi_{\text{old}}(\boldsymbol{a}|\boldsymbol{s})}, 1-\nu, 1+\nu\right)$ and $\nu \in [0,1)$ a hyperparameter that indicates how far away the new policy can deviate from the old one. The clip operator is defined as:

$$\text{clip}(x, \text{lower}, \text{upper}) = \begin{cases} \text{lower} & \text{if } x < \text{lower} \\ \text{upper} & \text{if } x > \text{upper} \\ x & \text{otherwise} \end{cases}$$

The pseudo-code of the PPO algorithm can be found in Algorithm 2.

Algorithm 2: Proximal Policy Optimization

Initialize the policy parameters θ_0, the value function parameters φ_0, the number of episodes J.

for $j = 0, 1, 2 \ldots, J$ **do**

Collect a set of trajectories $\mathcal{D}_j$ by running the current policy π_{θ_j}.

Compute the discounted reward for every step t for every trajectory $\tau = (s_0, a_0, \ldots, s_T) \in \mathcal{D}_j$: $\hat{R}_t = \sum_{t'=t}^{T} \gamma^{t'} r'_t, \forall t \in [0, T]$

Compute $\hat{A}_t$ using V_{φ_j}

Update the policy by maximizing the PPO surrogate objective:

$$\theta_{j+1} = \arg\max_{\theta} \frac{1}{|\mathcal{D}_j|T} \sum_{\tau \in \mathcal{D}_j} \sum_{t=0}^{T} \min \left(\frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_j}(a_t|s_t)} \hat{A}_t(s_t, a_t), g(\nu) \hat{A}_t(s_t, a_t) \right)$$

with stochastic gradient ascent.

Update the value function with the mean-squared error:

$$\varphi_{j+1} = \arg\min_{\varphi} \frac{1}{|\mathcal{D}_j|T} \sum_{\tau \in \mathcal{D}_j} \sum_{t=0}^{T} \left(V_\varphi(s_t) - \hat{R}_t \right)^2$$

with stochastic gradient descent.

Other policy optimization approaches have also been wildly used in the literature. Among them, actor-critic methods, such as those proposed by [Konda and Tsitsiklis, 1999, Peters and Schaal, 2008] approximate $R(\tau)$ with a Q-function called critic Q_ω that is updated with Q-learning (2.9). Additionally, the Soft Actor Critic (SAC), introduced by [Haarnoja et al., 2018] optimizes the policy by maximizing both the expected return and the entropy of the policy, encouraging exploration and robustness. Since these methods are not employed in the thesis, we will not go into detail about them.

2.3 Multi-Agent Reinforcement Learning

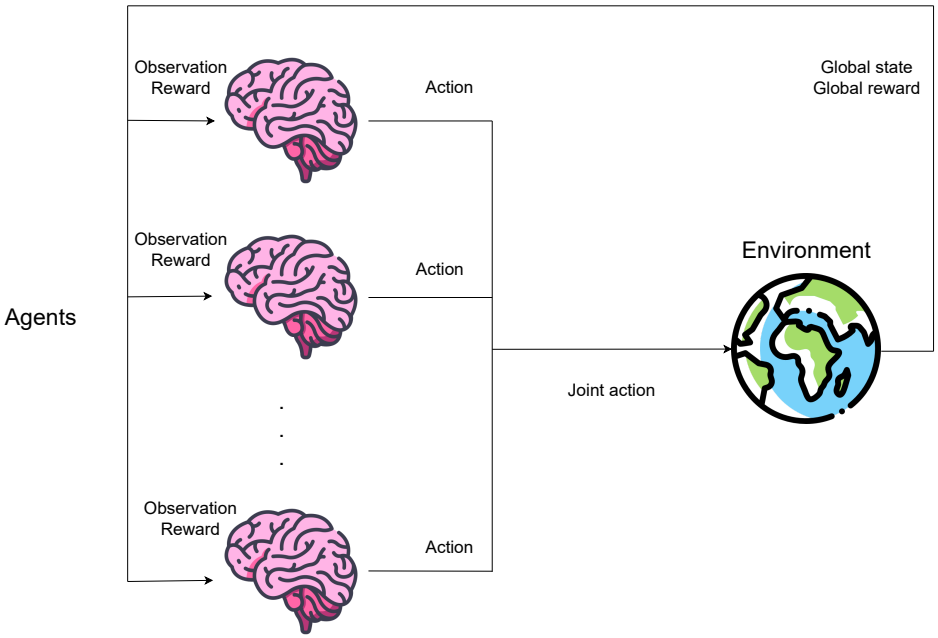


Figure 2.4: Diagram of a Markov Game

MARL is a paradigm where at least two agents concurrently interact with an environment in order to solve a sequential decision making problem. As every agent has an impact on the environment transitions and the reward function, it is a much more difficult problem to solve. This problem is usually formulated as a Stochastic Game (SG) [Shapley, 1953], also known as a Markov Game (MG) [Littman, 1994].

2.3.1 Mathematical framework

Markov Game

Definition 2.5 ([Shapley, 1953, Littman, 1994]). *(Markov Game) A MG is a tuple*
$(\mathcal{N}, \mathcal{S}, \mathcal{A} = \{\mathcal{A}^i\}_{i\in\mathcal{N}}, \mathcal{T}, \{\mathcal{R}^i\}_{i\in\mathcal{N}}, \gamma)$ *where:*

- $\mathcal{N} = \{1, \ldots, N\}$ *is the set of* $N \geq 1$ *agents.*

- $\mathcal{S}$ *the set of environmental states shared by all agents.*
- $\mathcal{A}^i$ *the set of actions of agent* i*.*
- $\mathcal{R}^i : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to \mathbb{R}$ *the reward function of agent* i*.*
- $\mathcal{T} : \mathcal{S} \times \mathcal{A} \to \Delta(\mathcal{S})$ *the transition probability function with* $\mathcal{A} = \mathcal{A}^1 \times \dots \mathcal{A}^N$ *the joint action space.*
- $\gamma \in [0, 1)$ *the discount factor.*

For an agent $i \in \mathcal{N}$, we note $-i = \mathcal{N} \setminus \{i\}$ the set of all agents except i. For example, we have $\boldsymbol{a} = (a^i, a^{-i}) \in \mathcal{A}$.

We define a trajectory $\tau = (s_0, \boldsymbol{a_0}, \dots, s_{T-1}, \boldsymbol{a_{T-1}}, s_T)$ such that at each time $t \in [0, T-1]$, the N agents observe the state s_t and simultaneously make the action $\boldsymbol{a_t} = (a_t^1, a_t^2, \dots, a_t^N)$. The environment then transitions to the next state $s_{t+1} \sim \mathcal{T}(\cdot|s_t, \boldsymbol{a_t})$ and returns the instantaneous rewards for all agents $\{\mathcal{R}^i(s_t, \boldsymbol{a_t}, s_{t+1})\}_{i \in \mathcal{N}}$.

As for a single-agent MDP, each agent $i \in \mathcal{N}$ wants to find a policy $\pi^i \in \Pi^i : \mathcal{S} \to \Delta(\mathcal{A}^i)$ that maximizes its long-term reward.

$$V^{\pi^i, \pi^{-i}}(s_0) = \mathbb{E}_{s_{t+1} \sim \mathcal{T}(\cdot|s_t, \boldsymbol{a_t}), a^{-i} \sim \pi^{-i}(\cdot|s_t)} \left[\sum_{t=0}^{T} \gamma^t \mathcal{R}^i(s_t, \boldsymbol{a_t}, s_{t+1}) | a_t^i \sim \pi^i(\cdot|s_t), s_0 \right] \tag{2.27}$$

In game theory, π^i is called a strategy. It can be a *pure strategy* (when π^i is a deterministic policy) or *mixed strategy* (when π^i is a stochastic policy). The main difference and difficulty compared to SARL, is that an agent's objective is influenced by the other agents' policies. The most common solution concept for non-cooperative MG is the Nash Equilibrium (NE) [Nash, 1951].

Definition 2.6. *(Nash Equilibrium) A strategy profile* $\pi^* = (\pi^{i,*}, \dots, \pi^{N,*})$ *is a NE of the MG* $(\mathcal{N}, \mathcal{S}, \mathcal{A} = \{\mathcal{A}^i\}_{i \in \mathcal{N}}, \mathcal{T}, \{\mathcal{R}^i\}_{i \in \mathcal{N}}, \gamma)$ *if and only if:*

$$V^{\pi^{i,*}, \pi^{-i,*}}(s_0) \geq V^{\pi^i, \pi^{-i,*}}(s_0), \forall s_0 \in \mathcal{S}, \forall i \in \mathcal{N}, \forall \pi^i \in \Pi^i \tag{2.28}$$

Conceptually, a NE is a joint policy π^* where none of the agents has any incentive to deviate, i.e, for each agent $i \in \mathcal{N}$ the policy $\pi^{i,*}$ is the best response of $\pi^{-i,*}$.

Partially Observable Setting

In real life multi-agent systems, agents do not always have access to the full environmental state but only an observation of the state. In multiple access for example, a device can observe its own buffer but not the buffers of the other devices. A stochastic game thus becomes partially observable.

Definition 2.7. *(Partially-Observable Stochastic Game (POSG)). A POSG is defined by the tuple* $(\mathcal{N}, \mathcal{S}, \mathcal{A} = \{\mathcal{A}^i\}_{i\in\mathcal{N}}, \mathcal{T}, \{\mathcal{R}^i\}_{i\in\mathcal{N}}, \gamma, \mathcal{O} = \{\mathcal{O}^i\}_{i\in\mathcal{N}}, O)$ *where* $(\mathcal{N}, \mathcal{S}, \mathcal{A} = \{\mathcal{A}^i\}_{i\in\mathcal{N}}, \mathcal{T}, \{\mathcal{R}^i\}_{i\in\mathcal{N}}, \gamma)$ *is a SG defined in Definition 2.5 and where:*

- $\mathcal{O}^i$ *is the observation set of agent* i*.*
- $O : \mathcal{S} \times \mathcal{A} \to \Delta(\mathcal{O})$ *the observation function such that* $O(\boldsymbol{o}|\boldsymbol{a}, s')$ *is the probability of observing* $\boldsymbol{o} \in \mathcal{O}$ *from the next state* s' *given the action* $\boldsymbol{a} \in \mathcal{A}$*.*

The policy of agent i *now becomes* $\pi^i \in \Pi^i : \mathcal{O} \to \Delta(\mathcal{A}^i)$*.*

Introducing partial observability in a SG drastically increases the difficulty of finding theoretically justified algorithmic solutions. A specific case of POSG is the Decentralized Partially Observable Markov Decision Process (Dec-POMDP), as described by [Oliehoek, 2012]. This is a wildly used framework in MARL where agents share the same collective reward.

Definition 2.8. *(Dec-POMDP) A Dec-POMDP is a subclass of POSG where* $\mathcal{R}^i = \mathcal{R}$ *for all agents* $i \in \mathcal{N}$*.*

Finding a solution for a finite horizon Dec-POMDP is NEXP-complete for $n \geq 2$ [Oliehoek, 2012]. NEXP is the set of problems that can be solved in a non-deterministic exponential time. "Non-deterministic" means that it requires a guess generated in a non-deterministic way and "exponential" means that it takes exponential time to verify that the guess is a solution.

2.3.2 Independent Learning

Introduced by [Tan, 1993], IL is the most straightforward way to expand the RL framework to multiple agents. The idea is to equip each agent in the MARL problem with a single-agent RL algorithm, considering the actions of the other agents as part of the environment.

Like in SARL, IL algorithms can be classified in value-based methods and policy-based algorithms.

Independent Deep Q-Networks (iDQN)

iDQN has been introduced by [Tampuu et al., 2017]. They combine independent Q-learning [Tan, 1993] with DQN [Mnih et al., 2015]: each agent i maintains its own Q-network $Q^i(s, a^i; \theta^i)$, and updates the DQN loss (2.10) independently and simultaneously.

iPPO

Several PG counterparts have been proposed for IL. The one that has expressed the best performance on the MARL benchmarks is Independent Proximal Policy Optimization (iPPO) [de Witt et al., 2020]. They generalize the PPO algorithm [Schulman et al., 2017] for a multi-agent setting. Each agent i updates its policy $\pi^i_{\theta^i}$ and maintains its own value network $V^i_{\varphi^i}(s^i)$ and performs PPO updates following (2.26).

iPPO is originally designed for agents that are *homogeneous* (i.e. sharing the same state space, action space and policy parameters). This design leverages the benefits of uniform learning dynamics and the mutual relevance of shared experiences among agents. However, the flexibility of IL allows its extension to environments with *heterogeneous* agents like iDQN. This adaptation, however, introduces additional layers of complexity. In particular, it not only complexifies the optimization problem but also exacerbates the challenge of non-stationarity due to the concurrent learning processes among diverse agents.

Both iDQN and iPPO take actions and maximize their cumulative return without considering the influence of the other agents. Even if this approach has the benefits of being fully distributed and decentralized by construction, it suffers from various theoretical limitations that can result in instabilities during training and convergence to sub-optimal policies [Tan, 1993]. Indeed, the environment in this framework is non-stationary because other agents are learning at the same time. Therefore, there is no theoretical guarantees for IL.

2.3.3 Centralized Training, Decentralized Execution

Recent research has been focusing on addressing these theoretical shortcomings. One of the most common ways of achieving this is to use a centralized critic during training, taking advantage of the fact that a lot of MARL problems can

be trained in a centralized way (in a factory for example) before being deployed in the real world. The benefit is to reduce or remove the issues of non-stationarity and partial observability raised by IL, while learning decentralized policies.

Value Factorization Algorithms (VDN, QMIX)

The second framework using CTDE is value factorisation. For cooperative problems only (single collective reward), value factorisation aims to decompose the joint state-action value function Q_{tot} into individual observation-action value functions Q_i. The idea is that maximizing the total Q-value should lead to maximizing the individual Q-values. Two of the most famous algorithms are Value Decomposition Network (VDN) [Sunehag et al., 2018] that learns a linear decomposition of the joint Q-function and QMIX [Rashid et al., 2018] that learns a more complex monotonic factorisation of it.

To go in more details, VDN assumes that the joint Q-function Q_{tot} can be additively decomposed into individual value functions Q_i:

$$Q_{tot}(o_1, \dots, o_N, a_1, \dots, a_N) = \sum_{i \in \mathcal{N}} Q_i(o_i, a_i).$$

The individual Q-functions are updated by backpropagating gradients using the joint reward through the total Q-value with the DQN loss (2.10).

QMIX, learns a more complex decomposition of the total Q-value. The authors introduce a *mixing network* that combines all these individual Q-values into a joint Q-value $Q_{tot} = \texttt{Mixing\_Network}(Q_1, Q_2, \dots, Q_n)$. As for VDN, the total Q-function is used to minimize the DQN loss (2.10) and the gradient is backpropagated to the individual Q-functions.

In practice, we want to factorize the total Q-value such that maximizing the total Q-value gives the same result as maximizing the individual Q-values:

$$\arg\max_a Q_{tot}(s, a) = \left[\arg\max_{a^n} Q_n(s^n, a^n)\right]_{n=1,\dots,N} \tag{2.29}$$

To do so, the mixing network enforces a monotonic constraint between Q_{tot} and Q_n:

$$\frac{\partial Q_{tot}}{\partial Q_n} \geq 0, \quad \forall n \tag{2.30}$$

which is fulfilled by constraining the parameters of the mixing network to be positive.

Policy Gradient with Centralized Critic

The second family of CTDE methods is PG based. This framework trains decentralized policies, that make decisions solely on their own individual observation and without any coordination. Yet, this framework uses a centralized critic during training that has access to any information in the environment. The role of the centralized critic is to leverage its global perspective to provide informed feedback during training, thereby helping the decentralized agents in learning optimal or near-optimal policies that collectively enhance the overall performance. Examples in this family include MADDPG and MAPPO, introduced by [Lowe et al., 2017] and [Yu et al., 2022] respectively. These frameworks extend the DDPG and PPO algorithms by incorporating a centralized critic in the gradient calculation process.

Nonetheless, CTDE approaches face significant challenges, particularly in scalability and computational complexity, which increases which the number of agents. These challenges arise from the exponential growth of state and action spaces, increased communication overhead, and the difficulties in managing partial observability and non-stationarity in complex environments. Furthermore, these algorithms also lack theoretical guarantees, making their deployment in a real world system less reliable.

2.3.4 Complementary Multi-Agent Frameworks

In this thesis, we focus on MARL frameworks capable of training fully decentralized policies.

Nevertheless, in order to offer a comprehensive overview about the field, we explore complementary MARL frameworks in this section. While they offer insightful perspectives, they were not explored in this work, and studying their applicability and benefits in a URLLC context could be further investigated in future works. First, the *Networked Agents* framework allows agents to exchange local information with their neighbors over a communication network. Notable examples are the Networked Actor Critic [Zhang et al., 2018], FQI [Zhang et al., 2021], SAC [Qu et al., 2020] and DGN [Jiang et al., 2018] algorithm. Next, in the

Learning to Cooperate framework, agents dynamically learn the most effective message to exchange with their peers. This framework is particularly interesting in scenarios where agents are constrained to exchange limited information, such as a few bits, rather than their entire observations. A few examples in this category include the RIAL and DIAL [Foerster et al., 2016] and the CommNet architecture [Sukhbaatar et al., 2016]. Lastly, the *Mean Field RL* framework presents a unique approach by aggregating the behaviors of multiple agents, thereby simplifying the complexity inherent in managing large-scale interactions. This method is particularly advantageous in scenarios with a very large number of agents, although it relies on the assumption that agents can be effectively represented by a aggregated value known as the "mean field". MF-Q, MF-AC [Yang et al., 2018] and RL for LSMFE [Subramanian and Mahajan, 2019] are such examples in this approach.

Table 2.1: Taxonomy of MARL algorithms

Category	**Advantages**	**Disadvantages**	**Algorithms**
Independent Learning	Decentralized algorithms	No theoretical guarantees	iDQN, iAC, iPPO
Centralized training with decentralized execution	Better theoretical guarantees	Not scalable to many agents	VDN, QMIX, MADDPG, MAPPO
Networked agents	Scalable to many agents, better theoretical guarantees	Constraining hypothesis, need agents to communicate	Networked Actor Critic, FQI, SAC, DGN
Learning to cooperate	Agents learn messages	No theoretical guarantees, need communication	RIAL, DIAL, CommNet
Mean-field RL	Aggregates many agents	Aggregated by a mean field	MF-Q, MF-AC, RL for LSMFE

Chapter 3

FilteredPPO: a Deep Reinforcement Learning Scheduler for Uplink IoT Traffic.

Contents

3.1 Introduction

In this chapter, we present a centralized solution to address the uplink MA problem specific to IoT devices with strict latency requirements. In this framework, devices can access a common communication medium, but only when the BS chooses to schedule them. This orchestration ensures the absence of collisions,

a persistent issue with conventional random access protocols. However, within this paradigm, the BS has limited visibility into the devices' buffers in order to minimize the channel resources dedicated to the optimization of scheduling. As a consequence, the BS needs to efficiently learn the traffic patterns of the devices in order to maximize the throughput, given its lack of real-time knowledge about when a device wants to transmit. Our model falls in the category of centralized protocols with limited communication and the model complexity is linear with the network size. In this chapter, we consider a probabilistic traffic model and packets with strict deadlines.

3.1.1 Related Work

Centralized solutions usually require a central controller to gather relevant information from every device every time it wants to allocate the medium. This constant communication is very heavy in energy and communication resources and thus is very impractical for IoT scenarios. However, centralized protocols have the advantage of allowing transmissions without collisions and potentially achieve a higher throughput when the load is high. One way to use a centralized approach while being energy efficient is by exploiting historical data with Machine Learning (ML) [Bi et al., 2015] to learn an optimal scheduling strategy. Yet, the biggest limitation of these data-based methods is to constitute a proper dataset, let alone getting a labeled training set if we intend to apply supervised learning. Another way to tackle this substantial limitation is to consider centralized solutions with very limited communication where the BS tries to predict the traffic of the devices. A popular method to learn an optimal scheduling policy under uncertainty is Multi-Armed Bandits [Slivkins, 2019, Yu et al., 2018], but unlike RL, actions are not conditioned to states and do not have impact on the environment. In RL, the authors in [Hribar et al., 2019] model a network controller with a deep Q-network to determine the next update time at which each sensor should transmit its observations. Similarly, and closest to our work, [Zhong et al., 2018] tackles the dynamic multi-channel access problem with an actor-critic model where the agent (the access point) can only sense the chosen channel at each iteration. However, the multi-channel selection problem tackled by the authors is much easier than the multiple access problem we tackle in this chapter as we assume the traffic arrivals of each device to be independent and

consider strict deadlines.

Even if previous studies has been done to propose centralized solutions for multiple access, to the best of our knowledge this approach is the first that offers a centralized approach tackling partial observability and latency constraints with DRL.

3.1.2 Contributions

Within this chapter, we approach the uplink MA problem with time-sensitive traffic as a partially observable problem. We address it by introducing a DRL algorithm and our key contributions can be summarized as follows:

- We formulate the general uplink MA problem as a POMDP. In this framework, the BS can only observe the state of the last polled device. To incorporate the strict latency constraint in our model, we impose the devices to transmit their packets within a given deadline δ so that if a packet is not sent before, it is discarded. This framework captures the practical challenges of real-time wireless networks [Hou and Kumar, 2013].
- We present FilteredPPO, a novel DRL algorithm designed to address the challenges posed by our MA problem. Our solution models the BS as a DRL agent: it combines the PPO algorithm [Schulman et al., 2017] with a RNN to handle partial observability [Hausknecht and Stone, 2015]. In addition, we incorporate invalid action masking [Huang and Ontañón, 2020] to speed up the training process and make our algorithm more efficient with large number of devices.
- We provide numerical evidence of the superior performance of our approach in terms of throughput over the traditional MA benchmarks. We demonstrate the effectiveness of our method in scenarios where packets are generated according to an heterogeneous periodic traffic, where users are synchronous and asynchronous.

3.2 System Model

We consider a network of K devices communicating with a BS over a wireless shared channel on the uplink. The wireless channel is supposed to be time-

slotted and at every slot, the BS polls one of the devices for a potential uplink transmission. We operate under the assumption of slot synchronization, meaning that all devices are aligned with a common slot start time. This synchronization is facilitated by the BS, which sends downlink signals to coordinate the timing across devices. All packets are supposed to require the same transmission time of one time slot and propagation delay is assumed to be negligible.

3.2.1 Traffic Model

We assume the traffic pattern of every device to be periodic, i.e. every period of N_p time slots, a device k has a probability ξ_k of receiving a new packet. All packets are supposed to require the same transmission time of one time slot. Each device has an individual constraint δ_k, such that when a packet has not been transmitted before the constraint δ_k, it is dropped. We allow the devices to be heterogeneous in the following sense:

- They can have different packet arrival probabilities ξ_k.
- They can have different packet delivery constraints δ_k.
- They are not synchronous: each device is assigned an offset parameter $\bar{f}_k \in [0, N_p]$.

However, the traffic period N_p is assumed to be the same for every device and is known by the BS. At every slot $t \geq 0$, the probability for a device $k \in [1, K]$ of having a new packet is:

$$\bar{\xi}_k(t|\bar{f}_k, \xi_k, N_p) = \mathbb{1}_{\{t[N_p]=\bar{f}_k\}}\xi_k \tag{3.1}$$

An illustration of the traffic model is shown in Figure 3.1.

3.2.2 Problem Formulation

Compared to traditional MA protocols like ALOHA [Roberts, 1975], devices do not decide when to transmit their packets. In our case, the BS is a central controller that decides whether or not a device can transmit. At every time step t, the BS schedules a sensor to send its packet. If the latter has indeed a packet to send, the transmission is successful. On the other hand, if it does not have a

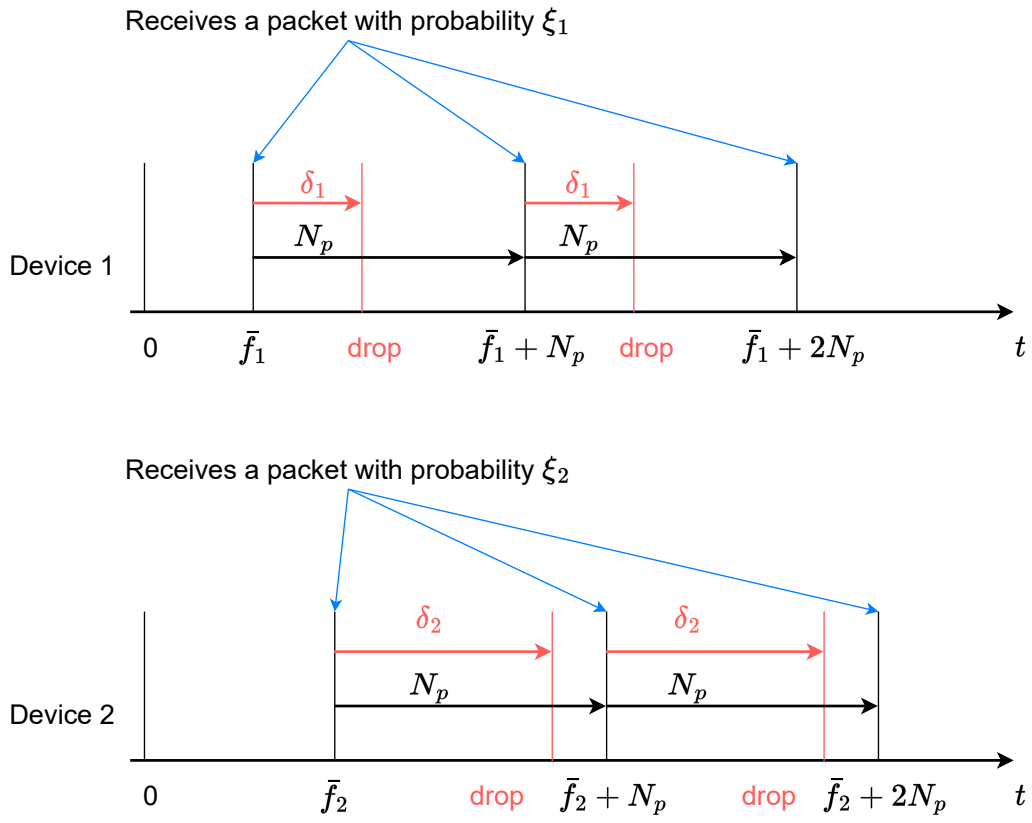


Figure 3.1: Traffic model of 2 heterogeneous devices. Every period N_p, they receive a packet with probability ξ_k and need to deliver it within δ_k slots. They are also not synchronous: they have an individual offset $\bar{f}_k$, $k \in \{1, 2\}$.

packet, it remains idle. We illustrate these interactions in Figure 3.2. The main advantage of this model is that it guarantees no collision as all decisions are made by the BS. However, in order to minimize the control overhead, we only allow the controller to have access to partial information, that is the information received by the device after polling it, which explains why the BS can poll a device that has no packet to send. Indeed, assuming a centralized agent that can gather relevant information before allocating the communication resources is not realistic to meet the strict performance requirements of the IoT. As a

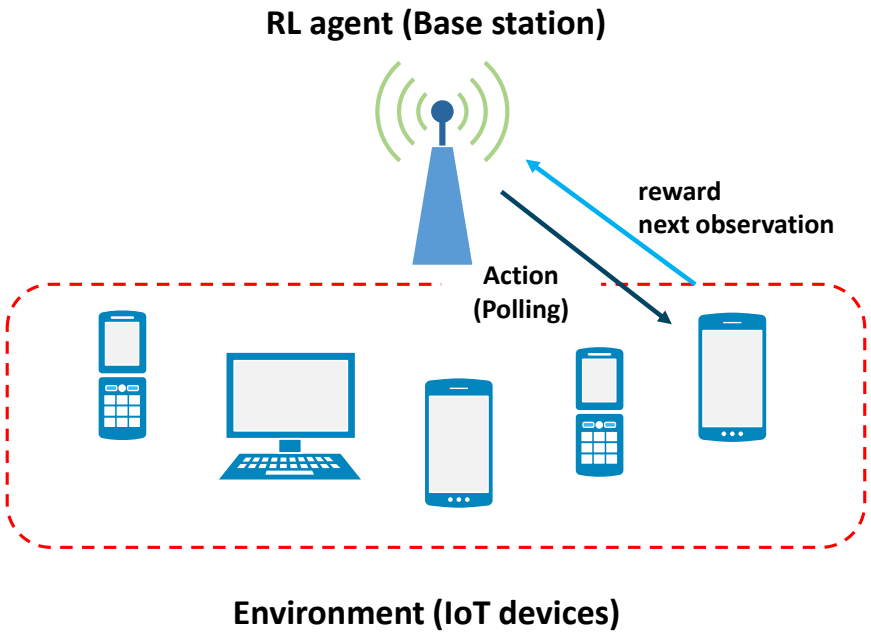


Figure 3.2: Multiple access problem modeled as a polling problem where the BS is the RL agent. The IoT devices share a common communication channel with the BS.

consequence, many packets can be lost and the throughput will be sub-optimal if the RL agent is not able to learn the traffic patterns successfully.

3.2.3 Partially Observable Markov Decision Process

We model the problem as a POMDP where an agent observes an environment and interacts with it. We define $s = (s^1, s^2, \ldots, s^K)$ the environmental state where $s^k \in \{0, 1, \ldots, \delta_k\}$ is the time a packet has spent in device k's buffer. For example, $s^k = 0$ means k's buffer is empty. The action set of the agent is $\mathcal{A} = \{1, 2, \ldots, K\}$: at each time, the BS schedules a device $a \in \mathcal{A}$ that can transmit its packet if its buffer is not empty. When a sensor i is scheduled, in addition to the main message, it also transmits the number of discarded packets η_t^k since the last time it was scheduled as an additional message. We thus define the reward of the agent r_t as a trade-off between maximizing the throughput (number of successful transmissions) and minimizing the number of discarded packets:

$$r_t(s_t, a) = \beta \mathbb{1}_{\{s_t^a > 0\}} + (1 - \beta)\frac{1}{1 + \eta_t^a} \tag{3.2}$$

where $\beta \in [0, 1]$ is a hyperparameter balancing the preferences of the agent between both quantities. Note that even if maximizing the throughput and minimizing the number of dropped packets seem equivalent, the agent has different incentives depending on the load. Indeed, if a few devices have very high arrival probabilities, the agent is almost certain to get a strictly positive reward when scheduling these devices. If the number of available slots is limited (inferior to the number of devices), only these devices will be scheduled and the learned policy would not be fair. Thus, balancing the number of successful transmissions with the number of discarded packets is a way to ensure fairness while also maximizing the throughput.

Every step t, the agent can only observe $o_t = (a_{t-1}, s_{t-1}^{a_{t-1}}) \in \mathcal{A} \times \mathbb{N}$, which is the device chosen at time $t - 1$ and its corresponding state. The agent keeps a history of the H most recent observations, $\hbar_t = (o_t, o_{t-1}, \ldots, o_{t-H+1})$. An action is selected following the policy $\pi(\cdot|\hbar_t)$, which is a probability distribution over the action space $\mathcal{A}$ given the H most recent observations $\hbar_t$.

Finally, we can write the transition from a state s_t to a state s_{t+1}. A component $k \in \{1, 2, \ldots, K\}$ of a new state, knowing the action and the previous state is:

$$s_{t+1}^k | a, s_t^k = \begin{cases} 0 & \text{if } a = k \\ 0 & \text{if } a \neq k \text{ and } s_t^k > \delta^k \\ s_t^k + 1 & \text{if } a \neq k \text{ and } 0 < s_t^k \leq \delta^k \\ X_k \sim \mathcal{B}(\xi_k) & \text{if } s_t^k = 0 \text{ and } t[N_p] = \bar{f}_k \end{cases} \tag{3.3}$$

where $\mathcal{B}(\cdot)$ is the Bernoulli distribution.

In other words, the next state of a device k becomes 0 if it has been polled in the last time slot or if the last state reached the constraint δ_k. If it has not been polled and has not reached the constraint yet, we increment the last state. Finally, if the last state was 0, we draw a new packet according to a Bernoulli distribution with parameter ξ_k if $t[N_p] = \bar{f}_k$.

3.3 FilteredPPO: a RL Approach for Access with Strict Deadlines

We propose FilteredPPO: a policy gradient algorithm that combines the PPO algorithm [Schulman et al., 2017] with invalid action masking [Huang and Ontañón, 2020] and the LSTM architecture [Hochreiter and Schmidhuber, 1997] to solve this POMDP problem.

The objective we aim at maximizing is the same as in (2.26)

However, the full state s is unknown to the RL agent as it can only see the states of the devices it has previously selected. We adapt the PPO theory to the partially observable setting by using the approach of [Hausknecht and Stone, 2015]. We use a recurrent architecture that approximates the underlying system state at time t, s_t based on past action-observation pairs we denote $\hbar_t$. For the recurrent architecture, we chose a LSTM architecture to alleviate the vanishing gradient problem that usually exists by using an RNN [Hochreiter and Schmidhuber, 1997].

We can define the policy loss based on (2.26):

$$L_\pi(\theta) = -\hat{\mathbb{E}}_B \left[\min \left(\frac{\pi_\theta(a_t|\hbar_t)}{\pi_{\text{old}}(a_t|\hbar_t)} \hat{A}_t, g(\nu)\hat{A}_t \right) \right], \tag{3.4}$$

where $\hat{\mathbb{E}}_B$ is the empirical average over a finite batch of trajectories, $g(\nu) = \text{clip}\left(\frac{\pi_\theta(a_t|\hbar)}{\pi_{\text{old}}(a_t|\hbar)}, 1-\nu, 1+\nu\right)$ and $\nu \in [0, 1)$.

To estimate the advantage function, we chose to use the simplest one using the empirical rewards [Schulman et al., 2015b]:

$$\hat{A}_t = \hat{R}_t - \hat{V}_\theta(\hbar_t) \tag{3.5}$$

with $\hat{R}_t = \sum_{t'=t}^{T} \gamma^{t'} r_{t'}$ the sum of future rewards obtained on the trajectory and $\hat{V}_\theta(\hbar_t)$ an estimate of the value function with a neural network. Note that the value and policy network share the same parameters θ.

The neural network we use is the one described in Figure 3.3. All hidden layers have 100 neurons. It takes the action-observation history as input and returns the policy vector and the value of the corresponding input. To update the head corresponding to the value estimator, we minimize the Mean Square Error (MSE) between our value estimator and the sum of future rewards:

$$L_V(\theta) = \hat{\mathbb{E}}_B(\hat{V}_\theta(\hbar_t) - \hat{R}_t)^2. \tag{3.6}$$

Therefore, the total loss we minimize is:

$$L(\theta) = L_\pi(\theta) + L_V(\theta). \tag{3.7}$$

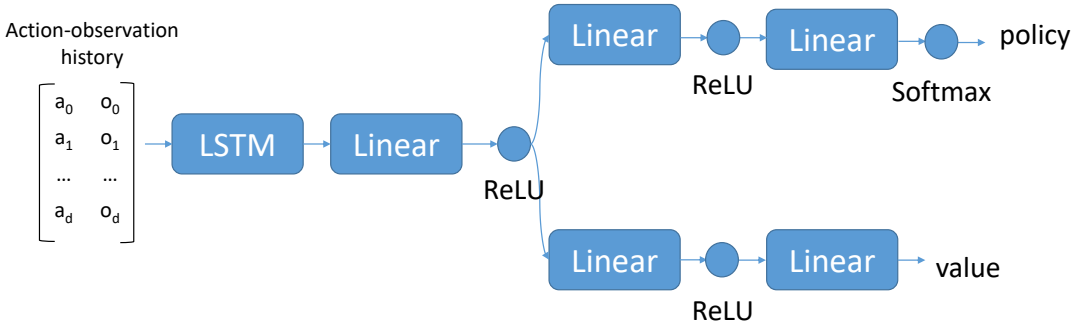


Figure 3.3: Neural network architecture

3.4 Speeding up the learning process with Action Masking

When the number of devices increases, the action space becomes larger, therefore training a policy becomes slower and sometimes requires a deeper architecture (more layers, more neurons) to learn the optimal policy. In order to address this issue, we use a solution called *invalid action masking* [Huang and Ontañón, 2020] to provide an algorithm, namely FilteredPPO, able to solve the polling problem for any number of agents with the same architecture and the same parameters.

The idea is to speed up the training process by masking actions which we know that are suboptimal for the agent because of the structure of the problem. Indeed, the traffic is periodic so one device cannot have more than one packet to transmit during the arrival period N_p. Thus, we mask the κ last actions made by the agent in the policy by setting their probability to 0. The authors of [Huang and Ontañón, 2020] theoretically and experimentally studied the action masking methodology for policy gradient algorithms and prove that it leads to valid policy gradient updates. In our experiments, we took $\kappa = N_p$.

Algorithm 3: PPO for centralized multiple-access.

Input Initial policy parameters θ_0 number of updates J
for *k = 0, 1, 2, ... J* **do**

1. Run the policy π_{θ_k} and collect a set of trajectories
2. Compute the advantage estimates $\hat{A}_t$ with (3.5)
3. Compute the policy loss $L_\pi(\theta)$ and the value loss $L_V(\theta)$ from (3.4) and (3.6)
4. Minimize the total loss $L(\theta) = L_\pi(\theta) + L_V(\theta)$ with multiple steps of stochastic gradient descent.

3.5 Experiments

We test our algorithm with different number of devices ranging between 6 and 60, with and without offsets. The environment parameters are chosen as follows:

- The arrival probabilities are chosen from $\{0.2, 0.5\}$ with probabilities $(0.5, 0.5)$ respectively.
- The deadlines are chosen from $\{5, 10, 15, 20\}$ with probabilities $(0.1, 0.1, 0.4, 0.4)$ respectively.
- The period is equal to $N_p = 20$ for all devices.

In the scenario with offsets, the offsets are chosen uniformly in $[0, N_p]$. Otherwise, they are all set to 0. In the following, the figures show the mean and standard deviation over multiple seeds.

3.5.1 Benchmarks

To evaluate the performance of our algorithm, we benchmark it against the following algorithms:

- **Random agent**: schedules a device uniformly at random.
- **Round Robin agent**: schedules devices in a cycle so that the resource is shared equally among the devices.

- **Slotted ALOHA**: all devices can access the medium with the same probability p. As it is not trivial to derive the optimal probabilities analytically, we set the transmission probabilities experimentally, such that the throughput is maximal.

- **Matching Agent**: in the scenario where all offsets are equal to 0, it is possible to derive the optimal scheduler if the statistics of the model are known (arrival probabilities, constraints, period). We call this algorithm the **Matching agent**. The idea is to model the scheduling problem as a *bipartite graph matching* problem where we need to match K devices with N_p slots. There is an edge between a device k and a slot t if the latter is inferior to the device's constraint, i.e. $t < \delta_k$. We set the weight of device k to ξ_k in order to maximize the throughput. Note that this scheduling method is completely unfair when the number of devices is greater than the number of slots N_p. To alleviate this issue, we could set the weights in order to maximize an α-fairness objective but this is beyond the scope of this thesis and we leave this for future work. In our experiments, we solve the maximum weighted matching problem using the Hungarian algorithm [Mills-Tettey et al., 2007].

3.5.2 Simulation results

The parameters of our algorithm are given in Table 3.1.

Table 3.1: Parameters of the experiments

Parameter	Value
Episode length	50
Discount factor: γ	0.8
Learning rate	0.002
Clipping hyperparameter: ε	0.1
Number of epochs	4
Preference parameter: β	0.3
History length H	K

We train our algorithm on 200000 timesteps (4000 episodes) and test it on 20000 timesteps (400 episodes). We update the neural network every 200 steps (4 episodes). The neural network weights are initialized with a random orthogonal

matrix as described in [Saxe et al., 2013].

Synchronous setting: First, in Figure 3.4, we test our algorithm on a scenario where all devices are synchronous. This allows us to benchmark it against the optimal scheduler knowing all the environment parameters (matching agent).

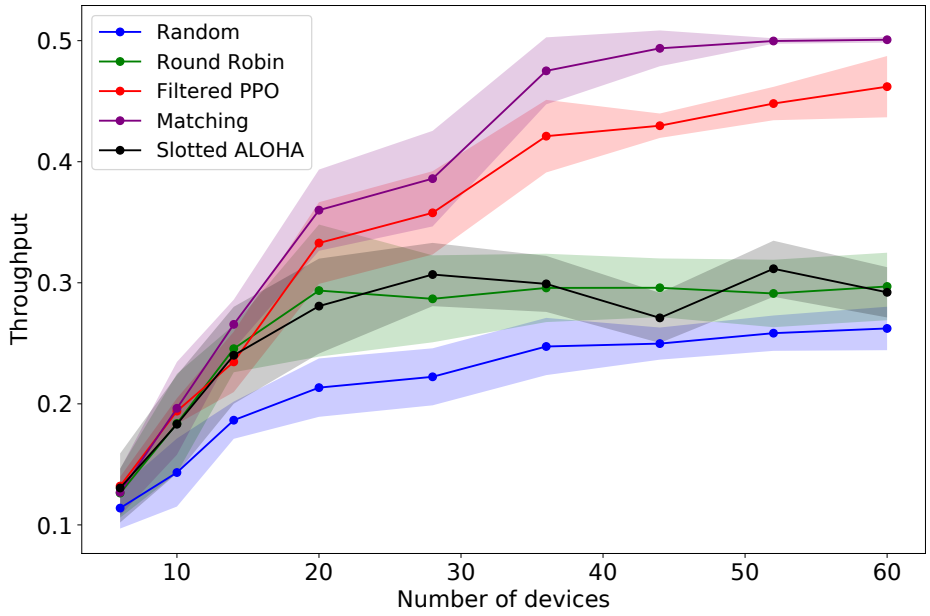


Figure 3.4: Evolution of the throughput with the number of devices. The devices are synchronous. The results are computed on 5 seeds.

Asynchronous setting: Second, we test our algorithm in the more general asynchronous setting. The results are shown in Figure 3.5, including the version of our proposed algorithm without invalid action masking, called No-filter PPO.

We can notice that in both scenarios, Filtered PPO successfully exploits the heterogeneity in the devices to outperform Round Robin, ALOHA and the Random agent for every number of transmitters in terms of throughput. In the synchronous setting, the performance of our algorithm is close to the performance of the matching algorithm. This is achieved despite the fact that our algorithm is not aware of the traffic probabilities and deadlines. Our algorithm still outperforms Round Robin, ALOHA and Random scheduling in the asynchronous setting where the matching agent cannot be applied. We also note that in the

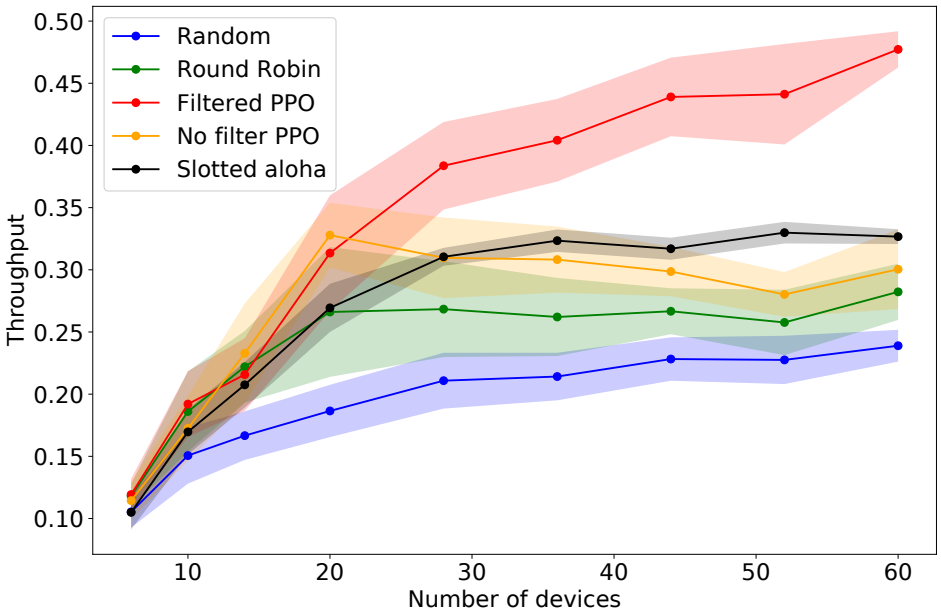


Figure 3.5: Evolution of the throughput with the number of devices. The devices are not synchronous. The results are computed on 10 seeds.

asynchronous setting, ALOHA performs better than Round Robin, unlike the synchronous one, because the probability of having a collision is smaller when devices are not synchronous.

Finally, we show the impact of invalid action masking on the throughput in the asynchronous setting. One the one hand, we can see in Figure 3.5 that when the number of devices increases, learning a good policy becomes more difficult for No-filter PPO, leading to a lower throughput than FilteredPPO. On the other hand, Figure 3.6 shows the evolution of the throughput during training for 36 asynchronous devices. We can see that FilteredPPO outperforms No-filter PPO quite fast and keeps increasing this difference until it converges to a high throughput.

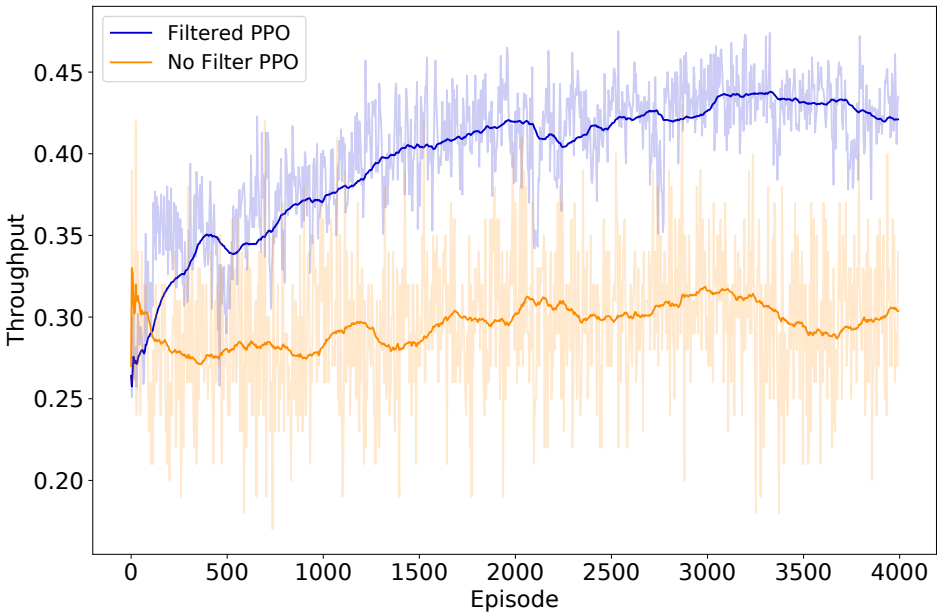


Figure 3.6: Evolution of the throughput during the training phase of PPO with and without invalid action masking. The experiment was made with 36 devices and with offsets. The dark curves show the moving average over 70 episodes.

3.6 Conclusion

In this chapter, we modelled the centralized uplink multiple access problem with strict deadlines in IoT as a polling problem with imperfect information and proposed a DRL algorithm for the BS to learn the polling strategy. We extended the PPO algorithm to the partially observable setting using a RNN and improved its performance on a large number of agents with invalid action masking. We showed that our method successfully manages to learn the traffic patterns of the transmitters despite the partial observability of our problem. Numerical results show that our solution outperforms traditional MA protocols and reaches a performance comparable to the performance of an optimal algorithm aware of the traffic characteristics and of the strict deadlines.

However, this approach is not without its limitations. A significant challenge arises from the inherent lack of real-time observability in the centralized polling system. This limitation becomes particularly evident in scenarios where devices with low arrival probabilities compete for the same slot, leading to occasional

suboptimal scheduling decisions by the polling agent. In such cases, a decentralized approach might offer a more efficient solution, reducing the likelihood of collisions due to the sparse traffic. To enhance the centralized polling method, one potential strategy is to enable the central agent to poll multiple devices simultaneously. While this could increase the throughput, it also introduces the risk of collisions and expands the action space for the scheduling agent into a combinatorial problem. Another critical aspect to consider is the dependency of our approach on discernible traffic patterns. In environments characterized by unpredictable traffic, such as those with Poisson arrivals, our model may struggle to develop effective scheduling policies. Addressing these challenges, Chapter 4 explores an innovative centralized solution where the BS is equipped to poll multiple devices in a single slot, adapting to environments with Poisson arrivals and enhancing the system's overall efficiency. On the other hand, Chapter 5 and Chapter 6 delve into decentralized solutions that could potentially circumvent the limitations of the centralized approaches.

Chapter 4

NOMA-PPO: Deep Reinforcement Learning for Uplink Scheduling in NOMA-URLLC Networks

Contents

4.1 Introduction

In this chapter, we extend the partially observable uplink scheduling framework introduced in Chapter 3 to NOMA-URLLC networks. Our previous investiga-

tions highlighted a central limitation in centralized scheduling approaches for URLLC: the limited visibility over the devices' state. This constraint significantly hinders the ability of this framework to meet the stringent reliability requirements of URLLC. To enhance reliability, we propose an innovative strategy where the BS is enabled to poll multiple users within the same time frame. It is also equipped with the NOMA technology to effectively mitigate interference from simultaneous packet transmissions.

However, this approach brings some new complexities. Indeed, enabling the polling of multiple users exponentially increases the action space of the RL agent, transforming the action selection problem into a combinatorial one at every time step. Furthermore, the RL agent not only has to deal with partial observability over user's buffer status but also faces limited visibility over the channel states. These channel states are crucial for optimizing the SIC decoding process in NOMA implementation.

To tackle these challenges, we present a novel DRL algorithm, named NOMA-PPO. This algorithm is specifically designed to tackle the uplink NOMA-URLLC scheduling challenge.

4.1.1 Related Work

Recent advances in DRL [Mnih et al., 2015] have been applied to solve several limitations in IoT systems [Chen et al., 2021] and could be potential solutions for the NOMA-URLLC problem. The approach of [Yang et al., 2020] models the massive access problem by transforming the URLLC constraint into a data rate constraint and learns a transmission strategy in order to maximize the network energy efficiency using cooperative MARL. However, the authors do not consider strict deadlines and do not address the theoretical limitations of decentralized MARL like the non-stationarity during training.

At last, several strategies leveraging DRL have been put out to deal with the URLLC constraint in NOMA systems. The authors of [Ahsan et al., 2022] propose Deep-SARSA to tackle the resource allocation problem at the BS for minimizing the error probability in uplink transmissions. Yet, the proposed solution does not take into account the packet arrival processes, assumes full observability of the system and does not impose strict deadlines. Additionally, the work of [Liu et al., 2021] optimizes a NOMA based GF protocol with DRL.

The authors use DRL to dynamically adjust the number of repetitions and radio resources in the proactive GF scheme. Yet, the approach, which is based on SA, still suffers from a high collision rate as the load increases, is not designed for handling both deterministic and sporadic traffic and fails to take advantage of channel correlations.

In this chapter, we consider a system in which the BS semi-blindly schedules the devices for their uplink transmissions, as it is done in grant-based access, however without the need for scheduling requests. Thanks to NOMA, the BS is able to poll multiple devices for a transmission in the same resource. We thus tackle a partially observable scheduling problem where the BS should strike a balance between acquiring scheduling information and avoiding excessive collisions. Our problem is characterized by two challenges, namely a combinatorial action space and a partially observable environment, that conventional DRL algorithms fail to handle.

4.1.2 DRL challenges for uplink URLLC

First, allowing the BS to poll multiple devices in a frame drastically increases the action space. For k devices, the decision maker needs to choose between 2^k actions, which is exponential in k. Few solutions have been proposed to address this problem in the literature. The most common one is proposed in [Dulac-Arnold et al., 2015]. The authors' idea is to project the large discrete action space in a continuous action space and thus solve a continuous action RL problem with the traditional Deep Deterministic Policy Gradient algorithm [Lillicrap et al., 2015]. However, this approach assumes that the discrete action space can be embedded in a continuous space, which is not straightforward for our MA problem. An alternative is the work of [Metz et al., 2017]. The authors solve a high dimensional action space RL problem with a RNN to sequentially predict the action vector, one dimension after the other. Nevertheless, not only does this algorithm assume that we know how to order the action dimensions, but the Q-value estimated for the last dimension is very noisy, especially when k is large. An extension of this paper is the Branching Dueling Q-Network (BDQ) [Tavakoli et al., 2018]. The authors solve a RL problem with a k-dimensional action space using a dueling architecture where there is a value network common for all dimensions and k advantage networks, one for every dimension. Yet, not

only is this solution ill-suited to manage partial observability, but it also cannot account for any prior knowledge the agent might have regarding the dynamics of the environment.

Second, as the BS is not aware of the whole environment and takes decisions solely based on partial observations of the environmental state, our problem can be modeled by a POMDP [Sondik, 1971]. When observations are not Markovian, traditional RL algorithms work with history dependent policies, an approach that can be rapidly computationally intractable as the number of possible histories grows exponentially with the horizon. A way to alleviate this problem is to introduce *belief states*, a probability distribution over the states, which is also a sufficient statistic for the past history and the initial state distribution. A POMDP can be then reformulated as a MDP in which the state space is the continuous belief state space. Traditional RL methods like Q-learning or policy gradient algorithms can finally be used on the resulting belief-MDP [Kaelbling et al., 1998].

Three main methods are proposed in the literature to derive or estimate a belief state: 1) the belief update formula [Kaelbling et al., 1998], 2) a RNN [Hausknecht and Stone, 2015] and 3) a generative model [Igl et al., 2018]. However, all these methods suffer from major drawbacks. While the belief update formula requires the knowledge of the environment dynamics (transition and observation function), using a RNN or a generative model introduces a new layer of complexity since there are now two phases involved: the belief estimation and the computation of the optimal policy. Additionally, since DNNs are black boxes, it is impossible to add any prior knowledge that the agent might have about the environment. Moreover, the learned beliefs are difficult to interpret and error might be propagated to the policy optimization phase.

An alternative to the belief state is the notion of *information state* [Subramanian et al., 2022] or *internal state* [Wiering and Van Otterlo, 2012, Section 12.4.2]. The idea is to derive a function of the history which is a sufficient statistic for estimating the environmental state. However, learning such a sufficient representation of the history is difficult as it is often task-specific.

4.1.3 Contributions and outline

The contributions of this chapter can be summarized as follows:

- We formulate a general MA problem with the URLLC constraint, considering packets with strict deadlines and NOMA uplink communications as a POMDP.

- We introduce the notion of *agent state* in order to theoretically address the POMDP formulation. We show that the agent state is a sufficient statistic for the past observation-action history that allows us to 1) express past actions and observations in a compact way, and 2) convert the POMDP problem to an MDP and benefit from the convergence properties of the DRL algorithms. This transformation can be extended to other wireless settings where partial observability regarding the buffer or channel evolution needs to be addressed.

- We propose a DRL algorithm, *NOMA-PPO*, that enhances the state-of-the-art algorithm PPO [Schulman et al., 2017] with two components: 1) a branching policy network architecture in order to linearly manage combinatorial action spaces. This idea is inspired by the BDQ architecture [Tavakoli et al., 2018] and extended to PG methods. 2) Bayesian policies, that incorporate prior information about the MA problem into the DRL agent [Titsias and Nikoloutsopoulos, 2018].

- We provide numerical evidence that our approach outperforms traditional MA and DRL benchmarks across 3GPP scenarios in terms of URLLC score, convergence speed, and fairness. Furthermore, we show that our algorithm is able to cope with different traffic models, a deterministic periodic and a probabilistic aperiodic traffic model in particular. Finally, our algorithm exhibits robustness against different channel configurations and demonstrates a successful exploitation of time-varying channel information.

In Section 4.2, we define the system model. Section 4.3 formulates the POMDP problem. Section 4.4 presents the NOMA-PPO approach and finally, Section 4.5 exposes the simulations and numerical results. To enhance readability, time is denoted in parentheses rather than as a subscript, differing from the convention used in the other chapters. We also provide two tables of notations, for the system model and the RL algorithms in Table 4.1 and in Table 4.2 respectively.

4.2 System Model

4.2.1 Network Model

We consider a time-slotted wireless network of K heterogeneous devices communicating with a BS over a wireless shared channel on the uplink. Every device has a single antenna and the BS is equipped with n_a antennas. The time is divided into radio frames of duration T_f and every frame is divided into five time-slots of duration T_s (see Figure 4.1). This division represents the minimum time required for the processes of polling, transmitting and acknowledging. The time synchronization among all devices is performed by the BS using downlink signals. During the first slot of every radio frame, the BS is allowed to poll a number of devices for a potential uplink transmission, described by the vector $\boldsymbol{a} = (a_1, a_2, \ldots, a_K) \in \{0,1\}^K$, where $a_k = 1$ when the device k is polled and $a_k = 0$ otherwise. It also allocates orthogonal resources for uplink pilot transmissions from the polled devices. After a guard interval, a *polled* device with at least a packet in its buffer becomes *active* and transmits during the third slot. Its transmission includes a pilot signal for channel estimation, sent using the orthogonal resource allocated by the BS. Its transmission also includes the buffer status of the device. We assume that all packets have the same size of L bits. After a guard interval, the BS acknowledges the reception of successful transmissions.

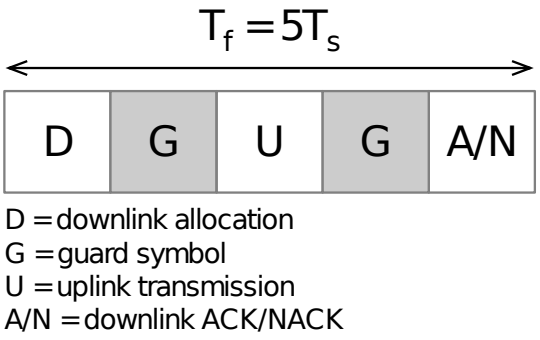


Figure 4.1: Radio Frame Structure.

The set of active users at frame $t \in \mathbb{N}$ is denoted $\mathcal{U}(t)$ and the number of active devices is denoted $U(t)$, i.e., $|\mathcal{U}(t)| = U(t)$. We denote also $\boldsymbol{u}(t) \in \{0,1\}^K$ the vector of active users at frame t such that $u_k(t) = \mathbb{1}\{k \in \mathcal{U}(t)\}$. Besides, we

define $\boldsymbol{\tau}^p(t)$, $\boldsymbol{\tau}^a(t)$, $\boldsymbol{\tau}^s(t)$ vectors of size K, where each component k represents the number of frames since the last time device k has been polled, active and successfully decoded, respectively.

We assume that the system is using NOMA [Saito et al., 2013] to improve the spectral efficiency of the network. NOMA allows several users to use the same frequency and time resources by superposing their signal in the power domain. At the receiver side, the BS applies SIC to decode the superposed signals.

Table 4.1: System Model Parameters

Notation	Description
K, n_a	Number of devices, number of antennas.
T_s, T_f	Slot length, radio frame length.
$\mathcal{U}(t)$, $\boldsymbol{u}(t)$	Set and vector of active devices at t.
$s_k(t)$, $p_k(t)$	Transmitted signal, power of k at t
$g_k(t)$, $\boldsymbol{h}_k(t)$	Large scale fading, fast fading of k at t
$\boldsymbol{r}_s(t)$	Received signal from active users.
$\boldsymbol{y}_k(t)$	Combined signal for k at t.
$\alpha(t)$	Decoding order permutation.
$\phi_k(t)$	Indicator for successful decoding of k.
$\eta_k(t)$	Received power for k at t.
$\gamma_k^{\text{no-SIC}}(t)$, $\gamma_k^{\text{SIC}}(t)$	SINR for k without SIC, with SIC at t.
$\varepsilon_k(t)$	Error probability of k at t.
n, $M^*(n,\varepsilon)$	block length, maximum code size.
B	SIC limitation.
f_k, N_p	Offset, period of packet generation for k.
d_k^h, δ_k	Head-of-line delay, latency constraint of k.
$\bar{\xi}_k(t\|f_k,\xi_k,N_p)$	Probability that k generates a new packet at t.
λ_k	Rate of packet generation at k.
$\boldsymbol{B}(t)$, $\boldsymbol{b}_k(t)$	Buffer status matrix and vector of k at t.
$\boldsymbol{d^h}(t)$	Head-of-line delays of users at t.
$\mathcal{T}^B$, $\mathcal{T}^H$	Buffer and channel state transition operators.

4.2.2 Interference Channel Model

We adopt a realistic channel model that has been adopted in the literature, based on the evaluation of the SINR [Salaün et al., 2020] and the finite block length regime, see e.g. [Ren et al., 2020].

Received Signal

In this model, a device $k \in \mathcal{U}(t)$, active in frame t, transmits a signal $s_k(t)$ of power $p_k(t) = \mathbb{E}[||s_k(t)||^2]$, where the expectation is taken over possible symbols. In a general formulation, transmit power could be controlled. However, for the sake of simplicity, this work focuses on scenarios with a fixed transmit power. The BS is supposed to receive the signal with n_a antennas and to perform Maximum Ratio Combining (MRC). The transmission of user k experiences a large scale fading $g_k(t)$, which accounts for the distance-dependent path-loss and shadowing, fast fading $\boldsymbol{h}_k(t) = [h_{k1}(t), \cdots, h_{kn_a}(t)]^T \in \mathbb{C}^{n_a \times 1}$ and thermal noise $\boldsymbol{n} \in \mathbb{C}^{n_a \times 1}$. The signal received by the BS at frame t from all active devices can thus be written as a superposition of $s_1(t), \ldots, s_{U(t)}(t)$ and thermal noise:

$$\boldsymbol{r_s}(t) = \sum_{k \in \mathcal{U}(t)} \boldsymbol{h}_k(t) \sqrt{g_k(t)} s_k(t) + \boldsymbol{n}(t) \tag{4.1}$$

where $n_i(t), i = 1, \ldots, n_a$ is an independent circularly symmetric white Gaussian process with distribution $\mathcal{CN}(0, \sigma_n^2 \boldsymbol{I})$. Thanks to the orthogonal pilots sent on the uplink, the BS is able to estimate the channel realizations of active devices. From now on, we assume that the BS has a perfect channel state information for decoding. In MRC, the signals received on the n_a antennas are combined using a weight vector $\boldsymbol{w}_k^H = \boldsymbol{h}_k$ for device k. The combined signal $\boldsymbol{y}_k(t) = \boldsymbol{w}_k^H \boldsymbol{r_s}$ for device k is thus:

$$\boldsymbol{y}_k(t) = \boldsymbol{h}_k^H(t)\boldsymbol{h}_k(t)\sqrt{g_k(t)}s_k(t) + \boldsymbol{h}_k^H(t)\boldsymbol{n}(t) + \sum_{j \in \mathcal{U}(t) \setminus \{k\}} \boldsymbol{h}_k^H(t)\boldsymbol{h}_j(t)\sqrt{g_j(t)}s_j(t) \tag{4.2}$$

We assume that BS antennas are sufficiently spaced so that the fading coefficients at every antenna are spatially uncorrelated and thus $\boldsymbol{h}_k \sim \mathcal{CN}(0, \boldsymbol{I})$ for all k. The fast fading process $h_{ki}(t)$, for $k = 1, ..., K$ and $i = 1, ..., n_a$, is supposed to follow a time-correlated Gauss-Markov model [Kobayashi and Caire, 2007]:

$$h_{ki}(t) = \bar{a}_k h_{ki}(t-1) + z_k(t) \tag{4.3}$$

where $z_k(t) \sim \mathcal{CN}(0, 1 - \bar{a}_k^2)$. The fading correlation coefficient $\bar{a}_k$ is modeled using the Jakes' model [Jakes and Cox, 1994]: $\bar{a}_k = J_0(2\pi v_k f_c T_f / c)$, where J_0 is the

Bessel function of the first kind and order 0, v_k is the speed of device k, f_c is the carrier frequency, c is the speed of light and $h_{ki}(0) \sim \mathcal{CN}(0,1)$. The coherence time for a device moving at speed v is $T_c = c/(8f_c v)$ [Tse and Viswanath, 2005]. The channels are supposed to be mutually independent across devices and constant during a frame (following a block fading channel model [Goldsmith, 2005]). We assume a rich scattering environment with stationary scatterers, as detailed in [Kobayashi and Caire, 2007, Goldsmith, 2005, Jakes and Cox, 1994]. We denote $\boldsymbol{H}(t) \in \mathbb{C}^{n_a \times K}$ the matrix of all channel realizations at time t and $\mathcal{T}^H$ the evolution process, i.e., $\boldsymbol{H}(t+1) \sim \mathcal{T}^H(\boldsymbol{H}(t))$.

Decoding Order

The SIC decoding order at each frame t can be seen as a permutation function $\alpha(t)$ over the set of active devices, i.e., $\alpha(t) : [1 : U(t)] \rightarrow \mathcal{U}(t)$. For any $i = 1, ..., U(t)$, $\alpha_i(t)$ is the i-th decoded device's index and for any $k \in \mathcal{U}(t)$, $\alpha_k^{-1}(t)$ is the rank of user k in the decoding process. When the BS tries to decode device $\alpha_i(t)$, it has already tried to decode all devices $\alpha_1(t)$,..., $\alpha_{i-1}(t)$. Each decoding might have been successful or not. Let $\phi_k(t)$ be the indicator whether an active device $k \in \mathcal{U}(t)$ has been successfully decoded by the BS ($\phi_k(t) = 1$) or not ($\phi_k(t) = 0$) and $\boldsymbol{\phi}(t) = (\phi_1(t), \ldots, \phi_K(t))$ the vector of all indicators. As a consequence, the signal received at the BS from $\alpha_i(t)$ is subject to the interference of $\alpha_j(t)$, $j > i$, i.e., from devices that have not been yet considered for decoding by the BS, and to the interference of $\alpha_j(t)$, $j < i$ whenever $\phi_{\alpha_j(t)} = 0$, i.e., from devices that have not been successfully decoded by the BS.

We now assume the decoding order that minimizes the total transmit power, given target rates on the uplink [Tse and Viswanath, 2005]: active devices are sorted in decreasing order of their received power at the BS, as follows:

$$\eta_{\alpha_1}(t) \geq \eta_{\alpha_2}(t) \geq \cdots \geq \eta_{\alpha_{U(t)}}(t) \tag{4.4}$$

where $\eta_k(t) = p_k(t) g_k(t) ||\boldsymbol{h_k}(t)||^2$. We denote $\boldsymbol{\eta}(t) = (\eta_1(t), \eta_2(t), \ldots, \eta_K(t))$ the vector of received powers. We denote $\boldsymbol{\eta}^o(t)$ the vector of powers received by the BS from the active devices, observed at time t thanks to the transmitted pilots, i.e., $\boldsymbol{\eta^o}(t) = \text{diag}(\boldsymbol{u}(t))\boldsymbol{\eta}(t)$.

Signal to Interference plus Noise Ratio

In absence of SIC, the signal (4.2) results in a SINR at the output of the combiner [Tokgoz and Rao, 2006]:

$$\gamma_k^{\text{no-SIC}}(t) = \frac{\eta_k(t)}{\sum_{j\neq k}\eta_{jk}(t)+\sigma_n^2} \tag{4.5}$$

where $\eta_{jk}(t) = p_j(t)g_j(t)\frac{|\boldsymbol{h}_k^H\boldsymbol{h}_j|^2}{||\boldsymbol{h}_k||^2}$.

With SIC however, we decode in the decreasing order of $\eta_k(t)$, so that part of the interference is potentially successively removed. The SINR with SIC writes now:

$$\gamma_k^{\text{SIC}}(t) = \frac{\eta_k(t)}{\underbrace{\sum_{j\in J_1}(1-\phi_j(t))\eta_{jk}(t)}_{\text{before } k \text{ in decoding order}} + \underbrace{\sum_{j\in J_2}\eta_{jk}(t)}_{\text{after } k} + \sigma_n^2} \tag{4.6}$$

where $J_1 = \{j \in \mathcal{U}(t), \alpha_j^{-1}(t) < \alpha_k^{-1}(t)\}$ is the set of devices that are considered for decoding before k and $J_2 = \{j \in \mathcal{U}(t), \alpha_j^{-1}(t) > \alpha_k^{-1}(t)\}$ is the set of devices that are decoded after k. Note that ϕ_j is determined iteratively: we are able to compute the SINR of device k, only once we know the outcome of the decoding for devices $j \in J_1$.

Achievable Rate with Finite Block Length

As the URLLC messages are often supposed to be very small [3GPP, 2018b] (in the factory automation scenario for instance), we adopt a finite block-length regime [Polyanskiy et al., 2010] for the calculation of the achievable rate. In this model, an encoder maps every L-bit message $m \in [1 : \hat{M}]$ to a codeword $c_m \in \mathbb{C}^n$, where $\hat{M} = 2^L$ is the size of the message space and n is the block length, also known as the number of complex channel uses. Codewords are subject to an average power constraint, i.e., $\frac{1}{\hat{M}}\sum_m ||c_m||^2 = n\eta$, where η is the received power per channel use. The codeword is transmitted over an Average White Gaussian Noise channel with noise variance σ^2. At the receiver, a decoder maps the channel output to an estimate $\tilde{m}$ of the message. The average error probability is defined as $\varepsilon = \mathbb{P}[\tilde{m} \neq m]$. A codebook $\{c_m \in \mathbb{C}^n, m \in [1 : \hat{M}]\}$ and a decoder whose average error probability is less than ε are called a $(\hat{M}, n, \varepsilon)$-code. For given

ε and n, the maximum code size is denoted $M^*(n,\varepsilon)$. Authors of [Polyanskiy et al., 2010] provide a normal approximation of the maximum achievable code rate:

$$\frac{\log_2 M^*(n,\varepsilon)}{n} \approx C(\gamma) - \sqrt{\frac{V(\gamma)}{n}} Q^{-1}(\varepsilon) \tag{4.7}$$

where $\gamma = \frac{P}{\sigma^2}$ is the Signal-to-Noise Ratio (SNR), P is the received power, $C(\gamma) = \log_2(1+\gamma)$ is the Shannon capacity[1], $V(\gamma) = \frac{\gamma}{2}\frac{\gamma+2}{(\gamma+1)^2}\log_2^2 e$ is the channel dispersion and $Q(x) = 1/\sqrt{2\pi}\int_x^\infty \exp(-t^2/2)dt$. Although (4.7) is an asymptotic approximation when n tends to infinity, it is tight for n as small as 200 [Polyanskiy et al., 2010]. When considering a block fading channel with channel realization h, (4.7) is valid *conditionnally* to the channel realization with $\gamma = \frac{P|h|^2}{\sigma^2}$. In our study, we further treat interference as noise, as it is usually done in the literature [Goldsmith, 2005, Chapter 15], and apply (4.7) with the SINR in (4.6). As a consequence, a packet of device k, transmitted at frame t, is not successfully decoded with probability:

$$\varepsilon_k(t) = Q\left(\sqrt{\frac{n}{V(\gamma_k^{\text{SIC}}(t))}}\left(C(\gamma_k^{\text{SIC}}(t)) - \frac{L}{n}\right)\right) \tag{4.8}$$

The downlink allocation and the acknowledgment are supposed to be error free.

SIC Limitation

We define an upper limit B on the number of possible multiplexed users which is characteristic of the SIC performance [Tse and Viswanath, 2005]. More specifically, a necessary condition for a device k to be decoded is that the number of active devices is less than B, i.e.

$$|\mathcal{U}(t)| \leq B \tag{4.9}$$

Finally, at frame t and for a user k, we can write $\phi_k(t)$ as a Bernoulli random variable of parameter $1-\varepsilon_k(t)$, i.e., $\phi_k(t) \sim \mathcal{B}(1-\varepsilon_k(t))$ when (4.9) is satisfied, and $\phi_k(t) = 0$ otherwise. The SIC decoding procedure is summarized in

[1]Note that contrary to [Polyanskiy et al., 2010], which considers the capacity per real dimension, we use a complex representation of the signal. As a consequence, C is the capacity per complex dimension [Tse and Viswanath, 2005, Chapter 5].

Algorithm. 4

Algorithm 4: SIC Decoding Procedure

input : $\mathcal{U}(t)$, $\boldsymbol{\eta}(t)$
output: $\boldsymbol{\phi}(t) = (\phi_1(t), \phi_2(t), \ldots, \phi_K(t))$
Initialize: $\phi_k(t) = 0, \forall k$.
if $|\mathcal{U}(t)| \leq M$ **then**
  foreach $k \in \mathcal{U}(t)$ *in decreasing order of* $\boldsymbol{\eta}(t)$ **do**
    Compute the SINR $\gamma_k(t)$ using (6) and $\boldsymbol{\phi}(t)$
    Compute $\varepsilon_k(t)$ using $\gamma_k(t)$ according to (8)
    Draw $\phi_k(t)$ from the Bernoulli distribution: $\phi_k(t) \sim \mathcal{B}(1 - \varepsilon_k(t))$, i.e. decode the packet with probability $1 - \varepsilon_k(t)$

4.2.3 Traffic Models

Packets are generated at the devices according to models of either probabilistic periodic traffic or probabilistic aperiodic traffic and are subject to a strict deadline constraint.

Probabilistic periodic traffic

In this model, directly inspired by [Hou and Kumar, 2013], a device k generates packets periodically every N_p radio frames with probability ξ_k. Devices are not synchronous, i.e., each device is assigned an offset parameter $\bar{f}_k \in [0, N_p]$ such that, at every radio frame $t \geq 0$, the probability for a device k of generating a new packet is: $\bar{\xi}_k(t|\bar{f}_k, \xi_k, N_p) = \mathbb{1}_{\{t[N_p]=\bar{f}_k\}}\xi_k$. Note that a specific case for this model is the *deterministic periodic traffic* as defined in [3GPP, 2018b, Annex A] for various use cases including for example factory automation, where $\xi_k = 1$ and $\bar{f}_k = 0$ for all k. Periodic transmissions may correspond to the periodic update of a position or the repeated monitoring of a characteristic parameter [3GPP, 2017h].

Probabilistic aperiodic traffic

This traffic model is defined in [3GPP, 2018b] and is based on the File Transfer Protocol (FTP) model 3 defined in [3GPP, 2015], however with a fixed packet length. At every device k, packets are generated according to a Poisson process

of rate λ_k. An aperiodic transmission may correspond to process, diagnostic or maintenance events that trigger the transmission [3GPP, 2017h].

Deadlines

Every device k has an individual latency constraint $\delta_k \in \mathbb{N}^*$ expressed in number of radio frames, such that a packet that has not been transmitted after δ_k radio frames is dropped. Let $\delta = \max_k \delta_k$. When a transmission fails, a device is allowed to retransmit the packet as long as it has not expired.

Buffers

We assume that devices have an infinite buffer and packets in the queue are delivered in a "first come, first served" manner. For every device k, the buffer at time t can be represented by a vector $\boldsymbol{b}_k(t) = [b_{k,1}(t), ..., b_{k,\delta}(t)] \in \mathbb{N}^\delta$ where $b_{k,d}(t) = i$ when device k has i packets with time-to-deadline d at time t. We denote $\boldsymbol{B}(t)$ the matrix of all buffer status at time t. The *head-of-line delay* $d_k^h(t)$ of user k at frame t is defined as $b_{k,d_k^h(t)}(t) \neq 0$ and $b_{k,d}(t) = 0$ for all $d < d_k^h(t)$. This is the smallest time-to-deadline in the buffer of device k. We note $\boldsymbol{d^h} = [d_1^h, d_2^h, \ldots, d_K^h]$ the vector of all head-of-line delays. When a device is polled, it chooses for transmission one of the packets associated to its head-of-line delay at random.

For a device k, the buffer status transits as follows: (a) Successfully decoded packets are removed from the buffer, i.e. $b_{k,d_k^h(t)-1}(t+1) = b_{k,d_k^h(t)}(t) - 1$ if $\phi_k(t) = 1$; (b) Other packets see their time-to-deadline decreased by one, i.e., $b_{k,d-1}(t+1) = b_{k,d}(t)$ for all $d > 1$. If $d = 1$, the packets expire and are removed from the buffers; (c) If m new packets are generated at the device, they enter the buffer with a deadline δ_k, i.e., $b_{k,\delta_k}(t+1) = m$.

We denote this operation $\mathcal{T}^B$, i.e.,

$$\boldsymbol{B}(t+1) \sim \mathcal{T}^B(\boldsymbol{B}(t), \boldsymbol{\phi}(t)) \tag{4.10}$$

When an active device is successfully decoded, its buffer status is known (or observed) to the BS. We thus denote $\boldsymbol{B}^o(t)$ the matrix of observed buffer status at time t.

4.3 Problem Formulation

4.3.1 Optimization Problem

The objective of the BS is to maximize the expected number of successful transmissions with respect to the policy π that maps the current observation history at frame t: $(\boldsymbol{B}^o(t), \boldsymbol{\eta}^o(t), \dots, \boldsymbol{B}^o(0), \boldsymbol{\eta}^o(0))$ to the vector of devices to schedule $\boldsymbol{a}(t)$. Moreover, buffers and channels are subject to the dynamics $\mathcal{T}^B$ and $\mathcal{T}^H$, respectively. The optimization problem (P) can thus be formulated as:

$$\begin{aligned} \max_{\pi} \quad & \mathop{\mathbb{E}}_{(\mathcal{T}^B, \mathcal{T}^H, \pi)} \left[\sum_{t=0}^{\infty} \sum_{k \in \mathcal{U}(t)} \gamma^t \phi_k(t) \right] \\ \text{s.t.} \quad & \boldsymbol{B}(t+1) \sim \mathcal{T}^B(\boldsymbol{B}(t), \boldsymbol{\phi}(t)) \\ & \boldsymbol{H}(t+1) \sim \mathcal{T}^H(\boldsymbol{H}(t)) \end{aligned} \tag{P}$$

where $\gamma \in [0, 1)$ is the discount factor that determines the importance of future rewards compared to immediate ones.

4.3.2 POMDP Formulation

To solve our problem, we adopt the POMDP framework, and we define its components as follows:

State space

At each step t, a state $\boldsymbol{s}(t)$ is defined as the concatenation of the buffer status $\boldsymbol{B}(t)$, the received powers $\boldsymbol{\eta}(t)$, and the observation $\boldsymbol{o}(t)$ obtained from the active users at the previous step:

$$\boldsymbol{s}(t) = \langle \boldsymbol{B}(t), \boldsymbol{\eta}(t), \boldsymbol{o}(t) \rangle \tag{4.11}$$

where $\boldsymbol{o}(t) = \langle \boldsymbol{u}(t-1), \boldsymbol{\phi}(t-1), \boldsymbol{B}^o(t-1), \boldsymbol{\eta}^o(t-1), r(t-1) \rangle$ is the vector of active users, the vector of decoded packets, the observed buffers, the observed received power and the reward at $t-1$ respectively.

Table 4.2: Algorithms Parameters

Notation	Description
$\boldsymbol{s}(t)$, $\boldsymbol{o}(t)$	State and observation at t.
$\boldsymbol{B}^o(t)$	Observed buffers at t.
$\boldsymbol{\eta}^o(t)$	Observed received power at t.
$\boldsymbol{\tau}^p(t)$, $\boldsymbol{\tau}^a(t)$, $\boldsymbol{\tau}^s(t)$	Last time the devices have been polled, active, successfully decoded.
$\boldsymbol{a}(t)$	Action of the agent at t.
$r(t-1)$	Reward at step $t-1$.
f^A	Transition function for the agent state.
$\boldsymbol{B}^A(t)$	Buffers representations by the agent.
$\boldsymbol{\eta}^A(t)$	Last known received power by the agent.
$\hbar(t)$	History of observations and actions at t
π_θ	Policy parameterized by θ.
$A^{\pi_{\theta_{\text{old}}}}$,$V^{\pi_{\theta_{\text{old}}}}$	Advantage and value functions.
V_φ	Value network parameterized by φ
$\hat{A}^{GAE}(t)$	Generalized Advantage Estimator.
ν	Clipping parameter for PPO.
γ	Discount factor.
EDF	EDF prior.
f_{ch}	Channel prior.
f	Bayesian prior over the agent state.
q	Posterior policy

Action space

The agent has the possibility to poll any subset of devices at every frame. The action space is thus defined as $\mathcal{A} = \{0,1\}^K$. For $\boldsymbol{a} = (a_1, a_2, \ldots, a_K) \in \mathcal{A}$, $a_k = 1$ if the agent polls device k and $a_k = 0$ otherwise. Note that the action space grows exponentially with the number of devices.

Transition function

When the system is in state $\boldsymbol{s}(t)$ at the beginning of a radio frame t, it transits to state $\boldsymbol{s}(t+1)$ at the end of the radio frame. The received power $\boldsymbol{\eta}(t)$ evolves with the channel realizations $\boldsymbol{H}(t)$ and is governed by $\mathcal{T}^H$. Finally the evolution of the buffers is described by $\mathcal{T}^B$. The next observation $\boldsymbol{o}(t+1)$ is computed using the observation function defined in the next subsection.

Observation space and observation function

At every frame t, the RL agent can only observe the last feedback from the active users: the set of active users, their channel realizations, the buffer status of successfully decoded devices and the reward. From the state $\boldsymbol{s}(t+1)$ and action $\boldsymbol{a}(t)$, the observation at time $t+1$ is deterministic and defined by:

$$\boldsymbol{o}(t+1) = \mathcal{O}(\boldsymbol{s}(t+1), \boldsymbol{a}(t)) \tag{4.12}$$

$$= \langle \boldsymbol{u}(t), \boldsymbol{\phi}(t), \boldsymbol{B}^o(t), \boldsymbol{\eta}^o(t), r(t) \rangle \tag{4.13}$$

In particular, $\boldsymbol{\phi}(t) \sim \mathcal{B}(1-\boldsymbol{\varepsilon}(t))$, $\boldsymbol{B^o}(t) = \text{diag}(\boldsymbol{u}(t) \odot \boldsymbol{\phi}(t))\boldsymbol{B}(t)$ and $\boldsymbol{\eta^o}(t) = \text{diag}(\boldsymbol{u}(t))\boldsymbol{\eta}(t)$. Note that when a device k is active but its packet is not decoded, the agent still has the information that this device has a packet to transmit through $u_k(t) = 1$.

Reward function

We define the reward function as the number of successfully decoded packets:

$$\mathcal{R}(\boldsymbol{s}(t), \boldsymbol{a}(t)) = \sum_{k \in \mathcal{U}(t)} \phi_k(t) \tag{4.14}$$

Note that unlike most RL approaches for multiple access [Xu et al., 2020, Chang et al., 2018, Guo et al., 2022], we do not penalize the agent when there is a collision or interference. The reason is that we want the agent to learn a tradeoff between sensing and transmitting. In our experiments, we have noticed that using a penalty for collisions did not improve the performance.

The POMDP formulated above aims at maximizing the optimization problem (P). In general, POMDP problems are known to be PSPACE-complete [Papadimitriou and Tsitsiklis, 1987], which means that they can be solved using a polynomial amount of memory space and are at least as hard as every other PSPACE problem. In order to solve this POMDP, we introduce a sufficient statistic for the history of past actions and observations, that we call the *agent state*, and that allows us to transform the POMDP problem into an MDP.

Agent State for Solving a POMDP

Definition 4.1 (Agent state)**.** *At the beginning of each frame $t \geq 1$, we define the* agent state $\boldsymbol{A}(t)$ *after the agent receives its observation $\boldsymbol{o}(t)$ as:*

$$\boldsymbol{A}(t) = \langle \boldsymbol{B}^A(t), \boldsymbol{\eta}^A(t), \boldsymbol{\tau}^p(t), \boldsymbol{\tau}^a(t), \boldsymbol{\tau}^s(t), r(t-1) \rangle, \tag{4.15}$$

where $\boldsymbol{\tau}^p(t)$, $\boldsymbol{\tau}^a(t)$ and $\boldsymbol{\tau}^s(t)$ are the number of frames from t since the last time the devices have been polled, active and have successfully transmitted respectively. $\boldsymbol{\eta}^A(t)$ is the last known received power of the active devices, i.e., its k-th column is $\boldsymbol{\eta}_k(t - \tau_k^a(t) - 1)$. The matrix $\boldsymbol{B}^A(t)$ is a representation of the buffers given the observations made by the BS at time t and is defined as follows. If an active user k has been successfully decoded in the previous frame, we update $\boldsymbol{b}_k^A(t)$ with the new observation, i.e., $\boldsymbol{b}_k^A(t) = \boldsymbol{b}_k^o(t-1)$. For all devices, we decrease the deadlines of the packets in the buffers representation at time $t-1$ by 1 and we remove the expired packets.

While $\boldsymbol{B}^o(t)$ is an immediate observation of the buffers of the active users at t, $\boldsymbol{B}^A(t)$ is a compact representation of all past buffer observations at t. Introducing $\boldsymbol{B}^A(t)$ allows us to incorporate the knowledge of the dynamics of the buffers in the agent. Yet, the agent is still not aware of the new arrivals. To summarise, the agent state at frame t, $\boldsymbol{A}(t)$, can be written as a function f^A of the observation $\boldsymbol{o}(t)$, the previous action $\boldsymbol{a}(t-1)$ and the previous agent state $\boldsymbol{A}(t-1)$, i.e.,

$$\boldsymbol{A}(t) = f^A(\boldsymbol{A}(t-1), \boldsymbol{o}(t), \boldsymbol{a}(t-1)) \tag{4.16}$$

Proposition 4.2. *$\boldsymbol{A}$ is a sufficient statistic for the action-observation history, i.e.,*

$$P(s(t)|\hbar(t)) = P(s(t)|A(t)) \tag{4.17}$$

Proof. Let $\boldsymbol{s(t)} = \langle \boldsymbol{B}(t), \boldsymbol{\eta}(t), \boldsymbol{o}(t) \rangle$. We need to prove that $\boldsymbol{A(t)}$ is a sufficient statistic for the history $\hbar(t)$ in order to predict $\boldsymbol{s(t)}$ i.e.:

$$P(\boldsymbol{s(t)}|\hbar(t)) = P(\boldsymbol{s(t)}|\boldsymbol{A(t)}) \tag{4.18}$$

According to the Bayes rule, we have:

$$\begin{aligned} P(\boldsymbol{s}(t)|\hbar(t)) &= P(\boldsymbol{B}(t), \boldsymbol{\eta}(t), \boldsymbol{o}(t)|\hbar(t)) \\ &= P(\boldsymbol{B}(t)|\boldsymbol{\eta}(t), \boldsymbol{o}(t), \hbar_t) \\ &\quad \times P(\boldsymbol{\eta}(t)|\boldsymbol{o}(t), \hbar(t)) \times P(\boldsymbol{o}(t)|\hbar(t)) \end{aligned}$$

Besides, as $\hbar(t) = \{a(0), o(1), a(1), ..., a(t-1), o(t)\}$ and $\boldsymbol{B}(t)$ is independent of $\boldsymbol{\eta}(t)$, we have:

$$P(\boldsymbol{s}(t)|\hbar(t)) = P(\boldsymbol{B}(t)|\hbar(t))P(\boldsymbol{\eta}(t)|\hbar(t))$$

Each channel is independent so let's compute $P(\eta_k(t)|\hbar(t))$. For a device k, $\eta_k(t)$ is conditionally independent of $\boldsymbol{a}_k(t), \boldsymbol{b}_k^o(t)$ and $r(t-1)$ given $\eta_k^o(t)$. Thus, we can write:

$$\begin{aligned} P(\eta_k(t)|\hbar(t)) &= P(\eta_k(t)|\eta_k^o(t-1), \dots, \eta_k^o(0)) \\ &\overset{(a)}{=} P(\eta_k(t)|u_k(t-1)\eta_k(t-1), \dots, u_k(0)\eta_k(0)) \\ &\overset{(b)}{=} P(\eta_k(t)|\eta_k(\tau_k^a(t)), \tau_k^a(t)) \\ &\overset{(c)}{=} P(\eta_k(t)|\boldsymbol{A}(t)) \end{aligned}$$

where (a) comes from the definition of η_k^o (see Section 4.2.2-2); (b) comes from the fact that the channel realizations are Markovian and that $\eta_k(\tau_k^a(t))$ is the last observed channel realization for the device k; (c) results from the conditional independence of $\eta_k(t)$ from $\boldsymbol{B}^A(t), \boldsymbol{\tau}^a(t), \boldsymbol{\tau}^s(t), r(t-1)$ given $\eta_k^A(t)$ and $\tau_k^a(t)$.

Finally, as each user's buffer is independent, we are going to prove by induction that $P(\boldsymbol{b}_k(t)|\hbar(t)) = P(\boldsymbol{b}_k(t)|\boldsymbol{A}(t)), \forall k, \forall t \geq 0$. First, $P(\boldsymbol{s}(0)|\hbar(0)) = P(s(0)|\ o(0)) = P(s(0)|\boldsymbol{A}(0))$. Let $t \geq 0$ and $k \in [1, K]$. Let's assume that $P(\boldsymbol{b}_k(t)|\hbar(t)) = P(\boldsymbol{b}_k(t)|\boldsymbol{A}(t))$. By definition of the agent state (see Definition 4.1) and as $\boldsymbol{b}_k(t+1)$ only depends on $\boldsymbol{b}_k^A(t+1)$, we have:

$$P(\boldsymbol{b}_k(t+1)|\boldsymbol{A}(t+1)) = P(\boldsymbol{b}_k(t+1)|\boldsymbol{b}_k^A(t+1)) \tag{4.19}$$

$$\boldsymbol{b}_k^A(t+1) = \begin{cases} \boldsymbol{b}_k^o(t+1) & \text{if } \phi_k(t) = 1 \\ \boldsymbol{b}_k^A(t) & \text{if } \phi_k(t) = 0 \end{cases} \tag{4.20}$$

Therefore,

$$P(\boldsymbol{b}_k(t+1)|\boldsymbol{b}_k^A(t+1)) = \begin{cases} P(\boldsymbol{b}_k(t+1)|\boldsymbol{b}_k^o(t+1)), & \text{if } \phi_k(t)=1, \\ P(\boldsymbol{b}_k(t+1)|\boldsymbol{b}_k^A(t)), & \text{if } \phi_k(t)=0, \end{cases} \tag{4.21}$$

Besides, as $\boldsymbol{b}_k(t+1)$ is conditionally independent of $r(t)$, $\boldsymbol{H}^o(t+1), \boldsymbol{u}(t)$ given $\boldsymbol{\phi}(t), \boldsymbol{b}_k^o(t+1)$ we can write $P(\boldsymbol{b}_k(t+1)|\boldsymbol{b}_k^o(t+1)) = P(\boldsymbol{b}_k(t+1)|\boldsymbol{o}(t+1))$. Therefore:

$$\begin{aligned} P(\boldsymbol{b}_k(t+1)|\boldsymbol{b}_k^A(t+1)) &\overset{(c)}{=} P(\boldsymbol{b}_k(t+1)|\boldsymbol{b}_k^A(t), \boldsymbol{o}(t+1)) \\ &\overset{(d)}{=} P(\boldsymbol{b}_k(t+1)|\boldsymbol{A}(t), \boldsymbol{a}(t), \boldsymbol{o}(t+1)) \\ &\overset{(e)}{=} P(\boldsymbol{b}_k(t+1)|\hbar(t+1)) \end{aligned}$$

where (c) comes from merging the equations in (4.21) and noticing that $\phi_k(t) \in \boldsymbol{o}(t+1)$; (d) comes from the fact that $\boldsymbol{b}_k(t+1)$ only depends on $\boldsymbol{A}(t), \boldsymbol{a}(t), \boldsymbol{o}(t+1)$ through $\boldsymbol{b}_k^A(t)$ and $\boldsymbol{o}(t+1)$. Finally, (e) comes from the induction hypothesis. □

Proposition 4.3. *The tuple* $(\mathcal{S}^A, \mathcal{A}, \mathcal{T}^A, \mathcal{R}^A)$ *forms an MDP where* $\mathcal{T}^A : \mathcal{S}^A \times \mathcal{A} \mapsto \Delta(\mathcal{S}^A)$ *is the agent state transition function and* $\mathcal{R}^A : \mathcal{S}^A \times \mathcal{A} \mapsto \mathbb{R}$ *the agent state reward function such that:*

$$\mathcal{T}^A(\boldsymbol{A}(t), \boldsymbol{a}(t)) = \sum_{\boldsymbol{o}(t+1)\in\Omega} f^A(\boldsymbol{A}(t), \boldsymbol{a}(t), \boldsymbol{o}(t+1)) P(\boldsymbol{o}(t+1)|\boldsymbol{a}(t), \boldsymbol{A}(t))$$

$$\mathcal{R}^A(\boldsymbol{A}(t), \boldsymbol{a}(t)) = \sum_{\boldsymbol{s}(t)\in\mathcal{S}} P(\boldsymbol{s}(t)|\boldsymbol{A}(t)) \mathcal{R}(\boldsymbol{s}(t), \boldsymbol{a}(t))$$

where:

$$P(\boldsymbol{o}(t+1)|\boldsymbol{a}(t), \boldsymbol{A}(t)) = \sum_{\boldsymbol{s}\in\mathcal{S}} \mathcal{O}(\boldsymbol{o}(t+1)|\boldsymbol{s}, \boldsymbol{a}(t)) \sum_{\boldsymbol{s}\in\mathcal{S}} \mathcal{T}(\boldsymbol{s}(t+1)|\boldsymbol{s}, \boldsymbol{a}(t))$$

Proof. The expressions of $\mathcal{T}^A$ and $\mathcal{R}^A$ are directly derived using the law of total probability and the Bayes formula. □

The problem formulation is summarized in Figure 4.2.

1. At the beginning of frame t, the system is at state $\boldsymbol{s}(t)$.
2. The agent can observe $\boldsymbol{o}(t) = \mathcal{O}(\boldsymbol{o}(t)|\boldsymbol{s}(t), \boldsymbol{a}(t-1))$.
3. It then computes the agent state using (4.16).
4. It makes an action $\boldsymbol{a}(t) \sim \pi(\boldsymbol{a}(t)|\boldsymbol{A}(t))$.
5. The system then transitions to the next state as follows: $\boldsymbol{s}(t+1) \sim \mathcal{T}(\boldsymbol{s}(t+1)|\boldsymbol{s}(t), \boldsymbol{a}(t))$ and the agent receives: $R(\boldsymbol{s}(t), \boldsymbol{a}(t)), \boldsymbol{u}(t), \boldsymbol{\phi}(t), \boldsymbol{\eta}^o(t), \boldsymbol{B}^o(t)$.

Transforming the POMDP problem in an MDP allows us to leverage DRL algorithms in order to solve the optimization problem (P).

4.4 Deep Reinforcement Learning Approach

4.4.1 Proximal Policy Optimization algorithm

Our approach is based on the PPO algorithm, as detailed in 2.2.3. We employ the highly efficient Generalized Advantage Estimation (GAE) method for advantage estimation, as described in [Schulman et al., 2015b]. This algorithm uses the temporal difference residuals $\delta^V(t) = r(t) - \gamma V(\boldsymbol{s}(t+1)) - V(\boldsymbol{s}(t))$ in order to define the *Generalized Advantage Estimator* $\hat{A}^{GAE}(t)$:

$$\hat{A}^{GAE}(t) = \sum_{l=0}^{\infty} (\gamma \lambda_{GAE})^l \delta^V(t+l) \tag{4.22}$$

where $\lambda_{GAE} \in [0, 1]$ adjusts the bias-variance tradeoff. This method manages to reduce the variance of the gradient estimate and stabilizes training at the cost of introducing a bias. In practice, the value function V is approximated by a DNN with parameters φ: V_φ.

4.4.2 Exploiting Prior Knowledge

In our scheduling problem, the Earliest Deadline First (EDF) scheduler, which schedules pending packets in the increasing order of their deadline, intuitively is a good heuristic when the environment is fully observable by the scheduler. EDF

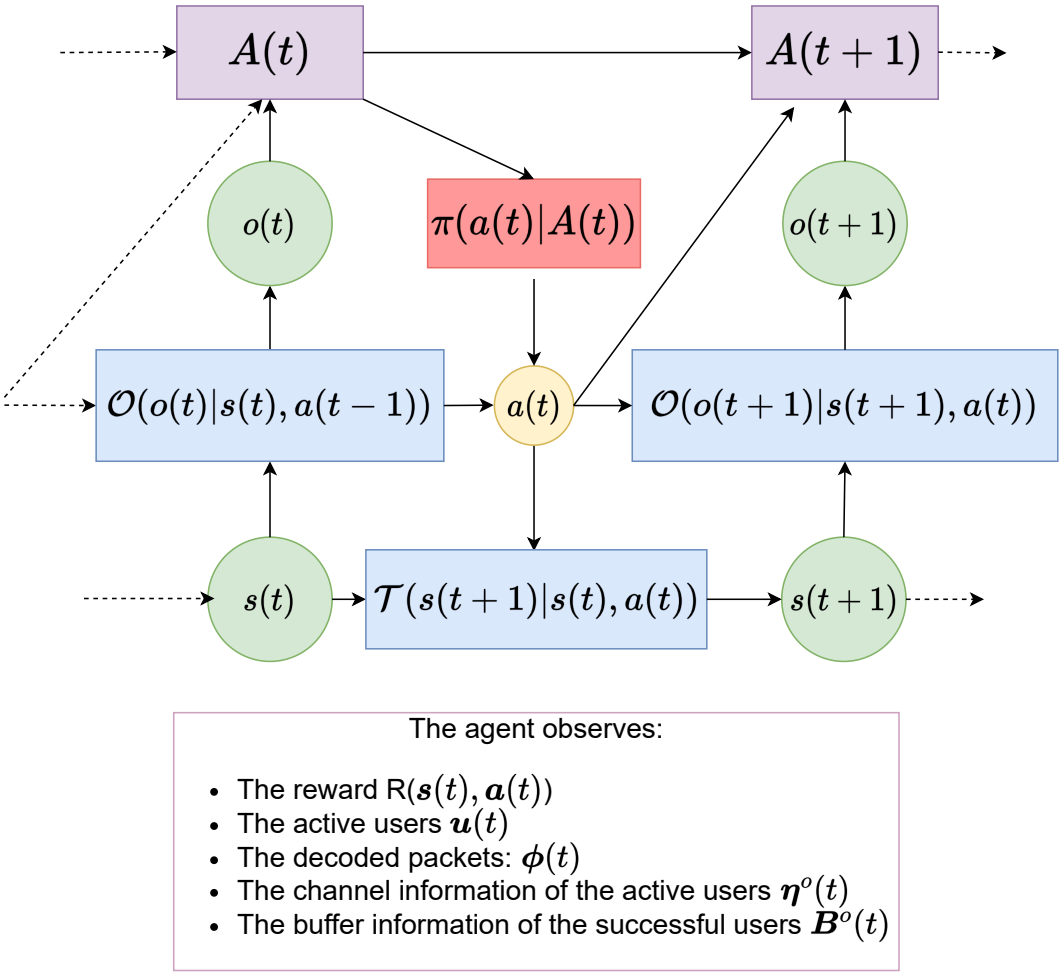


Figure 4.2: Formulation of the NOMA-URLLC problem.

is indeed known to be optimal in various deterministic [Stankovic et al., 1998] and stochastic (see e.g. [Moyal, 2013]) settings. We thus adapt it to NOMA as follows: given the devices' buffers estimates, $\boldsymbol{B}^A(t)$, EDF schedules the B users with the smallest head-of-line delay $d_k^h(t)$.

$$\text{EDF}(\boldsymbol{B}^{\boldsymbol{A}}(t)) = (a_1, \ldots, a_K), \tag{4.23}$$

$$\text{where } a_k = \begin{cases} 1 & \text{if } k \in \arg_B \min(\{d_1^h(t), \ldots, d_K^h(t)\}) \\ 0 & \text{otherwise} \end{cases}$$

Note that in our POMDP problem, EDF cannot be implemented in practice, as it requires full observability of the system, but can serve as a valuable benchmark. We can further allow the scheduler to take into account the channel state, by introducing a prior regarding the channel quality. In particular, we define a prior on the channel f_{ch} as follows:

$$f_{\text{ch}}(\boldsymbol{\eta}^{\boldsymbol{A}}(t), \boldsymbol{\tau}^{\boldsymbol{a}}) = (a_1, \dots, a_K), \tag{4.24}$$
$$\text{where } a_k = \begin{cases} 0 & \text{if } \eta_k^A \leq \eta^* \text{ and } \tau_k^a \leq \tau^* \\ 1 & \text{otherwise} \end{cases}$$

where $\eta^* \geq 0$ and $\tau^* \geq 0$ are hyperparameters to determine the quality of a channel. Typically, η^* is the threshold that indicates when a user will not be decoded with a high probability, regardless of the others' channels and τ^* is the coherence time that indicates whether the last information we have on the channel is relevant or outdated. The intuition behind this prior is that a user should remain inactive if it experiences a very "bad" channel.

The resulting prior f is thus a combination of the EDF and channel prior:

$$f(\boldsymbol{a}; \boldsymbol{A}) = EDF(\boldsymbol{B}^{\boldsymbol{A}}(t)) \odot f_{\text{ch}}(\boldsymbol{\eta}^{\boldsymbol{A}}(t), \boldsymbol{\tau}^{\boldsymbol{a}}) \tag{4.25}$$

In order to incorporate this prior knowledge into the RL agent, we introduce a Bayesian policy inspired by [Titsias and Nikoloutsopoulos, 2018]. We express the posterior policy $q(\boldsymbol{a}|\boldsymbol{A}; \theta_\pi)$ as a function of the prior over the agent state $f(\boldsymbol{a}; \boldsymbol{A})$ and the task specific policy $\pi(\boldsymbol{a}|\boldsymbol{A}; \theta_\pi)$ parameterized by θ_π with the Bayes rule:

$$q(\boldsymbol{a}|\boldsymbol{A}; \theta_\pi) \propto \pi(\boldsymbol{a}|\boldsymbol{A}; \theta_\pi) \odot f(\boldsymbol{a}; \boldsymbol{A}) \tag{4.26}$$

4.4.3 Algorithm Overview and Architecture

The neural network architecture is described in Figure 4.3. NOMA-PPO uses two neural networks, one for the policy and one for the critic. The input vector is the concatenation of the preprocessed buffer information $\boldsymbol{B}^A(t)$, the normalized timing information $\boldsymbol{\tau}^p(t)$, $\boldsymbol{\tau}^a(t)$, $\boldsymbol{\tau}^s(t)$, the channel information $\boldsymbol{\eta}^A(t)$ and the last reward $r(t-1)$. Its size is thus $5K + 1$.

Following a branching architecture, the policy network produces activation

probabilities for each user, generating K outputs as follows: $\pi_\theta(\boldsymbol{a}|\boldsymbol{A}) = (\pi_\theta(a_1|\boldsymbol{A}), \pi_\theta(a_2|\boldsymbol{A}) \ldots, \pi_\theta(a_K|\boldsymbol{A}))$. Inspired by the BDQ architecture [Tavakoli et al., 2018], which employs this approach for Q-learning, we handle the combinatorial action space in a manner that scales linearly with the number of users and adapt it to the PPO algorithm. These K outputs are then coordinated by a first block of hidden layers that are shared by all *branches*. This design balances the complexity of the model with the need to capture inter-dependencies among actions. While the single-layer branches simplify the model and enhance efficiency, the shared layers ensure that the critical inter-dependencies of our scheduling problem are captured.

On the other hand, the value network follows the same architecture of the policy network, except that it outputs a single value for the state value estimation. The procedure for training NOMA-PPO is developed in Algorithm 5. The training process for the algorithm consists of two phases: an initial offline training using synthetic data until satisfactory performance is achieved, followed by the deployment in the real environment where continuous updates of the policy and value network are performed in parallel, based on data collected during operation. Moreover, as system parameters evolve or change, periodic maintenance or retraining of the algorithm is necessary, either on a scheduled basis or when a decline in performance is observed.

4.5 Experiments

4.5.1 Simulation Settings and Implementation Details

Simulations are conducted at the MAC layer. Our simulation settings (see Table 4.3) adopt the parameters of the factory automation use case of the 3GPP 5G NR specifications on URLLC [3GPP, 2018b] and industrial IoT [3GPP, 2018a]. Our radio frame is made of five time-slots ($T_f = 5T_s$), whose duration T_s is equivalent to an OFDM symbol in NR. It can be decomposed into an information part of duration T_i and a cyclic prefix of duration T_{cp}, which both depend on the subcarrier spacing Δf: $T_s = T_i + T_{cp}$ with $T_i = 1/\Delta f$. From the signal bandwidth we substract the subcarriers dedicated to uplink pilots, so that, when there are U polled devices and n_p pilots per device, the number of complex channel uses is $n = (W - n_p U \Delta f) T_i$ [Tse and Viswanath, 2005, Chapter 5]. The

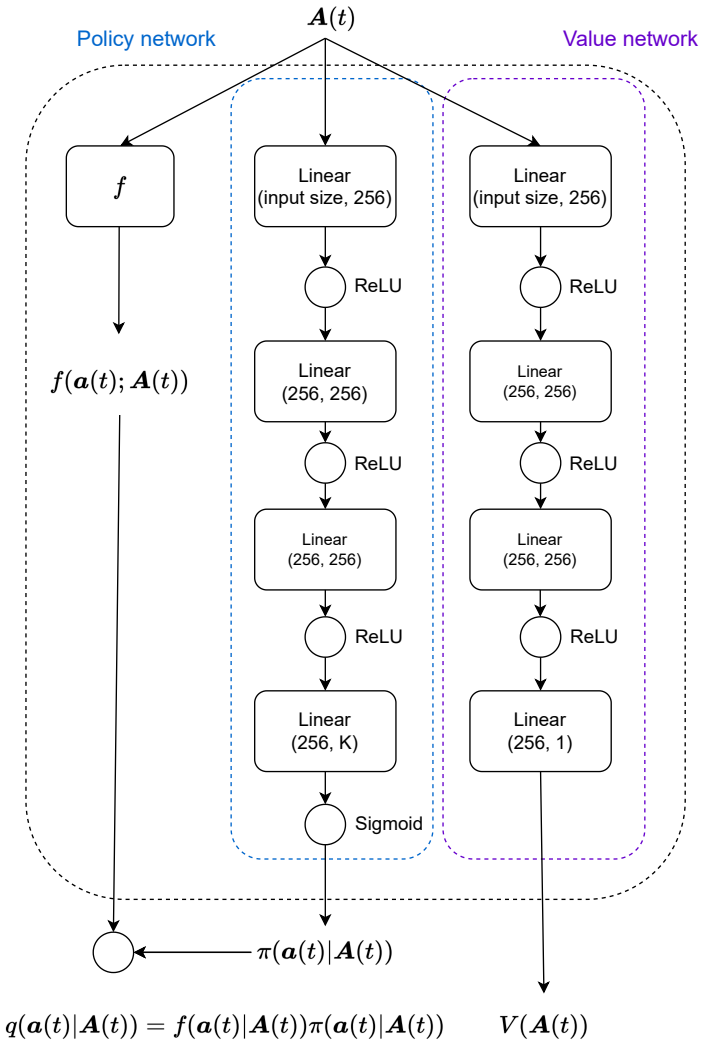


Figure 4.3: Architecture of the NOMA-PPO agent.

number of pilots per device can be obtained as follows: $n_p = \lceil W/W_c \rceil$, where $W_c = 1/(2T_d)$ [Tse and Viswanath, 2005, Chapter 2] is the coherence bandwidth and T_d is the delay spread.

Regarding the traffic model, we consider either a deterministic periodic traffic with period $1/\lambda$ or a probabilistic aperiodic traffic with average inter-arrival time $1/\lambda$. A packet can be decomposed into an information part of length L_i, a header part of length L_h, and a buffer description of length L_b, so that $L = L_i + L_h + L_b$. In URLLC, headers cannot indeed be neglected with respect to the message length. We assume that the information part, the header and the buffer

Algorithm 5: NOMA-PPO for URLLC uplink scheduling in NOMA systems.

Input: prior f, initial parameters of the policy network π_{θ_0} and the value network V_{φ_0};
for $j = 1, 2, \ldots, J$ **do**
  Run the posterior policy q_{θ_j} and collect a set of β trajectories $\{(\boldsymbol{A}_b(t), \pi_{\theta_j}(\boldsymbol{a}_b(t)|\boldsymbol{A}_b(t)), r_b(t))_{t=1,\ldots,T}\}_{b=1\ldots\beta}$.
  Compute the rewards-to-go $\hat{R}_b(t)$ for each trajectory: $\hat{R}_b(t) = \sum_{t'=t}^{T} \gamma^{t'} r_b(t')$
  Compute the values $V_{\phi_j}(\boldsymbol{A}_b(t))$ using the value network.
  Compute the advantage estimates $\hat{A}_b^{GAE}(t)$.
  Update the policy network by maximizing (2.26) with the Adam algorithm [Kingma and Ba, 2014]:

$$\theta_{j+1} = \arg\max_{\theta} \frac{1}{\beta T} \left[\sum_{b=1}^{\beta} \sum_{t=1}^{T} \min \left(\frac{\pi_\theta(\boldsymbol{a}_b(t)|\boldsymbol{A}_b(t))}{\pi_{\theta_j}(\boldsymbol{a}_b(t)|\boldsymbol{A}_b(t))} \hat{A}_b^{GAE}(t), g(\nu) \hat{A}_b^{GAE}(t) \right) \right]$$

  Update the value network by minimizing the mean-squared error with the Adam algorithm:

$$\varphi_{j+1} = \arg\min_{\varphi} \frac{1}{\beta T} \sum_{b=1}^{\beta} \sum_{t=1}^{T} \left(V_\varphi(\boldsymbol{A}_b(t)) - \hat{R}_b(t) \right)^2 \tag{4.27}$$

information are jointly encoded [Popovski et al., 2019]. The traffic parameters of Table 4.3 are taken from the factory automation use case of Release 16 [3GPP, 2018b].

For realistic numerical experiments, we partly adopt the scenario proposed in [3GPP, 2018b, Table A.2.2-1] for the factory automation use case, with a single BS. The network layout is a rectangle of size $\ell \times \ell'$; the BS is positioned at its center at a height $\tilde{h}_b$ and serves devices, each at height $\tilde{h}_d$ and moving with velocity v. Devices are uniformly distributed within the network area. Devices and BS benefit from antenna gains G_b and G_d respectively. For a speed of $v = 3$ km/h, we obtain a coherence time of $T_c = c/(8 f_c v) = 11.2$ ms, which corresponds to 63 radio frames. We choose an episode length of 200 frames that allows us to consider speeds below 1 km/h. The path-loss model is the ITU InH NLOS [3GPP, 2017g]. The BS has a noise figure N_F, so that the noise power

Table 4.3: Network Simulation Settings.

Parameter	Notation	Value
Carrier frequency	f_c	4 GHz
Bandwidth[a]	W	38.16 MHz
Subcarrier spacing	Δf	30 kHz
Delay spread[b]	T_d	100 ns
OFDM symbol information part	T_i	33.33 μs
OFDM symbol cyclic prefix[c]	T_{cp}	2.34 μs
SIC limitation	B	3
Information length	L_i	32 bytes
Headers[d] length	L_h	46 bytes
Buffer information[e]	L_b	14 bytes
Average inter-arrival rate	$1/\lambda$	2 ms
Deadline	δ	1 ms
Network layout	$\ell \times \ell'$	50×120 m^2
Noise Power Spectral Density	N_0	-174 dBm/Hz
BS noise figure	N_F	5 dB
BS antenna height	$\tilde{h}_b$	3 m
BS antenna gain	G_b	5 dBi
BS number of antennas	n_a	4
Device transmit power	p	23 dBm
Device antenna height	$\tilde{h}_d$	1.5 m
Device antenna gain	G_d	0 dBi
Device speed	v	3 km/h

[a] For a channel bandwidth of 40 MHz, the signal occupies 38.16 MHz after having excluded guard bands [3GPP, 2017a].
[b] Typical delay spread for an indoor hot-spot scenario with carrier frequency 4 GHz [3GPP, 2018b].
[c] Normal cyclic prefix duration for symbols not at the start or in the middle of the subframe [3GPP, 2017e].
[e] Size of an array of size 56 corresponding to a maximum deadline of 56 frames, i.e., 10 ms, with 3 bits entries giving the number of packets for every deadline.
[d] Headers include 2 bytes of CRC [3GPP, 2017c], 1 byte for MAC [3GPP, 2017b], 0 byte for RLC [3GPP, 2017f], 2 bytes for PDCP [3GPP, 2017d], 0 byte for SDAP [3GPP, 2020] and 40 bytes for IPv6 [Deering and Hinden, 2017].

is $\sigma_n^2 = N_0 W N_F$, where N_0 is the noise power spectral density. Typical values for the channel parameters are given in Table 4.3. We express the deadlines and

inter-arrival time in term of frames. Indeed, given the frame duration T_f, we can deduce that the average inter-arrival time of 2 ms corresponds to 11.2 frames and that the deadline of 1 ms to 5.6 frames.

The parameters of the DRL algorithms are given in Table 4.4. We preprocess the agent state as follows. In order to reduce the dimension of the buffer information, the matrix $\boldsymbol{B}^A(t)$ is transformed into a vector of size K of head-of-line delays for each agent. In order to improve stability of speed up training, we normalize $\boldsymbol{\tau}^p, \boldsymbol{\tau}^a, \boldsymbol{\tau}^s$ between 0 and 1 by taking $1/\boldsymbol{\tau}^p, 1/\boldsymbol{\tau}^a, 1/\boldsymbol{\tau}^s$. Finally, the channel threshold η^* is calculated using (4.8) such that the error probability in absence of interference corresponding to η^* is equal to 10^{-5}.

Table 4.4: Parameters of the DRL algorithms.

Parameter	Value
Input size (H_{in})	$5K + 1$
Hidden size (H)	256
Discount factor (γ)	0.3
Learning rate actor	10^{-4}
Learning rate critic	10^{-3}
Batch size	128
History length	K
Episode length (T)	200 slots
Training length (J)	10k episodes
Activation functions	ReLU
Number of seeds	5
λ_{GAE}	0.95

URLLC score

In order to compare our algorithm to the traditional benchmarks, we define the *URLLC score* as the number of successfully transmitted packets over the number of received packets. In the following experiments, the URLLC score is computed over 500 episodes which corresponds to approximately $2 \cdot 10^5$ generated packets according to the traffic parameters in Table 4.3. Therefore, a URLLC score of 1 means that the reliability is greater than $1 - 10^5$.

4.5.2 Benchmarks

For all baselines, when the BS receives two or more packets at the same time, we use the SIC procedure described in Section 4.2.2 to decode the packets.

- **Random Scheduler**: This scheduler schedules a subset of B devices uniformly at random.

- **EDF**: This scheduler schedules pending packets in the increasing order of their deadlines, see (4.23). Again, it cannot be implemented in practice on the uplink because of the assumed full observability of the device buffers.

- **SA-NOMA-SIC**: This baseline is a grant-free approach that follows the work of [Tegos et al., 2020]. It combines SA with SIC. At each frame, devices transmit their packet with the same probability p. Regarding re-transmissions, we use the *proactive* scheme [Mahmood et al., 2019a]: a user can re-transmit the same packet with probability p until it is delivered or expired. The probability p is empirically optimized such that the URLLC score is maximized for every scenario.

- **RDQN-NOMA Scheduler**: The standard DQN algorithm proposed by [Hausknecht and Stone, 2015] is the traditional approach to solve POMDP problems. The idea is to use an RNN to handle partial observability. We directly apply this algorithm in order to solve (P). The action space of the RL agent is the set of combinations of B or more devices to poll.

- **Branching DQN (BDQ)**: this baseline is a version of the Dueling Double DQN algorithm from [Tavakoli et al., 2018] that uses a branching architecture in order to handle a combinatorial action space.

- **iDRQN-NOMA**: This baseline is a fully distributed MARL algorithm for grant-free multiple access that follows the solution of [Xu et al., 2020] where each device is modeled by a Deep Q-network and decides to access the medium based on its local information: the state of its buffer and its channel state. This baseline uses a RNN, a GRU layer [Chung et al., 2014] in particular, as it is a standard approach to tackle partial observability. Additionally, we extend the work of [Xu et al., 2020] to NOMA systems by

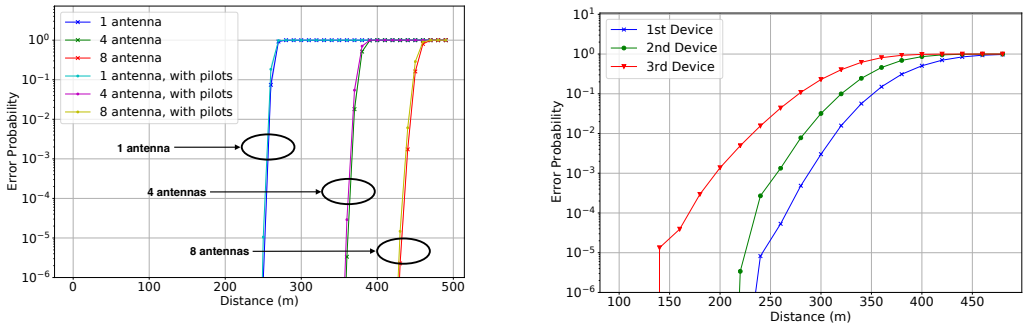
adapting the reward function as follows: at the end of every frame t, every user k receives the same reward:

$$R^k(s_k(t), a_k(t)) = \begin{cases} \sum_{i \in \mathcal{U}(t)} \phi_i(t) & \text{if } |\mathcal{U}(t)| \leq B \\ -1 & \text{otherwise} \end{cases} \tag{4.28}$$

- **NOMA-PPO-no-prior**: This baseline is the proposed approach, however without using prior information over the agent state.
- **NOMA-PPO-no-agent-state**: this approach is our NOMA-PPO algorithm without the agent state. It deals with partial observability using the action observation history as the input to the policy. It then processes it using a recurrent neural network (RNN) [Hausknecht and Stone, 2015].

In order to be fair in the experiments, we modify the frame structure of the two grant-free approaches SA-NOMA-SIC and iDRQN-NOMA and divide it into four time-slots of duration T_s: an uplink transmission symbol, a guard symbol, a downlink ACK/NACK and a guard symbol.

4.5.3 Study of the Channel Model



(a) For a single device.

(b) For three devices after SIC.

Figure 4.4: Packet error probability ε as a function of the distance to the BS.

In this section, we study the behavior of the channel. In Figure 4.4a, we show the channel error probability ε as a function of the distance between a device and the BS, involved in a point-to-point transmission without interference. Results

are shown for different number of antennas at the BS and with or without the pilot signals. We see that there are roughly three regimes that can be distinguished. When the distance is small, the error probability is very small (less than 10^{-6}). When the distance to the BS is too large, the error probability is close to 1. In this regime, there is no hope to guarantee URLLC requirements. In an intermediate regime that depends on the number of antennas and the number of decoded devices, the error probability is not negligible but the URLLC requirements could be met with an appropriate scheduling. In this case, the SINR model is required to benefit from the channel evolution for every device. As expected, increasing the number of antennas at the BS improves the reliability. At last, reserving some resource for pilots has a negligible influence on the performance.

In Figure 4.4b, we show the error probability as a function of the distance of the three devices from the BS. The three devices transmit simultaneously to the BS, which performs SIC. The resulting error probabilities are shown for the first, second and third decoded signals respectively. We again observe the three regimes, however with an offset according to the rank of decoding. The intermediate regime ranges here approximately between 150 m and 300 m.

4.5.4 Convergence Analysis

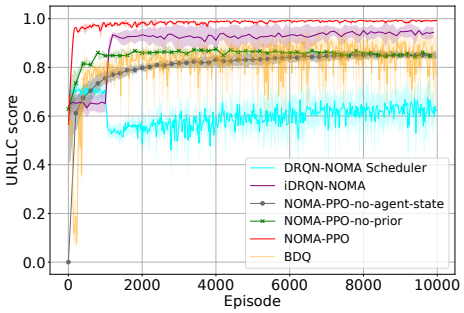
In Figure 4.5a, we show the evolution of the URLLC score during the training of 18 learning agents under the probabilistic aperiodic traffic. First, we can see that NOMA-PPO converges the fastest and with the smallest variance to its asymptotic value. Second, we see that not only does the prior help NOMA-PPO reach a better optimum, it also increases the convergence speed and reduces the variance. Third, we can observe that NOMA-PPO-no-prior's performance closely aligns with NOMA-PPO-no-agent-state. This suggests that utilizing the agent state achieves the same benefits as managing partial observability with a RNN, but with reduced complexity. However, we can see that training NOMA-PPO-no-agent-state is longer, primarly due to the higher number of parameters in the GRU layer. Fourth while the MARL grant-free approach, iDRQN-NOMA, reaches the second best optimum in terms of URLLC score, it converges slower than NOMA-PPO and with a larger variance. This can be accounted for by the fact that agents must coordinate independently, solely using the BS's feedback. Furthermore, we observe that the DRQN-NOMA scheduler does not

manage to converge due to the combinatorial action space. Indeed, there are $2^K - \sum_{k=0}^{B-1} \binom{K}{k} = 261,972$ possible actions for $K = 18$ and $B = 3$, thus choosing the appropriate action is challenging. In light of the lack of convergence of this algorithm, we exclude it from future experimental baselines. Finally, we observe that the BDQ algorithm comes third in term of asymptotic URLLC score but suffers from high variance. We notice that for the DRQN-NOMA scheduler and iDRQN-NOMA, the score does not evolve in the first thousand episodes. It is because of the “warm up" stage where we collect trajectories in order to fill the replay buffer of the agents without updating them.

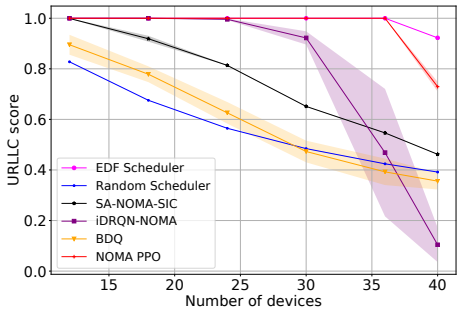
4.5.5 Performance in the 3GPP Scenario

In Figure 4.5, we study the performance of our algorithm in the 3GPP scenarios with two different traffic models. On the one hand, we study in Figure 4.5b the evolution of the URLLC score as a function of the number of devices on the deterministic periodic traffic and on the other hand, the evolution of the URLLC score and the Jain's Index on the probabilistic aperiodic traffic in Figure 4.5c and Figure 4.5d respectively. The Jain's index is computed with the URLLC scores.

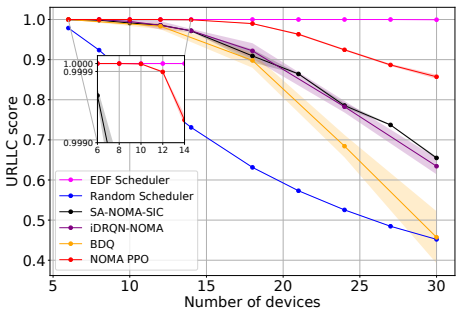
We observe that our approach, NOMA-PPO, consistently outperforms all benchmarks in URLLC score and fairness across various scenarios, with the exception of the EDF scheduler, which benefits from full observability over devices' buffers. This superior performance can be explained by our technical contributions to better handle partial observability and the combinatorial action space, augmented with prior knowledge. Indeed, we first observe that our proposed solution surpasses the BDQ algorithm, particularly in high-density device scenarios, where it often converges to suboptimal policies. While BDQ can handle large action spaces, its inability to effectively manage partial observability leads to the convergence to suboptimal policies as the number of users increases. In contrast, NOMA-PPO successfully mitigates the issues of partial observability thanks to the integration of the agent state. Regarding grant-free methods, we observe distinct behaviors under different traffic patterns. For instance, the iDRQN-NOMA algorithm, for instance, struggles to converge with over 30 devices in deterministic periodic traffic but outperforms SA-NOMA-SIC for up to 18 users in probabilistic aperiodic traffic. After exceeding this user threshold, the



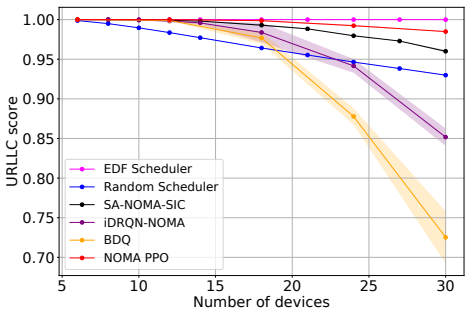
(a) Evolution of the URLLC score during training for 18 users.



(b) URLLC score in the 3GPP deterministic periodic scenario.



(c) URLLC score in the 3GPP probabilistic aperiodic scenario.

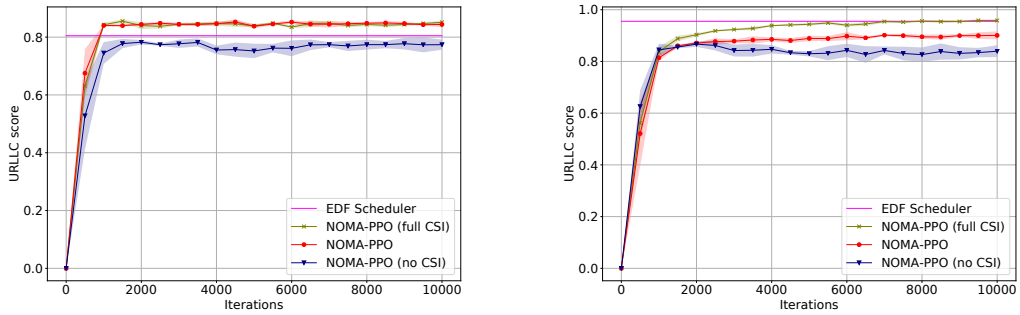


(d) Jain's fairness index in the 3GPP probabilistic aperiodic scenario.

Figure 4.5: Performance metrics in the 3GPP scenario.

performance deteriorates as coordinating a high number of users becomes challenging. This complexity leads to instabilities inherent to independent learning, that suffers from non-stationarity arising from concurrent learning processes. In comparison, NOMA-PPO offers a central control mechanism for transmissions, which proves advantageous over distributed methods, especially as these tend to result in increased collisions with a growing number of users. Additionally, its capability to incorporate prior information about the wireless system enables the discovery of more effective policies in scenarios with an increasing number of users. This feature ensures not just efficient scheduling of devices with favorable channel conditions but also prioritization of packets close to their deadlines.

Finally, it is important to note that the DRL baselines, despite their potential, often get trapped in local optima, leading to suboptimal policies that even underperform compared to a random scheduler when the number of users increases.



(a) $T_c = 1.4$ms, 10 users. (b) $T_c = 0.34$ms, 10 users.

Figure 4.6: Evolution of the URLLC score during training under different channel conditions.

4.5.6 Performance in Different Channel Conditions

In this subsection, we analyze the behavior of NOMA-PPO under diverse channel conditions.

Figure 4.6 shows the evolution of the URLLC score during the training as a function of the number of iterations in the probabilistic aperiodic traffic of the 3GPP scenario where parameters are listed in Table. 4.3, for 10 devices. Besides, we test two different values for the deadline-coherence time ratio where packets have a deadline of 10 ms, a coherence time of 1.4 ms (Figure 4.6a) and 0.34 ms (Figure 4.6b).

We compare the NOMA-PPO agent with:

- **NOMA-PPO (full CSI)**: the version of NOMA-PPO where the channel information of all users is observable.
- **NOMA-PPO (no CSI)**: the version of the algorithm where we remove the channel information from the agent state.

First, we can see on all figures that the complete observability of the users' channels enriches the agent state and results in superior performance compared to the versions lacking this feature. Second, as depicted in Figure 4.6a, when the coherence time is long enough, NOMA-PPO manages to accurately estimate the true channel based on the agent state. This enables it to reach similar performance to NOMA-PPO with full CSI. Third, we observe in Figure 4.6b, when the coherence time is too short, that NOMA-PPO fails to reach the performance of

NOMA-PPO with full observability but still outperforms the one with no CSI. The reason is that the coherence time must be large enough compared to the deadline so that the algorithm has time to both sense the channel and schedule the packet when the channel conditions are favorable. Furthermore, we notice that NOMA-PPO outperforms the EDF scheduler in Figure 4.6a. The reason for this difference stems from the inherent design of EDF scheduler which focuses on serving packets based purely on their deadlines, neglecting the variability in channel conditions. Indeed, we observe that in Figure 4.6a, where the coherence time is longer, the channel remains relatively stable over several frames, which enables NOMA-PPO to exploit the channel conditions, thus demonstrating superior performance over EDF. In contrast, when the coherence time is short, as in Figure 4.6b, the rapidly fluctuating channel conditions render channel-based decisions less predictable and less exploitable. In such scenarios, the straightforward deadline-driven approach of EDF showcases robust performance.

Finally, we noted through simulations not depicted here, that when the number of users is too small, NOMA-PPO (no CSI) attains equivalent performance to both NOMA-PPO (full CSI) and NOMA-PPO. This can be explained by a multi-user diversity gain that increases with the number of devices.

4.5.7 Complexity Analysis

Table 4.5: FLOPs for the Deep Neural Networks.

Algorithm	FLOPs
NOMA-PPO	$3,072K + 263,424$
BDQ	$4,096K + 394,496$
iDRQN-NOMA (1 agent)	$406,528K$

We evaluate the complexity of our Deep Learning architecture in terms of Floating Point Operations (FLOP)s that occur during a single forward pass of the neural network. Given that a connection between 2 neurons involves 2 operations (a multiplication and an addition) the FLOPs of a linear layer operation is $2 \times$ input size $\times$ output size. The GRU layer is made of four operations: the reset gate, the update gate, the candidate hidden state and the new gate [Chung et al., 2014], each being made of matrix multiplications, additions, and activation functions. Let H_{in} the size of the input and H the size of the hidden layer. We

can express the FLOPs of each unit:

- The reset and update gate have the same structure: 2 matrix multiplications, 3 additions and 1 sigmoid activation. The resulting number of FLOPs for both gates is thus: $4H(H_{in} + H) + 6H$.
- The new gate contribution includes 2 matrix multiplications, a Hadamard product, a *tanh* activation and the addition of the bias terms: $2H(H_{in} + H) + 4H$.
- Finally, the new hidden state involves two Hadamard products, and two additions: $4H$.

In addition, the complexity of the GRU depends on the size of the history. In our problem, we set the history size to the number of users K. In total, the number of FLOPs of a GRU layer is $6HK(H_{in} + H) + 10HK$.

The number of FLOPs of the learning algorithms are given in Table 4.5 (using the numerical values of Table 4.4). First, we can see that all algorithms have a linear complexity in the number of users and differ by their slope. Second, the BDQ algorithm has a complexity greater than NOMA-PPO due to the use of an advantage and a value network during inference. Third, the complexity of one iDRQN agent is larger than the complexity of the centralized approach we propose due to the use of a GRU layer to process the action-observation history.

4.6 Conclusion

In this chapter, we propose a novel approach for satisfying URLLC requirements and strict deadlines in IoT networks, employing NOMA for uplink communications.

Our proposed approach, NOMA-PPO, addresses the challenges posed by the NOMA uplink URLLC scheduling problem, namely the combinatorial action space and the partial observability. NOMA-PPO tackles these challenges by bringing three technical contributions. First, it formulates the NOMA-URLLC problem as a POMDP, and introduces the concept of *agent state*, as sufficient statistic for the past actions and observations. This reformulation allows us to extend the state-of-the-art PPO algorithm to handle a combinatorial action space

thanks to a branching policy network. Finally, NOMA-PPO is able to incorporate prior knowledge over the system into the learning algorithm by employing a Bayesian policy. We demonstrate that our approach outperforms traditional MA and DRL benchmarks in 3GPP scenarios for different traffic models (probabilistic aperiodic and deterministic periodic) in terms of URLLC score, fairness, and convergence speed. Finally, we show that our algorithm is robust under diverse channel configurations and is capable to leverage channel information.

In the next two chapters, we explore how the GF access framework and how MARL can be applied to our URLLC problem.

Chapter 5

SeqDQN: Multi-Agent Deep Reinforcement Learning for Uplink URLLC with Strict Deadlines

Contents

5.1 Introduction

In this chapter, we propose a fully decentralized approach to solving the uplink MA problem with strict deadlines we have considered so far. We equip each device with a deep MARL algorithm in order to learn a transmission protocol by interacting with the other devices and the environment. First, we conduct a thorough analysis of the performance of traditional MARL algorithms, such as iDQN and QMIX, when applied to our specific problem. This analysis allows us to evaluate the strengths and weaknesses of the most used algorithms in the literature and use them as benchmarks on our problem. Second, we introduce SeqDQN, a distributed MARL solution inspired by the two-timescale training paradigm [Arslan and Yüksel, 2017] where agents do not update their Q-functions concurrently. In SeqDQN, devices update their Q-function sequentially, starting with the devices with the most stringent latency requirement. This training strategy aims to mitigate the challenges posed by non-stationarity emerging from IL when training decentralized agents.

Considering each user as a RL agent is the most natural approach to apply RL to MA. Several studies have investigated this problem. We can classify them in two categories: independent learning (IL) and centralized-training with decentralized execution (CTDE).

5.1.1 Independent Learning

Because of its ability to train fully decentralized agents, IL is the most used framework in MA. For instance, [Chang et al., 2018, Xu et al., 2020, Xu et al., 2018] apply IL to Dynamic Spectrum Access (DSA), where secondary users use

RL to learn a transmission protocol. Regarding coexistance with existing protocols such as TDMA and ALOHA, the authors of [Yu et al., 2019] show that iDQN can learn a throughput-optimal strategy while other nodes are traditional MA protocols. However, none of these papers take into consideration the theoretical shortcomings mentioned earlier. Furthermore, no previous work has applied IL to a MA problem to tackle a URLLC problem on the uplink with deadlines.

5.1.2 Centralized-Training with Decentralized-Execution

The other major MARL framework is CTDE and it has also been applied to MA. Tan et al. [Tan et al., 2021] justify the use of the CTDE architecture and in particular the centralized training by leveraging mobile edge computing. They apply the QMIX algorithm to tackle the DSA problem, modelling it as a Dec-POMDP problem [Oliehoek, 2012]. Another application of CTDE is the work of [Kassab et al., 2020] that applies the MADDPG algorithm [Lowe et al., 2017] to a DSA problem for event monitoring by IoT devices. In spite of the potential of CTDE for MA, it hasn't been applied to a problem with a strict latency constraint and the way to train a centralized critic remains an issue for the IoT because of the communication overhead.

5.1.3 Contributions

In this chapter, we tackle the uplink MA problem in the context of URLLC with a decentralized approach. We consider heterogeneous devices, modeled with a DRL agent, that need to transmit short packets to a BS on the uplink given a strict deadline. Our contributions are the following:

- We rigorously formulate the uplink URLLC multi-agent problem as a Dec-POMDP, incorporating the characteristics and constraints of the URLLC scenario.
- We implement and evaluate the most commonly used MARL algorithms in the literature, namely iDQN and QMIX, in the context of our uplink URLLC multi-agent problem.
- We introduce SeqDQN, a distributed MARL algorithm where agents do not update their Q-functions simultaneously. Instead, they update their

Q-function sequentially, starting with the devices with the most stringent latency requirement. The advantages of this method are: 1) We reduce the non-stationarity caused by multiple agents learning concurrently, which is a major drawback of IL, 2) our proposed method is more scalable to a large number of agents than CTDE and 3) training is much faster than the existing MARL algorithms (iDQN and QMIX).

- We evaluate our solution against traditional MA baselines and MARL algorithms across multiple scenarios within our system model, under a probabilistic and deterministic periodic traffic. We show that not only does our method outperform the MA baselines but it also reaches a URLLC performance equal to or surpassing those of established MARL benchmarks.

5.2 Problem formulation

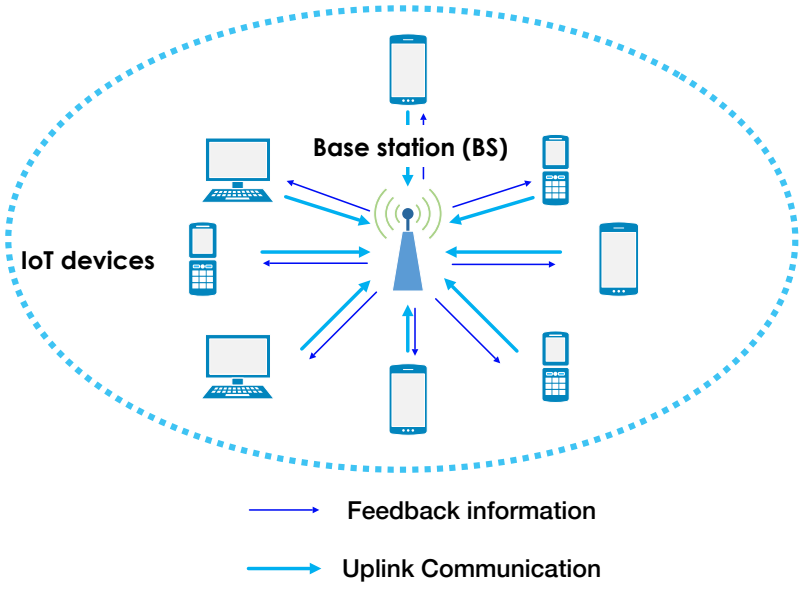


Figure 5.1: System Model With Heterogeneous Devices.

5.2.1 System model

We consider a network of K devices communicating with a BS over a wireless shared channel on the uplink (Figure 5.1). Time is slotted and at every slot, devices can choose to transmit a packet or to remain idle. All packets are supposed to require the same transmission time of one time slot and the propagation delay is assumed to be negligible. We assume that all devices are synchronized, i.e. aligned with a common slot start time. The BS provides downlink signals to facilitate this synchronization. Moreover, each device has an individual air interface latency constraint δ_k, such that a packet is dropped if it has not been transmitted within δ_k slots after its arrival in the buffer. We assume slot synchronization, i.e. all devices are aligned on a common start of each slot. Finally, we consider a collision channel model: if a collision occurs, the packets are not delivered to the BS but can be re-transmitted until their deadline is reached. After each slot, the users have access to what happened in the slot, i.e., whether a transmission was successful, the channel was idle or a collision occurred. This information is obtained by a feedback signal (ACK/NACK) broadcast from the BS to all the devices. In order to evaluate the different MARL algorithms, we define a *URLLC score* as the number of packets delivered before expiry divided by the number of generated packets.

5.2.2 Traffic model

We study the performance of MARL using two traffic models based on the framework of [Hou and Kumar, 2013] and the 3GPP [3GPP, 2018b].

Probabilistic Periodic Traffic

Inspired by [Hou and Kumar, 2013], we first consider a framework where the traffic pattern of every device is periodic, i.e., every period of N_p^k time slots, device k receives a packet with probability ξ_k. We allow the devices to be heterogeneous in the sense that they can have different packet arrival probabilities ξ_k, different periods N_p^k and they are not synchronous: Each device is assigned an offset parameter $\bar{f}_k \in [0, N_p^k]$, so that packet arrivals can occur only at time instants $\bar{f}_k + mN_p^k$, where m is an integer. At every slot $t \geq 0$, the probability for a device $k \in [1, K]$ of having a new packet is $\bar{\xi}_k(t|\bar{f}_k, \xi_k, N_p^k) = \mathbb{1}_{\{t[N_p^k]=\bar{f}_k\}}\xi_k$.

This traffic model is the same as the one in Chapter 3 with the difference that we now consider individual packet generation periods.

Deterministic periodic traffic

Defined in [3GPP, 2018b, Annex A], the deterministic periodic traffic is a special case of the previous framework with all arrival probabilities equal to 1. The main challenge of this framework is for the devices to learn the optimal schedule. This optimal schedule can be used subsequently by a contention-free grant-free access algorithm as standardized in 3GPP Release R15 for URLLC [3GPP, 2018b]. In the probabilistic and deterministic periodic traffic models, we assume that $\delta_k \leq N_p^k$ for all k and thus all devices have a one-packet buffer.

5.2.3 Decentralized Partially Observable Markov Decision Process

We formulate our URLLC problem as a Dec-POMDP [Oliehoek, 2012] where agents cannot see the full state and take actions based on their own observations.

In our case, let $\mathbb{S} = \mathbb{S}_1 \times \mathbb{S}_2 \times \cdots \times \mathbb{S}_K$ be the set of environmental states where $\boldsymbol{s_t} = (s_t^1, s_t^2, \dots s_t^K) \in \mathbb{S}$ is the concatenation of all individual observations. The devices have a buffer of size 1 and the local state of a device k at slot t is $s_t^k = (d_t^k, c_{t-1})$, where $d_t^k \in \mathbb{N}$ is the time in number of slots that the packet has already spent in the buffer and c_{t-1} is the last feedback from the BS. We set $c_{t-1} = 1$ if at $t-1$ a packet was successfully transmitted, -1 if a collision occurred and 0 if the channel was idle.

A local action of a device k is $a_t^k \in \mathbb{A}^k = \{0, 1\}$, where $a_t^k = 1$ if the device transmits and 0 otherwise. A device k makes an action according to a policy function $\pi^k : \mathbb{S}^k \to \Delta(\mathbb{A}^k)$. This function can be probabilistic or deterministic.

We now specify the reward function $\mathcal{R} : \mathbb{S} \times \mathbb{A} \to \mathbb{R}$. In a Dec-POMDP, agents have identical interests, which means that they have the same reward function. Users collectively want to maximize the URLLC score by maximizing the number of successful transmissions. At each slot t, agents get as reward $r_t = \mathcal{R}(\boldsymbol{s_t}, \boldsymbol{a_t})$ the ACK/NACK feedback from the BS ($+1$ if a packet is successfully transmitted, -1 if a collision occurred, 0 if the channel was idle). In other words, $r_t = c_t$. The objective for each agent k is to find a policy π^k that maximizes the expected

cumulative discounted reward over a finite horizon T:

$$\mathbb{E}_{\boldsymbol{s_{t+1}}\sim\mathcal{T},a^{-k}\sim\pi^{-k}}\left[\sum_{t=0}^{T}\gamma^t\mathcal{R}(\boldsymbol{s_t},(a_t^k,a_t^{-k})|a_t^k\sim\pi^k(\cdot|s_t^k),\boldsymbol{s_0}\right] \tag{5.1}$$

with $\gamma\in(0,1]$ the discount factor, and $\mathcal{T}:\mathbb{S}\times\mathbb{A}\mapsto\mathbb{S}$ the transition function. In our problem, we consider a finite horizon Dec-POMDP where an episode is of length T slots.

Note that it is also possible to model this system with a more general framework where agents have individual interests. However, we have chosen the Dec-POMDP framework as: 1) The main CTDE algorithms have been designed for Dec-POMDP only [Lowe et al., 2017, Sunehag et al., 2018, Rashid et al., 2018]; 2) We experimentally have observed that the performance of IL algorithms with our system model is very similar whether we use an individual or a collective reward.

5.3 Sequential DQN (SeqDQN)

Our proposed algorithm, called SeqDQN, is inspired by the idea of two-timescale training [Arslan and Yüksel, 2017] which tries to take advantage of iDQN without the non-stationarity issue. In SeqDQN, n DQN agents update their Q-function sequentially on different Exploration Phase (EP). During one EP, only one agent updates its policy while the others take actions according to their last learnt policy. Thus, we remove the non-stationarity caused by the other learning decision makers and each agent solves a POMDP problem during each EP in order to learn a best response to the others. The authors of [Arslan and Yüksel, 2017] show that this methodology provides a decentralized Q-learning framework where algorithms converge to equilibrium policies almost surely in fully observable weakly acyclic games.

To tackle our MA problem, we combine the two-timescale training with three elements. First, we use a GRU [Cho et al., 2014] to address partial observability in POMDP [Hausknecht and Stone, 2015]. Second, we create clusters of devices with the same deadline. Clusters are then trained sequentially from the smallest deadline to the largest one. When a cluster is trained, the users it includes update their policy one after the other until convergence. Third, devices that

have not been trained so far do not transmit to remove stochasticity during the EP and speed up the learning.

The pseudo-code of SeqDQN is shown in Algorithm 6. Inside a cluster, agents are trained sequentially G cycles. In an EP, when the policy of the learning agent has not improved in L episodes, we consider that the agent has converged and end the EP. We set the Q-function of the agent to the best one learnt during this EP in terms of total reward. We train the clusters J rounds so that the first trained clusters can adjust their policies to the ones with a larger deadline. At each round, the learning rate is decreased by a factor α. In practice, we set up the training sequence and clusters of users once at the beginning of the experiment. The BS groups the users by deadline and assigns them an ID corresponding to an EP where a device is allowed to update its Q-network. When an EP is finished, the BS communicates what device starts its EP via the feedback signal. There is no additional information exchange during an episode. The sequential training is a pre-processing set up that does not need to be repeated once the network is in operation. During the network operation, the decisions are fully decentralized.

Algorithm 6: SeqDQN for distributed multiple access

Initialize the Q-networks $Q_1, Q_2, \ldots, Q_N$;
Cluster the users in subsets $C_1, C_2, \ldots C_d$ with users in C_i having a deadline δ_i such that $\delta_i < \delta_{i+1} \forall i$.
for $j = 1, 2, \ldots, J$ **do**
    $\eta \leftarrow \eta \times \alpha$
    for $i = 1, 2 \ldots, d$ **do**
        if $i = 1$ **then**
            Set the policies of clusters $c > i$ to not transmit.
        Fix the policies of clusters $c < i$ to their current policy.
        for cycle $= 1, 2, \ldots, G$ **do**
            `/* Sequentially train the users of` C_i G `times */`
            for $m \in C_i$ **do**
                `/* EP of agent` m `*/`
                Run DQN for agent m and fix π^{-m}.
                Update Q^m with (2.10) and learning rate η.
                if π^m *has not improved in* L *episodes* **then**
                    End EP
                    Set the policy of π^m to the best one.

5.4 Experiments

5.4.1 Baselines

We compare the above algorithms to two baselines:

- **Contention-based grant-free access (GF) [Mahmood et al., 2019a]**: All devices with a packet to transmit access the channel with the same probability p. We optimize the transmission probability p experimentally for every number of agents such that the URLLC score is maximised. We consider the reactive scheme so that when a device receives a NACK feedback from the BS after transmitting, it will re-transmit the same packet with probability p until the reception of an ACK feedback or the expiry of the packet.

- **Round Robin (RR) scheduler**: This algorithm schedules devices in a cycle, so that the time resource is fairly shared between them. Devices are ranked in ascending order with deadlines plus offsets.

5.4.2 Simulation settings

We consider the three different settings for our experiments: a *dense probabilistic periodic traffic*, a *sparse probabilistic periodic traffic* and a *deterministic periodic traffic* from 4 to 28 users depending on the scenario. The parameters of these traffic models are given in Table 5.1 and the parameters of the MARL algorithms are given in Table 5.2. These numbers of devices are in line with the 3GPP recommendations for URLLC [3GPP, 2018b], where the use cases consider up to 10 users per cell. For every traffic model, periods N_p^k are chosen uniformly under the condition $\delta_k < N_p^k$. In Table 5.1, $\mathcal{U}$ designates the uniform distribution over a finite set or an interval. The hyper parameters have been optimized with grid search.

5.4.3 Convergence speed of MARL algorithms

We analyze in Figure 5.2 the evolution of the performance of the three MARL algorithms during the training phase for the dense probabilistic periodic traffic

Table 5.1: Parameters of the traffic models

Parameters	**Probabilistic Dense**	**Probabilistic Sparse**	**Deterministic**
Arrival Probabilities	$\mathcal{U}\{0.2, 0.4, 0.6, 0.8\}$	$\mathcal{U}\{0.05, 0.1\}$	1
Deadlines	$\mathcal{U}\{5, 10, 15, 20\}$	$\mathcal{U}\{2, 4\}$	$\mathcal{U}\{4, 6, 8, 10\}$
Periods (N_p^k)	$\mathcal{U}\{10, 20\}$	$\mathcal{U}\{5, 10\}$	$\mathcal{U}\{8, 10\}$
Offsets	$\mathcal{U}[0, 4]$	$\mathcal{U}[0, 4]$	$\mathcal{U}[0, 4]$

with 12 devices. The conclusions are similar for other traffic models and numbers of devices. Instead of comparing the algorithms as a function of the number of episodes, we present results as a function of the number of gradient updates in the x-axis. The reason is that iDQN and QMIX perform as many updates as the number of devices in every episode, whereas SeqDQN trains a single device. The number of gradient updates is thus more representative of the convergence speed.

First, we can see that SeqDQN manages to reach a better optimum at the end of the training. Second, we can see that it is the fastest algorithm to converge. This can be explained by two arguments. When a cluster of devices is being trained, the devices with a higher deadline are inactive. Therefore, the stochasticity of the environment is removed so it is easier for the learning devices to converge. Moreover, as only one agent explores at the same time, exploration is not affected by the noise caused by the concurrent learning of other agents as it is the case for iDQN and QMIX. Third, we observe that the training of SeqDQN has more variance than QMIX's. This can be explained by the fact that training is not centralized and devices are not trained simultaneously. Additionally, when a new agent starts its EP, it needs to adapt to the new policies of the other devices which creates variance. Reducing the learning rate at every round helps mitigating this effect. Finally, we notice that QMIX outperforms iDQN in terms of convergence speed and optimum attained, which is expected as centralized training is supposed to encourage cooperation.

Table 5.2: Parameters of the Q-learning algorithms

Parameter	Value
Discount factor (γ)	0.9
Initial learning rate (η)	10^{-3}
Learning rate decrease factor (α)	0.2
Batch size	128
History length	20
Episode length (T)	200 slots
Training length	50k episodes
Final exploration rate (ε)	0.1
Number of cycles (G)	5
Convergence criterion (L)	300
Training rounds for clusters (J)	3
Q-network architecture	1 GRU layer + 3 Linear layers
Activation functions	ReLU
Linear layer	100 neurons
GRU layer	100 neurons
Number of seeds	5

5.4.4 URLLC score

We run simulations with a dense probabilistic periodic traffic (Figure 5.3a), a sparse probabilistic periodic traffic (Figure 5.3b) and a periodic deterministic traffic (Figure 5.3c). We show the evolution of the URLLC score as a function of the number of devices. First, it is clear that the MARL algorithms outperform the MA baselines (GF and RR) in every scenario. We observe that the GF protocol performs very poorly in dense settings when the load is high (Figure 5.3a and Figure 5.3c) while it performs very well in a sporadic traffic when the load is lower (Figure 5.3b). Indeed, the larger the number of packets to transmit, the larger the number of collisions, thus the lower the performance of the GF protocol. On the contrary, we can see in Figure 5.3a and Figure 5.3c that the RR scheduler performs better when the load is high as 1) it schedules one packet at a time and thus avoids collisions; 2) the probability of scheduling a user with a packet is higher. Nevertheless, it performs very poorly in the sparse periodic traffic (Figure 5.3b) as it is not aware when a device has a packet to

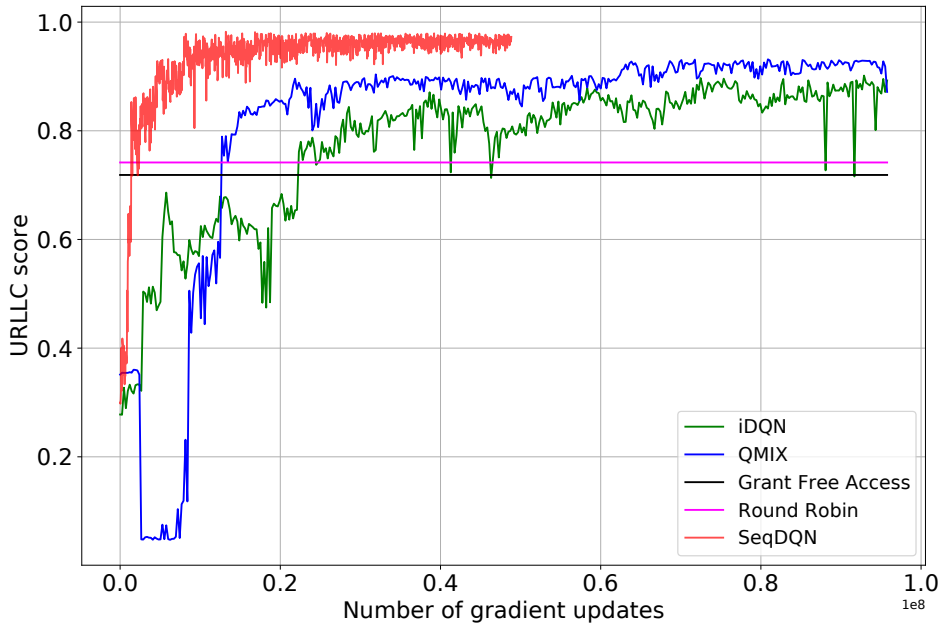
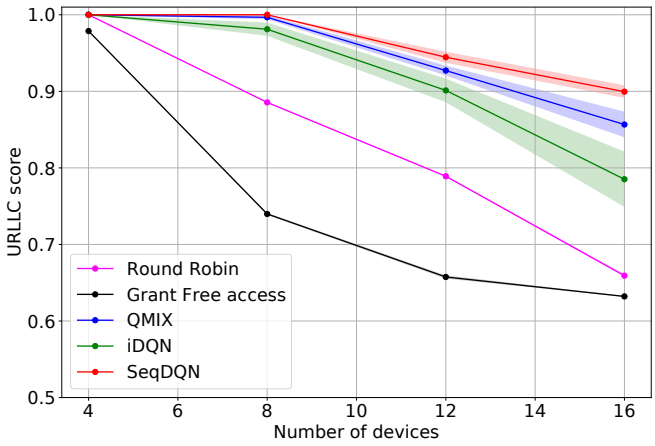
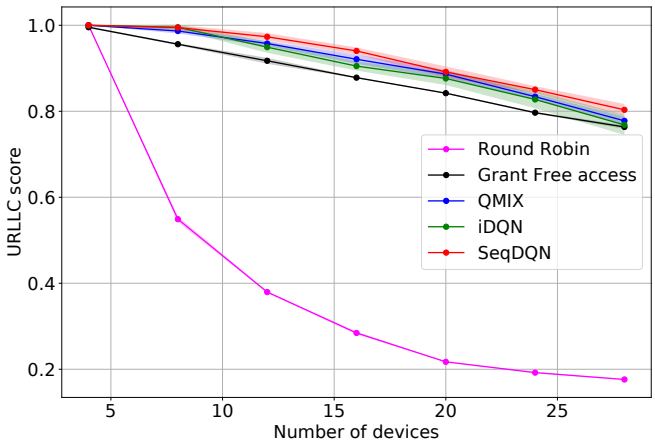


Figure 5.2: Evolution of the URLLC score during the training for the dense probabilistic periodic traffic with 12 devices.

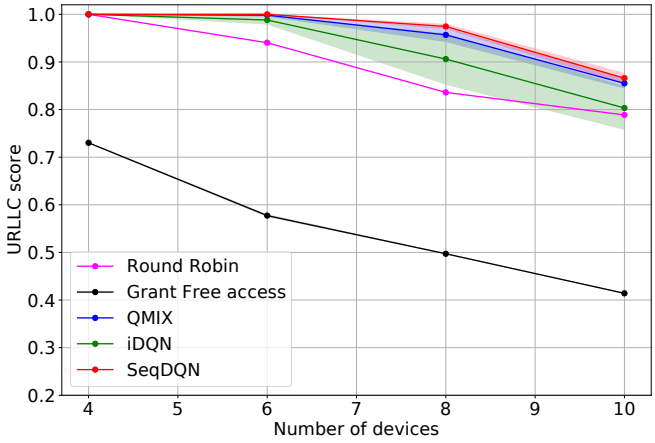
transmit. Furthermore, we can notice that in all three scenarios, SeqDQN not only outperforms iDQN and QMIX, but it also has less variance when we change the initialization of the neural networks. This can be explained by the fact that in SeqDQN, only one agent explores at a time whereas in iDQN and QMIX, all agents explore concurrently which makes an equilibrium harder to reach. We also observe that the variance of iDQN increases with the number of devices. Indeed, positive rewards become quite sparse as they are obtained when one device is active while all other ones remain idle. This problem is known in the literature on MARL, see e.g. [Lowe et al., 2017]. Finally, we note that in some cases, QMIX's performance is similar to SeqDQN's. The centralized training that encourages agent coordination can explain this. However, it necessitates a lot of communication during training, whereas SeqDQN avoids this problem because training is decentralized.



(a) Dense periodic probabilistic traffic.



(b) Sparse periodic probabilistic traffic



(c) Periodic deterministic traffic.

Figure 5.3: Evolution of the URLLC score with respect to the number of devices for various traffic models.

5.5 Conclusion

In this chapter, we model the uplink MA problem in the context of URLLC, characterized by strict deadlines as a Dec-POMDP. We investigate distributed solutions around the application of MARL to learn efficient transmission protocols tailored to this URLLC context. We assess the performance of two prominent MARL algorithms, iDQN and QMIX, and in response to the challenges posed by scalability and non-stationarity inherent to multi-agent environments, we introduce our novel distributed algorithm SeqDQN.

SeqDQN offers three advantages over existing MARL frameworks. First, by prioritizing devices with the most time-critical information to learn their transmission protocols first, SeqDQN attains a superior operating point. Second, the introduction of sequential agent training accelerates the overall training process. Third, the progressive training of agents and policy fixing for the already trained agents allow us to handle a larger number of devices.

However, even if the MARL approaches exhibit promising results for uplink URLLC, they suffer from substantial limitations. First, all three MARL algorithms suffer from the absence of theoretical guarantees which raises concerns regarding their real-world deployment, as algorithms may exhibit divergence as we increase training steps or the number of devices. Second, as mentioned in Chapter 3, the learning algorithms are not adapted when the traffic patterns become less predictable, with Poisson arrivals for instance. Indeed, coordinating with other devices becomes more complex, as individual devices lack knowledge about the other's state.

We try to address these challenges in Chapter 6 by extending our uplink URLLC framework to Dynamic Multi-Channel Access.

Chapter 6

Multi-Agent Proximal Policy Optimization for Dynamic Multi-Channel URLLC Access

Contents

6.1 Introduction

In this chapter, we build upon the uplink URLLC framework introduced in Chapter 5, extending its application to Dynamic Multi-Channel Access (DMCA). Recognizing the challenges posed by aperiodic traffic patterns to MARL algorithms, we exploit the expansive 40 MHz bandwidth available in industrial IoT scenarios [3GPP, 2017g]. By dividing this bandwidth into multiple orthogonal frequency subchannels, we enrich the system model, providing a more complex and diverse environment for MARL algorithms to express their potential. After reviewing the literature about DMCA, we found that existing research has yet to fully resolve the complexities of DMCA in URLLC networks, especially under conditions of time-varying heterogeneous channels and diverse traffic profiles. To address this gap, we introduce an innovative approach based on Deep MARL. Our methodology leverages the theoretical framework of TRPO in a multi-agent setting to meet the specific challenges and requirements of the URLLC-DMCA problem.

6.1.1 Related Work

Traditional random access protocols have been extended to the DMCA problem in order to meet the URLLC requirements. For example, the work of [Qi et al., 2020] proposes a multi-channel ALOHA-type GF algorithm. However, the authors assume that the users are aware of all the channel states and thus only good channels are selected. Furthermore, this approach does not adapt to the dynamics of the environment and is therefore sub-optimal.

DRL approaches have also been considered to tackle the DMCA problem. The most natural way to extend single-agent DRL to multi-agent DRL is IL [Tan, 1993] as mentioned in the previous chapters. The most notable examples of IL applied to the DMCA problem are the work of [Zhong et al., 2019] that introduces an actor critic algorithm for DMCA, the P-DDPG algorithm [Wang et al., 2020], where the authors predict the channel state with a Channel Prediction Module and use this predicted value as prior information for the DRL agent. Some other works tackle the Multi-Channel Access (MCA) aspect without the time-varying aspect of the channels. For example, the work of [Sohaib et al., 2021] allows users to access multiple channels in one frame thanks to a branching architecture and

the authors of [Ye et al., 2021] combine Q-learning with a RNN in order to tackle the MCA problem for heterogeneous networks. However, these studies assume a single multiple-frequency channel, i.e. all users observe the same channel state which is not realistic in a wireless context. Moreover, they do not address the theoretical limitations of IL such as the non-stationarity caused by the concurrent learning of all agents and do not provide any convergence guarantees to their approaches.

Futhermore, IL has been applied to DSA where the channel state is good when it is not used by a primary user and bad otherwise. For instance, the work of [Naparstek and Cohen, 2019] proposes Q-learning based agents equipped with a RNN similarly to [Ye et al., 2021] to maximize the network utility in DSA. One alternative to address the limitations of IL is CTDE where agents are allowed to exchange information during training in order to reduce the issues of non-stationarity. The authors of [Tan et al., 2022] apply CTDE to the DSA problem. However, the users adopt the listen-before-talk mechanism to access the spectrum which is not suitable to URLLC networks due to the strigent latency requirements and the small packet size. Besides, even if CTDE approaches can help agents learn more effectively by leveraging global information, convergence guarantees are still an active area of research and agents do not necessarily converge to an optimal or even stable policies.

To summarize, prior research has not tackled the DMCA problem in URLLC networks under time-varying heterogeneous channels and traffic profiles. In addition, existing studies have predominantly explored off-policy algorithms (such as Q-learning or actor-critic approaches) for the DMCA problem and these methods suffer from theoretical guarantees in the multi-agent context. In contrast, TRPO techniques have showcased superior performance across a variety of tasks in both single-agent [Duan et al., 2016] and multi-agent DRL, be it in a IL paradigm with iPPO [de Witt et al., 2020] or a CTDE framework with Multi-Agent Proximal Policy Optimization (MAPPO) [Yu et al., 2022]. Yet, those approaches still lack the *monotonic improvement guarantees*, characterizing the TRPO methods. One reason for their good empirical success may be the parameter sharing and homogeneous agents. Recent work of [Kuba et al., 2021], provides a TRPO algorithm with monotonic improvement guarantees within the multi-agent framework. In this chapter, we adapt their theoretical model to address the URLLC-DMCA

problem, tailoring the approach to meet its specific challenges and requirements.

6.1.2 Contribution and Outline

In this chapter, our contributions can be summarized as follows:

- We formulate a DMCA problem in a URLLC network with heterogeneous users that need to deliver a short packet within a strict deadline on the uplink as a Dec-POMDP.
- We consider a general framework where packets are generated according to either a probabilistic aperiodic or a probabilstic periodic traffic. Within this context, users observe multiple time-varying orthogonal sub-channels.
- We introduce two PPO solutions to solve the Dec-POMDP. The first one, MCA-PPO, is theoretically-justified and enjoys the monotonic improvement guarantee. The second one, MCA-iPPO is a fully decentralized approach, which, while lacking rigorous theoretical assurances, exhibits good empirical results and offers a simplified training procedure.
- Finally, we validate the superiority of our proposed methods on different scenarios. Our results consistently surpass traditional MA benchmarks and established off-policy DRL algorithms.

6.2 System Model

6.2.1 Network Model

We consider a network with K users communicating with a BS over N time-varying orthogonal wireless channels on the uplink. We assume that time is slotted, and that a single packet can be transmitted within the duration of one time slot. Moreover, all devices maintain slot synchronization, aligning them on a consistent start for every slot. At every slot, a user can select one or several channels to send one or several replicas of its packet along with a pilot to facilitate the decoding process.

Finally, at the end of every frame, the BS broadcasts a feedback message $\boldsymbol{\alpha} \in \{-1, 0, 1\}^N$ to the users detailing the outcomes of the transmissions on

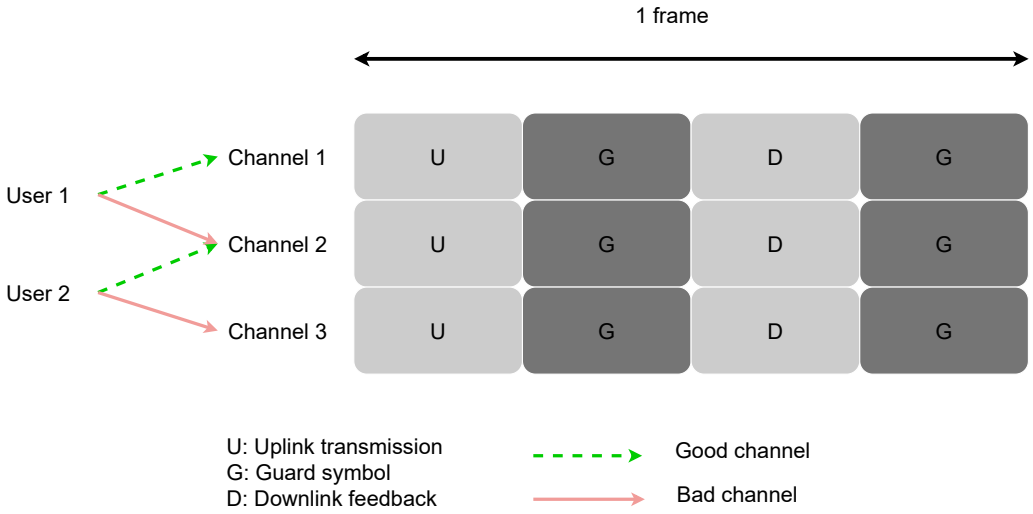


Figure 6.1: System model and slot structure

every channel. In particular, they receive an ACK ($\alpha^n = 1$) if a packet has been successfully transmitted with channel n, a NACK message ($\alpha^n = -1$) if the BS did not manage to decode the packet, and an IDLE message ($\alpha^n = 0$) to indicate that the channel was idle. Operating within a Time Division Duplexing (TDD) framework, our system is able to utilize channel reciprocity, a principle asserting that the wireless channel characteristics are symmetric, meaning that they provide identical responses in both forward and reverse communication directions [Tang et al., 2021]. By leveraging this principle, the feedback message from the BS conveys necessary information, enabling users to ascertain the state of their respective communication channels during a given frame.

6.2.2 Traffic Model

Devices initiate packet generation following either a deterministic or probabilistic traffic.

Periodic Traffic

In this model inspired by [Hou and Kumar, 2013], each device k generates packets periodically every N_p radio frames with probability ξ_k. Devices are not synchronous, and are assigned an offset parameter $\bar{f}_k \in [0, N_p]$ such that, at every radio frame $t \geq 0$, the probability for a device k of generating a new packet is: $\bar{\xi}_k(t|\bar{f}_k, \xi_k, N_p) = \mathbb{1}_{\{t[N_p]=\bar{f}_k\}}\xi_k$.

Aperiodic Traffic

This model comes from 3GPP specifications [3GPP, 2018b] and is based on the File Transfer Protocol (FTP) model 3 defined in [3GPP, 2015], but with a fixed size for each packet. At every device k, packets are generated according to a Poisson process of rate λ_k.

In order to model the strict latency constraint of URLLC networks, each user k needs to deliver its packet within an individual air interface latency constraint, δ_k; if a packet has not been transmitted before δ_k slots after its arrival in the buffer, it is discarded. If a transmission fails (due to a collision or a decoding error), the packet can be retransmitted up until its deadline is met.

We assume that the devices have unlimited buffer capacity and that the packet queue operates on a “first come, first served" principle. For any device k, we define the buffer status at a time t by the vector $\boldsymbol{b}_t^k \in \mathbb{N}^{\delta^k}$, where $b_t^{k,d} = i$ indicates that device k has i packets with a deadline duration of d at time t. The matrix of the buffers of all devices at time t is represented by $\boldsymbol{B}_t$. The buffer status of a device k transits as follows: (a) Successfully decoded packets are removed from the buffer (b) Other packets see their time-to-deadline decreased by one. Expired packets are removed from the buffers; (c) New generated packets enter the buffer with a deadline δ_k.

6.2.3 Channel Model

Every channel between a user and the BS follows the Gilbert-Elliot channel model [Gilbert, 1960]: at any given slot each channel can be in one of two states: a good channel state, ensuring successful transmission, or a bad channel state, leading to a transmission failure.

The state switching pattern is represented by a Markov chain. For each slot

t, the channel state is represented by $\boldsymbol{\eta_t} \in \{0,1\}^{K\times N}$ where $\eta_t^{k,n}$ represents the state of the n-th channel of user k in slot t. We assume that each channel state can only change at the beginning of each frame and remains constant during the frame. The channel n of a user k evolves according to the transition matrix: $\begin{pmatrix} 1-p_{k,n} & p_{k,n} \\ \tilde{p}_{k,n} & 1-\tilde{p}_{k,n} \end{pmatrix}$ where $p_{k,n}$ is the probability that the state of the n-th channel of k changes from bad to good and $\tilde{p}_{k,n}$ the probability that it switches from good to bad.

We adopt a collision channel model: when the BS receives a single packet via the good channel resource n, it can successfully decode it. When several users transmit on the same channel during the same frame, a collision occurs and no packet is delivered to the BS whatever their respective channel state.

6.3 Problem Formulation

This problem can be formulated by a Dec-POMDP [Oliehoek, 2012].

A Dec-POMDP is a cooperative MG where agents take decisions based on individual observations about the environment. In our problem, the Dec-POMDP elements are defined as follows:

- The state $\boldsymbol{s}_t \in \mathcal{S}$ is the concatenation of the current buffer status and the current channel state i.e. $\boldsymbol{s}_t = (\boldsymbol{B}_t, \boldsymbol{\eta}_t)$.
- Each user k observes $\boldsymbol{o}_t^k$, which is made of its own buffer $\boldsymbol{b}_t^k$, the last channel observation $\boldsymbol{\eta}_{t-1}^k$ and the last ACK/NACK from the BS $\boldsymbol{\alpha}_{t-1}$: $\boldsymbol{o}_t^k = (\boldsymbol{b}_t^k, \boldsymbol{\eta}_{t-1}^k, \boldsymbol{\alpha}_{t-1})$.
- At every slot t, each agent k selects an action $\boldsymbol{a}_t^k \in \{0,1\}^N$ based on its observation $\boldsymbol{o}_t^k$ and policy $\pi^k(\cdot|\boldsymbol{o}_t^k)$ where $a_{n,t}^k = 1$ if agent k transmits a packet using the channel n. We denote the global action $\boldsymbol{A}_t \in \{0,1\}^{K\times N}$, this is the concatenation of all individual actions.
- The next state of the system is drawn with the transition function $\mathcal{T} : \mathcal{S} \times \mathcal{A} \mapsto \Delta(\mathcal{S})$. This function follows the buffer and channel state's dynamics defined in Sections 6.2.2 and 6.2.3.
- Finally, we define the reward r_t at frame t as the total number of successful transmissions across all channels.

$$r_t = \sum_{n=1}^{N} 1_{(\alpha_t^n = 1)} \tag{6.1}$$

Each user k aims to optimize the following objective:

$$\mathbb{E}_{\boldsymbol{s}_{t+1} \sim \mathcal{T}, a^{-k} \sim \pi^{-k}} \left[\sum_{t=0}^{T} \gamma^t r_t | a_t^k \sim \pi^k(\cdot | \boldsymbol{o}_t^k), \boldsymbol{s}_0 \right] \tag{6.2}$$

where $\gamma \in (0, 1]$ is the discount factor that allows the agents to balance immediate rewards with future ones.

In order to compare the performance of the MA protocols for the URLLC problem, we define a *URLLC score* as the ratio between the number of packets delivered before expiration and the number of generated packets.

6.4 Multi-Agent Deep Reinforcement Learning

6.4.1 Policy Gradient for Multi-Agent Systems

TRPO algorithms [Schulman et al., 2015a], have been introduced in SARL in order to ensure the *monotonic improvement* property, which guarantees a non-decreasing performance of the policy at every iteration.

This property has been extended to MARL in [Kuba et al., 2021] thanks to the Multi-Agent Advantage Decomposition lemma:

Lemma 6.1 (Multi-Agent Advantage Decomposition [Kuba et al., 2021])**.** *In any cooperative Markov game, given a joint policy $\boldsymbol{\pi}$, for any state $\boldsymbol{s}$, and any agent subset $k_{1:m}$, the global advantage $A_{\boldsymbol{\pi}}^{k_{1:m}}(\boldsymbol{s}, \boldsymbol{a}^{k_{1:m}})$ can be decomposed into a summation of each agent's local advantages $A_{\boldsymbol{\pi}}^{k_j}$:*

$$A_{\boldsymbol{\pi}}^{k_{1:m}}(\boldsymbol{s}, \boldsymbol{a}^{k_{1:m}}) = \sum_{j=1}^{m} A_{\boldsymbol{\pi}}^{k_j}(\boldsymbol{s}, \boldsymbol{a}^{k_{1:j-1}}, a^{k_j}) \tag{6.3}$$

Lemma 6.1 gives us a methodology for the agents to update their local policies while guaranteeing monotonic improvement. Indeed, let agents take actions sequentially by following an arbitrary order $k_{1:K}$. Agent k_1 takes action $\bar{a}^{k_1}$ such that $A_{\boldsymbol{\pi}}^{k_1}(\boldsymbol{s}, \bar{a}^{k_1}) > 0$. Agent k_2 selects action $\bar{a}^{k_2}$ such that $A_{\boldsymbol{\pi}}^{k_2}(\boldsymbol{s}, \bar{a}^{k_1}, \bar{a}^{k_2}) > 0$.

For the remaining $m = 3, \dots, K$, each agent k_m selects an action $\bar{a}^{k_m}$ such that $A_{\boldsymbol{\pi}}^{k_m}(\boldsymbol{s}, \bar{a}^{k_{1:m-1}}, \bar{a}^{k_m}) > 0$. Thus, Lemma 6.1 guarantees that the global advantage $A_{\boldsymbol{\pi}}(\boldsymbol{s}, \boldsymbol{A})$ is positive and therefore the performance is guaranteed to improve.

To summarize, the work of [Kuba et al., 2021] demonstrates that the monotonic improvement property holds in Multi-Agent TRPO when each agent updates its local policy sequentially and taking into account all previous agents' updates.

6.4.2 Multi-Channel Access Proximal Policy Optimization

The PPO algorithm [Schulman et al., 2017] is a TRPO algorithm that leverages the principle of limiting the magnitude of policy updates, using first-order optimization techniques only. In a multi-agent setting where the monotonic improvement property holds, each agent k_m is equipped with a PPO algorithm and aims to maximize the following objective with respect to parameters θ^{k_m} and ϕ^{k_m}:

$$\mathbb{E}_{\boldsymbol{s},\boldsymbol{a}\sim(\pi_{\theta_{\text{old}}},\mathcal{T})}\left[\min\left(\frac{\pi_{\theta^{k_m}}^{k_m}(\boldsymbol{a}^{k_m}|\boldsymbol{o}^{k_m})}{\pi_{\text{old}}^{k_m}(\boldsymbol{a}^{k_m}|\boldsymbol{o}^{k_m})}\hat{A}_{\phi^{k_m}}, g(\nu)\hat{A}_{\phi^{k_m}}\right)\right] \tag{6.4}$$

with $g(\nu) = \text{clip}\left(\frac{\pi_{\theta^{k_m}}(\boldsymbol{a}^{k_m}|\boldsymbol{o}^{k_m})}{\pi_{\text{old}}(\boldsymbol{a}^{k_m}|\boldsymbol{o}^{k_m})}, 1-\nu, 1+\nu\right)$ and $\nu \in [0, 1)$ a hyperparameter that indicates how far away the new policy can deviate from the old one and where $\hat{A}_{\phi^{k_m}}$ is a global advantage estimator with parameters ϕ^{k_m}.

We propose two different methods to estimate the global advantage.

- **MCA-PPO**, a multi-agent PPO version where the monotonic improvement property holds. We have:

$$\hat{A}_{\phi^{k_m}} = M^{k_{1:m}} \tag{6.5}$$

  where $M^{k_{1:m}}$ is the compound policy ratio introduced by [Kuba et al., 2021]. It is a joint advantage estimator that takes into account the previous policy updates and allow us to apply Lemma 6.1.

  It is defined as follows:

$$M^{k_{1:m}} = \frac{\pi^{k_{1:m-1}}(\boldsymbol{a}^{k_{1:m-1}}|\boldsymbol{o}^{k_{1:m-1}})}{\pi_{\text{old}}^{k_{1:m-1}}(\boldsymbol{a}^{k_{1:m-1}}|\boldsymbol{o}^{k_{1:m-1}})} A_{\phi}^{\text{GAE}}(\boldsymbol{s}, \boldsymbol{A}) \tag{6.6}$$

with $A_{\phi}^{\text{GAE}}(\boldsymbol{s}, \boldsymbol{A})$ the global advantage estimate, parameterized by ϕ and computed at the BS with GAE [Schulman et al., 2015b]. Details of the algorithm are provided in Algorithm 7. Note that contrary to IL and CTDE versions, only one global advantage estimate is required for all agents.

In order to train MCA-PPO, we use the on-policy property of policy gradient algorithms. This property divides the algorithm's operation into 2 phases: an *execution phase* and a *training phase*. During the execution phase, MCA-PPO runs the current joint policy and stores trajectories while using the communication resources for data transmission to the BS. During the training phase, data transmission is stopped and the communication resources are used to share the compounded policy ratios between devices in order to update the policy. The training phase is described in Figure 6.2.

- **MCA-iPPO**. As training the DRL agents offline may be inconvenient in some cases, we propose an IL alternative, MCA-iPPO, where

$$\hat{A}_{\phi^{k_m}} = A_{\phi^{k_m}}^{\text{GAE}}(\boldsymbol{o}^{k_m}, \boldsymbol{a}^{k_m}) \tag{6.7}$$

where $A_{\phi^{k_m}}^{\text{GAE}}(\boldsymbol{o}^{k_m}, \boldsymbol{a}^{k_m})$ is the global advantage estimate of agent k_m. Training this method follows the single-agent PPO algorithm [Schulman et al., 2017] and considers the other agents as part of the environment.

Finally, in order to tackle the partially observable aspect of our problem, we use a RNN [Hausknecht and Stone, 2015] to enable agents to take actions based on their previous actions and observations as we have done in the previous chapters as well. The intuition is that the RNN's hidden states estimate a belief state over the underlying system state. We use the GRU architecture [Cho et al., 2014] for both of our MCA-PPO variants.

6.5 Simulation Results

6.5.1 Simulation Settings

The parameters of our traffic model are adopted from the factory automation use case of the 3GPP 5G NR specifications on URLLC [3GPP, 2018b] where we

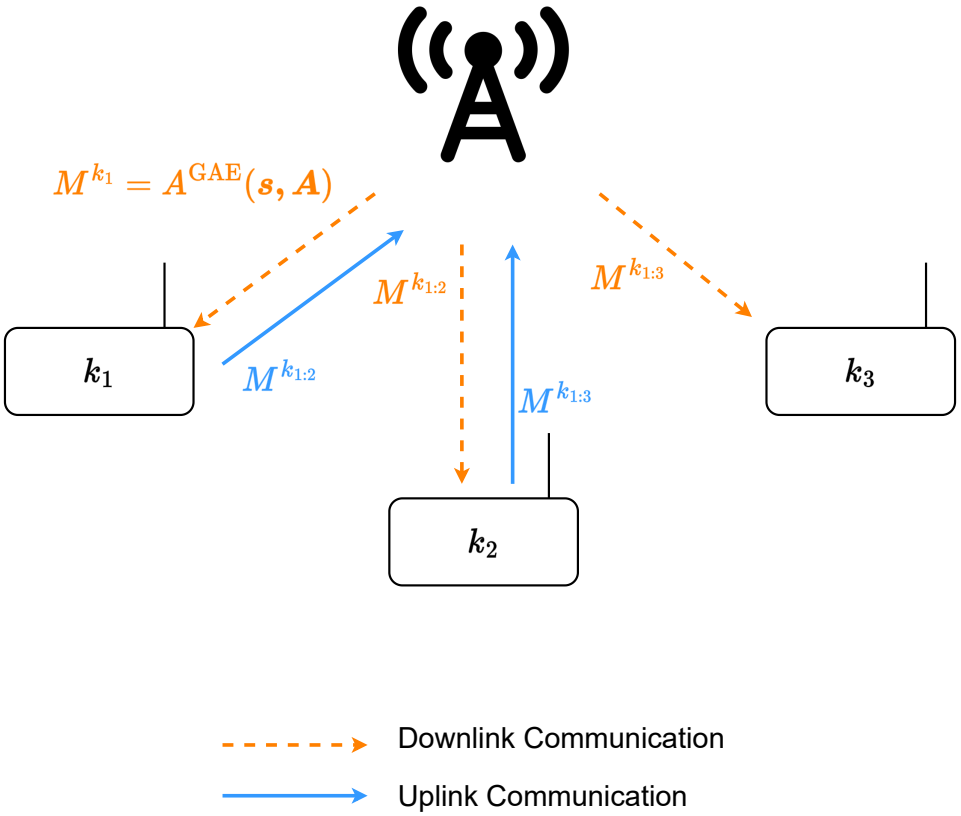


Figure 6.2: Training phase of MCA-PPO. Communication resources are used to train the agents. 1) The BS draws a permutation of the agents $k_{1:3}$, computes the global advantage function, and sends it to user k_1; 2) User k_1 updates its policy, computes $M^{k_{1:2}}$ and sends it to the BS; 3) The BS sends $M^{k_{1:2}}$ to user k_2 which updates its policy, computes $M^{k_{1:3}}$, and transmits it to the BS; 4) The BS sends $M^{k_{1:3}}$ to user k_3 that updates its policy, knowing all previous updates.

Algorithm 7: MCA-PPO

Initialize policy parameters $\theta_0^1, \dots, \theta_0^K$ for each agent and the global value function parameters ϕ

for *iteration* $i = 1, \dots, I$ **do**

  Switch the devices to *execution mode* and execute the joint policy $\boldsymbol{\pi_{\theta_i}}(\pi_{\theta_i}^1, \dots, \pi_{\theta_i}^K)$.

  Save trajectories $\{(\boldsymbol{o}_{b,t}^k, \boldsymbol{a}_{b,t}^k, \boldsymbol{o}_{b,t+1}^k, r_{b,t})\}_{b=1,\dots,\beta} \forall k \in [\![1, K]\!], \forall t \in [\![1, T]\!]$ in the buffer.

  Compute the rewards-to-go $\hat{R}_{b,t}$ for each trajectory: $\hat{R}_{b,t} = \sum_{t'=t}^{T} \gamma^{t'} r_{b,t'}$

  Switch the devices to *training mode*.

  for *epoch* $e = 1, \dots, E$ **do**

    Compute the global advantage function $A^{GAE}(\boldsymbol{s}, \boldsymbol{A})$ with V_ϕ and GAE.

    Draw a random permutation of the agents $k_{1:K}$

    Set $M^{k_1} = A^{GAE}(\boldsymbol{s}, \boldsymbol{A})$.

    for *agent* $k_m = k_1, \dots, k_K$ **do**

      Update actor k_m and derive $\theta_{i+1}^{k_m}$ by maximizing the following objective with the Adam algorithm [Kingma and Ba, 2014]

$$\max_{\theta} \frac{1}{\beta T} \left[\sum_{b=1}^{\beta} \sum_{t=1}^{T} \min \left(\frac{\pi_{\theta}^{k_m}(\boldsymbol{a}_{b,t}^{k_m} | \boldsymbol{o}_{b,t}^{k_m})}{\pi_{\theta_i^{k_m}}^{k_m}(\boldsymbol{a}_{b,t}^{k_m} | \boldsymbol{o}_{b,t}^{k_m})} M_{b,t}^{k_{1:m}}, g(\nu) M_{b,t}^{k_{1:m}} \right) \right]$$

      Compute (unless $m = K$):

$$M^{k_{1:m+1}} = \frac{\pi_{\theta_{i+1}^{k_m}}^{k_m}(\boldsymbol{a}_{b,t}^{k_m} | \boldsymbol{o}_{b,t}^{k_m})}{\pi_{\theta_i^{k_m}}^{k_m}(\boldsymbol{a}_{b,t}^{k_m} | \boldsymbol{o}_{b,t}^{k_m})} M^{k_{1:m}}$$

    Update the global value network by minimizing the mean-squared error with the Adam algorithm:

$$\phi_{i+1} = \arg\min_{\varphi} \frac{1}{\beta T} \sum_{b=1}^{\beta} \sum_{t=1}^{T} \left(V_\phi(\boldsymbol{s}_{b,t}) - \hat{R}_b(t) \right)^2 \tag{6.8}$$

consider deadlines of 1ms and an inter-arrival time of 2ms. Every slot is made of 1 OFDM symbol for a subcarrier spacing of 30kHz so that its time duration is equal to $T_s = 35.67\mu$s [3GPP, 2017e]. We consider the two following settings for our experiments:

- An *homogeneous setting* where all users' traffic is aperiodic with the same rate λ and the same deadlines δ. Given that our radio frame is made of four time-slots, we can express realistic values of λ and δ in terms of number of frames i.e. $\lambda = 1/14 = 0.07$ packet per user and per frame and $\delta = 7$ frames. The channel switch probabilities $p_{k,n}$ and $\tilde{p}_{k,n}$ are equal to 0.8.

- An *heterogeneous setting* with 6 users and 16 channels. Devices have a deadline chosen uniformly in the subset $\{1\text{ms}, 2\text{ms}\}$. Half of them have a periodic traffic with arrival probabilities chosen uniformly in $\{0.2, 0.4, 0.6, 0.8\}$ and no offset. The other half an aperiodic one. The channel switch probabilities $p_{k,n}$ and $\tilde{p}_{k,n}$ are equal and chosen uniformly in $\{0.2, 0.4, 0.6, 0.8\}$. The parameters of the learning algorithms are given in Table 6.1.

Table 6.1: Parameters of the learning algorithms

Parameter	Value	Algorithm
Discount factor (γ)	0.4	All
E	5	MCA-PPO, MCA-iPPO
I	2000	MCA-PPO, MCA-iPPO
Learning rate policy	$3 \cdot 10^{-4}$	MCA-PPO, MCA-iPPO
Learning rate critic	10^{-3}	MCA-PPO, MCA-iPPO
Batch size	64	All
History length	K	All
Update target frequency	100	iDRQN
Episode length (T)	200 slots	All
Final exploration rate	0.1	iDRQN

6.5.2 Baselines

In order to assess the performance of our algorithms, we introduce the two traditional baselines:

- **Contention-based grant-free access (GF access)**: All devices with a packet to transmit can simultaneously access multiple channels. Each channel is accessed with the same probability p. This access probability is empirically optimized for every user at every experiment in order to maximize the URLLC score. We employ a reactive scheme: when a device receives a NACK feedback from the BS, it will re-transmit the same packet with probability p until an ACK feedback is received or the time to deadline of the packet expires.

- **Independent Deep Recurrent Q-Networks (iDRQN)**: This baseline represents a widely-adopted DRL algorithm where each agent is modeled by a Deep Q-network and selects what channel to access through to a RNN specifically composed of a GRU layer. This standard approach has been previously employed in [Ye et al., 2021] for example.

6.5.3 Study of the Subchannel Assumption

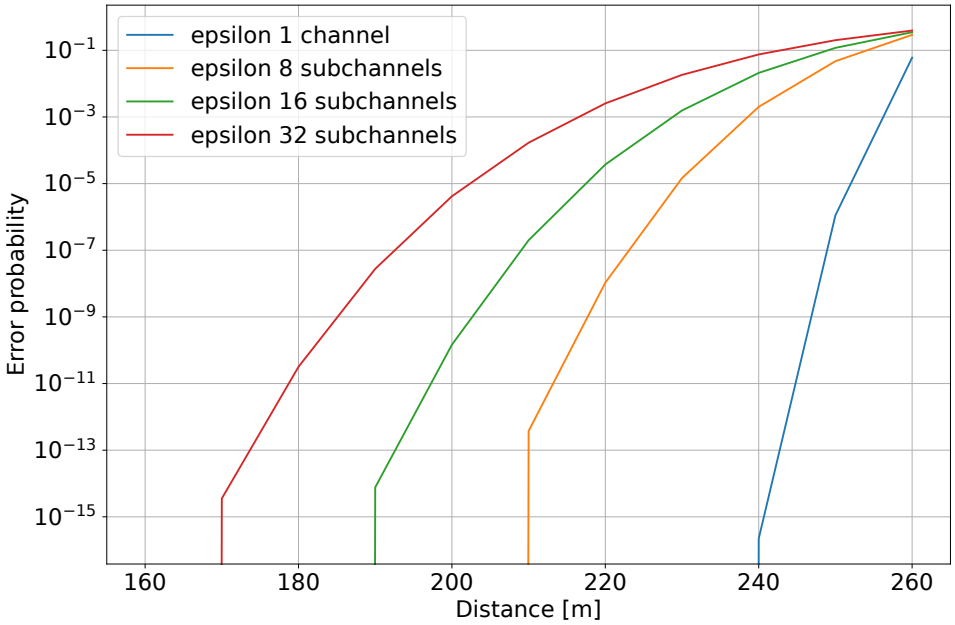


Figure 6.3: Packet Error Probability as a Function of the Distance to the BS.

First of all, we test the assumption regarding the division of the bandwidth

in several orthogonal subchannels for URLLC scenarios to confirm its feasibility and realism. This study is depicted in Figure 6.3 where we analyze the evolution of the packet error probability as a function to the distance to the BS under the 3GPP scenario defined in Chapter 4. The specific parameters employed for this figure can be found in Table. 4.3. We compute the error probability for 1, 8, 16 and 32 subchannels. We observe that for the factory automation scenarios [3GPP, 2018b] where devices are located within 200m of the BS, the error probability remains below 10^{-5} for 32 orthogonal subchannels aligning to the URLLC requirements. This confirms our hypothesis and our choice to split the bandwidth for URLLC applications.

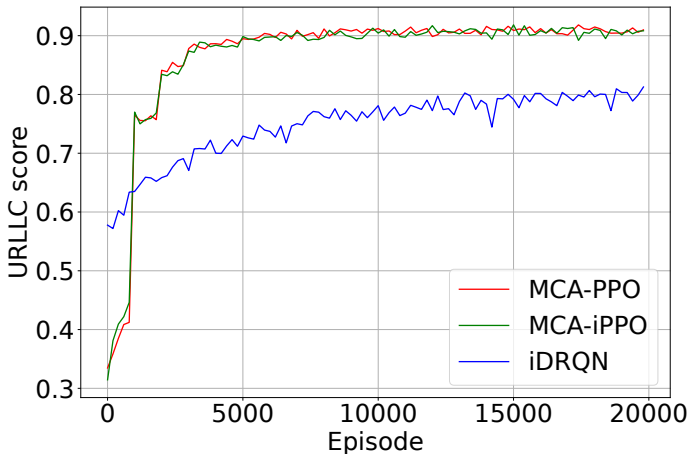
6.5.4 Convergence Speed of the Algorithms

We show in Figure 6.4a the evolution of the URLLC score during the training of 16 DRL agents under the homogeneous setting. First, we can see that both PPO approaches have similar training behavior and reach the same optimum. They both outperform the Q-learning approach in terms of convergence speed and asymptotic URLLC score.

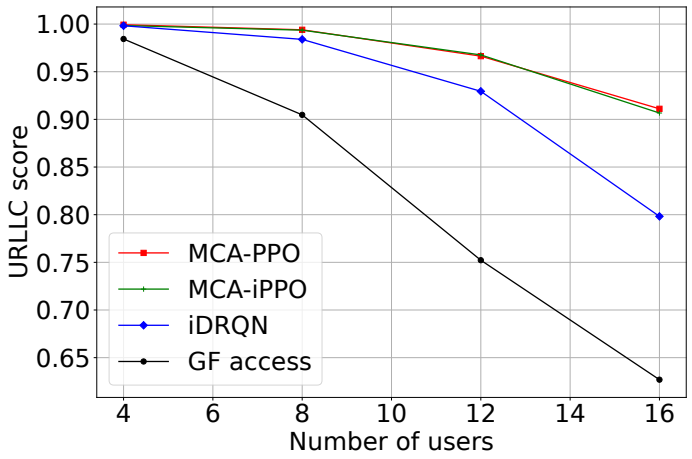
6.5.5 URLLC Score

On the one hand, in Figure 6.4b, we investigate the performance of our proposed methods within a homogeneous environment, varying the number of users from 4 to 16. On the other hand, Figure 6.4c examines the influence of the load per frame on the URLLC score.

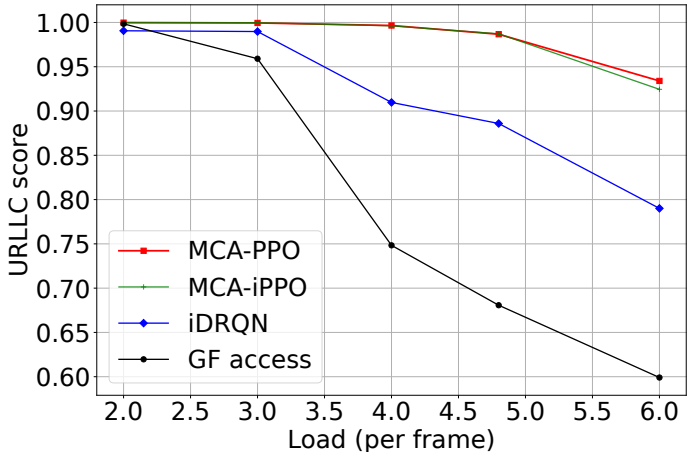
First, we can see in both scenarios that MCA-iPPO reaches the same performance as MCA-PPO in spite of its theoretical shortcomings. The surprising performance of iPPO has been highlighted in the MARL literature [de Witt et al., 2020]. Second, we can see that while iDRQN outperforms the GF access algorithm, it remains inferior to the methodologies proposed in this work. Finally, we observe in the heterogenous setting, that the gap between our methods and the GF access baseline increases as we increase the load. Indeed, while the GF access algorithm treats all agents homogeneously, MCA-PPO and MCA-iPPO manage to exploit the heterogeneity and the diversity in the users' characteristics and learn a better transmission protocol.



(a) Training of 16 homogeneous users.



(b) Homogeneous setting.



(c) Heterogeneous setting.

Figure 6.4: Evolution of the URLLC score as a function of (a) the number of training episodes; (b) the number of users; (c) the load per frame.

6.6 Conclusion

This work introduces a novel approach to the DMCA problem in URLLC networks, a challenge pertinent to various IoT applications. The DMCA problem is formulated as a Dec-POMDP accommodating devices with heterogeneous traffic models and system parameters. We introduce two PPO algorithms, namely MCA-PPO and MCA-iPPO, to handle the complexities of the URLLC environment with strict deadlines. While the MCA-PPO is theoretically robust and benefits from the monotonic improvement property, it requires offline training. On the contrary, MCA-iPPO, in spite of its theoretical limitations, employs a more straightforward training process and empirically achieves a performance comparable to MCA-PPO. Our proposed algorithms have been validated in both homogeneous and heterogeneous scenarios, consistently outperforming the traditional GF access baseline and iDRQN algorithm. Further work may explore more complex channel models using SIC for example.

In addition to the advancements in DMCA for URLLC networks, the MCA-PPO approach is also promising for a broad spectrum of applications regarding MA. In particular, it presents significant potential in the field of multi-connectivity where each device has the capability to send replicas of its packet to multiple BS [Segura et al., 2022]. Furthermore, our methodology can be adeptly applied to Modern Random Access strategies, where devices transmit packet replicas across several time slots of a single frame [Berioli et al., 2016]. This scenario can essentially be conceptualized as a DMCA problem, but within the time domain.

Chapter 7

Conclusion and Future Work

Contents

7.1 Conclusion

This thesis explores the application of DRL, both from a single-agent and a multi-agent perspective to develop efficient transmission protocols within an industrial IoT environment, under the stringent constraints of URLLC. Indeed, traditional MA protocols struggle with several significant challenges when dealing with the URLLC requirements. On the one hand, centralized protocols, despite their effectiveness in device coordination and avoiding collisions, are hindered by significant communication overhead and the resultant latency from the coordination process. On the other hand, GF protocols, although promising alternatives for URLLC due to their ability to bypass the latency of traditional four-way handshake protocols, face collisions. Additionally, the current protocols are not tailored to handle the diversity in device and the sporadic nature of the traffic in dynamic environments. This thesis aims to address these challenges through the innovative application of DRL strategies.

Our research initially focuses on a framework modeling the uplink MA problem as a centralized problem where the BS schedules devices in order to prevent collisions. To minimize latency, we eliminate the usual coordination communications between devices and the BS, a move that introduced partial observability into the system. We tackle this problem by developing FilteredPPO, a novel scheduling algorithm. By integrating PPO with RNN and invalid action masking, FilteredPPO demonstrates superior performance over conventional benchmarks in scenarios with periodic traffic.

As FilteredPPO struggles to meet the URLLC requirements under aperiodic traffic, we extend our scheduling framework by incorporating the NOMA technology, allowing the BS to poll multiple users within a single frame. We introduce the concept of *agent state*, to better manage partial observability and develop NOMA-PPO, a DRL algorithm that efficiently deals with the combinatorial action space using a branching policy network and can incorporate prior knowledge about the system evolution through a Bayesian prior. Our experiments, conducted under realistic 3GPP conditions confirm the effectiveness of NOMA-PPO in terms of URLLC performance, fairness, and convergence speed, outperforming traditional MA and DRL benchmarks across various scenarios.

We then move to the decentralized version of our MA problem and explore the application of deep MARL to tackle it, leading to the creation of SeqDQN. This

distributed MARL algorithm updates Q-functions sequentially, starting with devices having the strictest latency requirements. SeqDQN presents advantages in scalability and training speed over traditional MARL approaches, and reduces non-stationarity issues.

Finally, we extend the decentralized framework by efficiently utilizing the bandwidth through orthogonal channel division. This transforms our URLLC problem into a DMCA problem with heterogeneous devices. We solve it by proposing two PPO solutions: MCA-PPO and MCA-iPPO. While MCA-PPO offers a theoretically grounded approach with monotonic improvement guarantees, MCA-iPPO provides a decentralized, empirically effective alternative with a simpler training process. We show that our approach outperforms the existing MA and DRL benchmarks on different scenarios.

7.2 Future Work

In this section, we explore potential directions for future research related to DRL-based protocol for uplink access in URLLC networks. These perspectives can be classified in two categories: short-term directions that increment the foundations laid in this thesis, and long-term objectives addressing more fundamental scientific challenges associated with the application of DRL to uplink URLLC.

7.2.1 Short-Term Research Directions

- First of all, future work could aim to apply and test the algorithms developed in this thesis within more realistic and complex settings. For instance these algorithms could be evaluated using a network digital twin, similar to the work of [Vilà et al., 2023] that use this simulator to train a DRL solution for the radio access network problem. Additionally, incorporating more sophisticated environmental assumptions, such as complex scattering or advanced mobility models for users can also be an alternative to complexify the system and enhance the realism and applicability of the research.

- Another direct extension of this thesis would be to incorporate power control mechanisms for the devices. The integration of power control aligns

naturally with the decentralized approach envisioned in our work, where the action of each device could be defined in terms of transmit power, chosen from either a continuous or discrete set of values. This concept can also be adapted to fit within the centralized framework. With the assumed frame structure, the BS has indeed the possibility to control this power by including in the polling message a transmit power control command for the device. Importantly, the duration of our frame is designed to be shorter than the coherence time, thus enabling effective and timely power control even in the context of time-varying channels. While the application of DRL to power optimization has already been investigated ([Tan et al., 2020, Zhang et al., 2020, Zhang and Liang, 2020]), its specific application and implications in the context of URLLC networks remain an open area for exploration.

- In Chapter 6, we mentioned that MCA-PPO algorithms presented in this thesis could be directly applied to the Modern Random Access problem. This particular problem involves selecting a subset of time slots within a frame for a device to transmit replicas of its packet. The main difference in this context lies in the decoding procedure, which is essential for resolving collisions that may occur within any given time slot.

- Finally, an intuitive direction for future research involves expanding the scope of NOMA-PPO to a multi-agent setting. While we have successfully demonstrated the efficiency of our centralized solution involving a single BS, exploring the dynamics of multiple BSs, each employing the NOMA-PPO algorithm, is a promising way to improve the URLLC capacity. This investigation would focus on how these BSs could interact and coordinate their scheduling decisions to effectively mitigate inter-cell interference and enhance overall URLLC performance.

7.2.2 Long-Term Research objectives

- This thesis has investigated the application of MARL from the IL and CTDE perspectives in order to learn fully decentralized policies. The third popular framework that we have not explored is the *networked agents* paradigm that allows agents within a short range to communicate and ex-

change information. This assumption can be realistic as IoT devices are able to communicate with neighbors within a limited communication range as highlighted in [Park and Saad, 2019]. A key enabler for such localized communication among IoT devices is Device-to-Device (D2D) communications. The potential of this technology in fulfilling IoT requirements for 5G networks, particularly in mMTC and URLLC, has been underscored in various studies [Militano et al., 2015, Chang et al., 2021, MB et al., 2020]. In the MARL context, D2D communications could facilitate the sharing of local states, actions and rewards, within a neighborhood before accessing the communication channel. This would allow the application of the networked agents' framework, which has significantly stronger theoretical foundations compared to IL and CTDE. Several works, such as those by Lin et al. and Yang et al. [Lin et al., 2020, Yang et al., 2020], have already demonstrated the benefits of leveraging information exchange in wireless networks for MARL applications. However, D2D communications bring their own set of challenges such as devices discovery, interference management or mode selection [Asadi et al., 2014]. Therefore, combining the networked agents MARL framework with the D2D communication technology is a promising and interesting direction for future research.

- A second interesting framework yet to be explored in this thesis, which shows considerable promise for addressing the MA problem is the *mean-field regime*. In this regime, each individual agent in a vast population is influenced by a cumulative measure derived from all other agents. This approach is utilized both for evaluating the performance of established protocols, as seen in studies on ALOHA and CSMA [Bordenave et al., 2008, Duffy, 2010], and in the context of *mean field games* [Cousin et al., 2011], which are employed to develop control policies for individual devices. The latter has been applied to multiple access in wireless systems [Huang et al., 2003, Bertucci et al., 2018], particularly in scenarios where system dynamics are known. Mean-field regimes have started to be investigated in relation to RL [Subramanian and Mahajan, 2019, Elie et al., 2020, Yang et al., 2018]. Learning in mean field models is especially adapted in environments with a massive number of devices. While this area remains largely unexplored in the context of wireless communications, it holds sig-

nificant potential. Future work could focus on mean field models not only in deriving learning policies but also in providing analytical insights and guarantees about system behavior in scenarios with high density of devices, in mMTC scenarios for example.

- A final promising framework for long-term future work is multi-connectivity, which holds significant potential for uplink URLLC. This technology is particularly beneficial in enhancing coverage, especially in environments with obstacles and in scenarios involving user mobility. Multi-connectivity can also substantially improve both reliability and latency by enabling devices to transmit duplicate data to multiple BS. However, it presents certain challenges. One major challenge is coordinating user transmissions across different BS, ensuring efficient and collision-free communication. Besides, mitigating interference in this context is a complex issue, especially when D2D communications are integrated into the system, adding an additional layer of complexity. In addressing these challenges, RL emerges as a promising solution, offering potential strategies to effectively manage the intricate dynamics of multi-connectivity in uplink URLLC scenarios.

Bibliography

[3GPP, 2015] 3GPP (2015). Feasibility Study on Licensed-Assisted Access to Unlicensed Spectrum. TR 36.889, 3rd Generation Partnership Project (3GPP). (Cited on pages 66 and 110.)

[3GPP, 2016] 3GPP (2016). Study on scenarios and requirements for next generation access technologies. TR 38.913, 3rd Generation Partnership Project (3GPP). (Cited on page 1.)

[3GPP, 2017a] 3GPP (2017a). NR; Base Station (BS) radio transmission and reception. TS 38.104, 3rd Generation Partnership Project (3GPP). (Cited on pages 7 and 80.)

[3GPP, 2017b] 3GPP (2017b). NR; Medium Access Control (MAC) protocol specification. TS 38.321, 3rd Generation Partnership Project (3GPP). (Cited on pages 4, 5 and 80.)

[3GPP, 2017c] 3GPP (2017c). NR; Multiplexing and channel coding. TS 38.212, 3rd Generation Partnership Project (3GPP). (Cited on page 80.)

[3GPP, 2017d] 3GPP (2017d). NR; Packet Data Convergence Protocol (PDCP) specification. TS 38.323, 3rd Generation Partnership Project (3GPP). (Cited on page 80.)

[3GPP, 2017e] 3GPP (2017e). NR; Physical channels and modulation. TS 38.211, 3rd Generation Partnership Project (3GPP). (Cited on pages 80 and 117.)

[3GPP, 2017f] 3GPP (2017f). NR; Radio Link Control (RLC) protocol specification. TS 38.322, 3rd Generation Partnership Project (3GPP). (Cited on page 80.)

[3GPP, 2017g] 3GPP (2017g). Study on channel model for frequencies from 0.5 to 100 GHz. TR 38.901, 3rd Generation Partnership Project (3GPP). (Cited on pages 79 and 106.)

[3GPP, 2017h] 3GPP (2017h). Technical Specification Group Services and System Aspects; Study on Communication for Automation in Vertical domains (CAV). TS 22.804, 3rd Generation Partnership Project (3GPP). (Cited on pages 66 and 67.)

[3GPP, 2018a] 3GPP (2018a). Study on NR industrial Internet of Things (IoT). TR 38.825, 3rd Generation Partnership Project (3GPP). (Cited on page 77.)

[3GPP, 2018b] 3GPP (2018b). Study on physical layer enhancements for NR ultra-reliable and low latency case (URLLC). TR 38.824, 3rd Generation Partnership Project (3GPP). (Cited on pages 7, 10, 12, 64, 66, 77, 79, 80, 95, 96, 99, 110, 114 and 119.)

[3GPP, 2020] 3GPP (2020). Evolved Universal Terrestrial Radio Access (E-UTRA) and NR; Service Data Adaptation Protocol (SDAP) specification. TS 37.324, 3rd Generation Partnership Project (3GPP). (Cited on page 80.)

[Abreu et al., 2018] Abreu, R., Berardinelli, G., Jacobsen, T., Pedersen, K., and Mogensen, P. (2018). A blind retransmission scheme for ultra-reliable and low latency communications. In *2018 IEEE 87th Vehicular Technology Conference (VTC Spring)*, pages 1–5. IEEE. (Cited on page 5.)

[Ahsan et al., 2022] Ahsan, W., Yi, W., Liu, Y., and Nallanathan, A. (2022). A reliable reinforcement learning for resource allocation in uplink noma-urllc networks. *arXiv preprint arXiv:2201.06027*. (Cited on pages 9 and 56.)

[Al-Garadi et al., 2020] Al-Garadi, M. A. et al. (2020). A survey of machine and deep learning methods for internet of things (IoT) security. *IEEE Commun. Surveys Tuts.*, 22(3):1646–1685. (Cited on page 8.)

[Amari, 1993] Amari, S.-i. (1993). Backpropagation and stochastic gradient descent method. *Neurocomputing*, 5(4-5):185–196. (Cited on page 15.)

[Arslan and Yüksel, 2017] Arslan, G. and Yüksel, S. (2017). Decentralized Q-Learning for Stochastic Teams and Games. *IEEE Trans. Autom. Control*, 62(4):1545–1558. (Cited on pages 92 and 97.)

[Asadi et al., 2014] Asadi, A., Wang, Q., and Mancuso, V. (2014). A survey on device-to-device communication in cellular networks. *IEEE Communications Surveys & Tutorials*, 16(4):1801–1819. (Cited on page 127.)

[Ayoub et al., 2021] Ayoub, I., Hmedoush, I., Adjih, C., Khawam, K., and Lahoud, S. (2021). Deep-irsa: A deep reinforcement learning approach to irregular repetition slotted aloha. In *2021 10th IFIP International Conference on Performance Evaluation and Modeling in Wireless and Wired Networks (PEMWN)*, pages 1–6. IEEE. (Cited on page 9.)

[Bennis et al., 2018] Bennis, M., Debbah, M., and Poor, H. V. (2018). Ultrareliable and low-latency wireless communication: Tail, risk, and scale. *Proceedings of the IEEE*, 106(10):1834–1853. (Cited on page 4.)

[Berioli et al., 2016] Berioli, M., Cocco, G., Liva, G., Munari, A., et al. (2016). Modern random access protocols. *Foundations and Trends® in Networking*, 10(4):317–446. (Cited on pages 6 and 121.)

[Bertsekas, 2012] Bertsekas, D. (2012). *Dynamic programming and optimal control: Volume I*, volume 1. Athena scientific. (Cited on page 21.)

[Bertucci et al., 2018] Bertucci, C., Vassilaras, S., Lasry, J.-M., Paschos, G. S., Debbah, M., and Lions, P.-L. (2018). Transmit strategies for massive machine-type communications based on mean field games. In *2018 15th International Symposium on Wireless Communication Systems (ISWCS)*, pages 1–5. IEEE. (Cited on page 127.)

[Bi et al., 2015] Bi, S., Zhang, R., Ding, Z., and Cui, S. (2015). Wireless communications in the era of big data. *IEEE communications magazine*, 53(10):190–199. (Cited on page 40.)

[Bianchi, 2000] Bianchi, G. (2000). Performance analysis of the ieee 802.11 distributed coordination function. *IEEE Journal on selected areas in communications*, 18(3):535–547. (Cited on page 3.)

[Biglieri et al., 2007] Biglieri, E., Calderbank, R., Constantinides, A., Goldsmith, A., Paulraj, A., and Poor, H. V. (2007). *MIMO wireless communications*. Cambridge university press. (Cited on page 7.)

[Bockelmann et al., 2018] Bockelmann, C., Pratas, N. K., Wunder, G., Saur, S., Navarro, M., Gregoratti, D., Vivier, G., De Carvalho, E., Ji, Y., Stefanović, Č., et al. (2018). Towards massive connectivity support for scalable mmtc communications in 5g networks. *IEEE access*, 6:28969–28992. (Cited on page 1.)

[Bordenave et al., 2008] Bordenave, C., McDonald, D., and Proutiere, A. (2008). Performance of random medium access control, an asymptotic approach. *ACM SIGMETRICS Performance Evaluation Review*, 36(1):1–12. (Cited on page 127.)

[Brown et al., 2018] Brown, G. et al. (2018). Ultra-reliable low-latency 5g for industrial automation. *Technol. Rep. Qualcomm*, 2:52065394. (Cited on page 1.)

[Casini et al., 2007] Casini, E., De Gaudenzi, R., and Herrero, O. D. R. (2007). Contention resolution diversity slotted aloha (crdsa): An enhanced random access schemefor satellite access packet networks. *IEEE transactions on wireless communications*, 6(4):1408–1419. (Cited on page 6.)

[Chang et al., 2021] Chang, B., Li, L., Zhao, G., Chen, Z., and Imran, M. A. (2021). Autonomous d2d transmission scheme in urllc for real-time wireless control systems. *IEEE Transactions on Communications*, 69(8):5546–5558. (Cited on page 127.)

[Chang et al., 2018] Chang, H.-H. et al. (2018). Distributive dynamic spectrum access through deep reinforcement learning: A reservoir computing-based approach. *IEEE Internet Things J.*, 6(2):1938–1948. (Cited on pages 9, 70 and 92.)

[Chen et al., 2018] Chen, H. et al. (2018). Ultra-reliable low latency cellular networks: Use cases, challenges and approaches. *IEEE Commun. Mag.*, 56(12):119–125. (Cited on pages 1 and 2.)

[Chen et al., 2021] Chen, W., Qiu, X., Cai, T., Dai, H.-N., Zheng, Z., and Zhang, Y. (2021). Deep reinforcement learning for internet of things: A comprehensive survey. *IEEE Communications Surveys & Tutorials*, 23(3):1659–1692. (Cited on page 56.)

[Cho et al., 2014] Cho, K., van Merrienboer, B., Bahdanau, D., and Bengio, Y. (2014). On the properties of neural machine translation: Encoder-decoder approaches. *CoRR*, abs/1409.1259. (Cited on pages 97 and 114.)

[Chung et al., 2014] Chung, J., Gulcehre, C., Cho, K., and Bengio, Y. (2014). Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*. (Cited on pages 17, 18, 82 and 88.)

[Cousin et al., 2011] Cousin, A., Crépey, S., Guéant, O., Hobson, D., Jeanblanc, M., Lasry, J.-M., Laurent, J.-P., Lions, P.-L., Tankov, P., Guéant, O., et al. (2011). Mean field games and applications. *Paris-Princeton lectures on mathematical finance 2010*, pages 205–266. (Cited on page 127.)

[Cuozzo et al., 2022] Cuozzo, G., Cavallero, S., Pase, F., Giordani, M., Eichinger, J., Buratti, C., Verdone, R., and Zorzi, M. (2022). Enabling urllc in 5g nr iiot networks: A full-stack end-to-end analysis. In *2022 Joint European Conference on Networks and Communications & 6G Summit (EuCNC/6G Summit)*, pages 333–338. (Cited on page 4.)

[de Witt et al., 2020] de Witt, C. S., Gupta, T., Makoviichuk, D., Makoviychuk, V., Torr, P. H., Sun, M., and Whiteson, S. (2020). Is independent learning all you need in the starcraft multi-agent challenge? *arXiv preprint arXiv:2011.09533*. (Cited on pages 34, 107 and 119.)

[Deering and Hinden, 2017] Deering, S. and Hinden, R. (2017). Internet Protocol, Version 6 (IPv6) Specification. RFC RFC 8200, Internet Engineering Task Force (IETF). (Cited on page 80.)

[Dileep, 2020] Dileep, G. (2020). A survey on smart grid technologies and applications. *Renewable energy*, 146:2589–2625. (Cited on page 1.)

[Ding et al., 2021] Ding, J., Nemati, M., Pokhrel, S. R., Park, O.-S., Choi, J., and Adachi, F. (2021). Enabling grant-free urllc: An overview of principle and enhancements by massive mimo. *IEEE Internet of Things Journal*, 9(1):384–400. (Cited on page 7.)

[Duan et al., 2016] Duan, Y., Chen, X., Houthooft, R., Schulman, J., and Abbeel, P. (2016). Benchmarking deep reinforcement learning for continuous

control. In *International conference on machine learning*, pages 1329–1338. PMLR. (Cited on page 107.)

[Duffy, 2010] Duffy, K. R. (2010). Mean field markov models of wireless local area networks. *Markov Processes and Related Fields*, 16(2):295–328. (Cited on page 127.)

[Dulac-Arnold et al., 2015] Dulac-Arnold, G., Evans, R., van Hasselt, H., Sunehag, P., Lillicrap, T., Hunt, J., Mann, T., Weber, T., Degris, T., and Coppin, B. (2015). Deep reinforcement learning in large discrete action spaces. *arXiv preprint arXiv:1512.07679*. (Cited on page 57.)

[Elayoubi et al., 2019] Elayoubi, S. E. et al. (2019). Radio resource allocation and retransmission schemes for urllc over 5g networks. *IEEE Journal on Selected Areas in Communications*, 37(4):896–904. (Cited on page 5.)

[Elie et al., 2020] Elie, R., Perolat, J., Laurière, M., Geist, M., and Pietquin, O. (2020). On the convergence of model free learning in mean field games. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 7143–7150. (Cited on page 127.)

[Feng et al., 2019] Feng, Y., Nirmalathas, A., and Wong, E. (2019). A predictive semi-persistent scheduling scheme for low-latency applications in lte and nr networks. In *ICC 2019 - 2019 IEEE International Conference on Communications (ICC)*, pages 1–6. (Cited on page 5.)

[Foerster et al., 2016] Foerster, J., Assael, I. A., De Freitas, N., and Whiteson, S. (2016). Learning to communicate with deep multi-agent reinforcement learning. *Advances in neural information processing systems*, 29. (Cited on page 37.)

[Gilbert, 1960] Gilbert, E. N. (1960). Capacity of a burst-noise channel. *Bell system technical journal*, 39(5):1253–1265. (Cited on page 110.)

[Goldsmith, 2005] Goldsmith, A. (2005). *Wireless communications*. Cambridge university press. (Cited on pages 63 and 65.)

[Goodfellow et al., 2016] Goodfellow, I., Bengio, Y., and Courville, A. (2016). *Deep learning*. MIT press. (Cited on page 15.)

[Guo et al., 2022] Guo, Z., Chen, Z., Liu, P., Luo, J., Yang, X., and Sun, X. (2022). Multi-agent reinforcement learning-based distributed channel access for next generation wireless networks. *IEEE Journal on Selected Areas in Communications*, 40(5):1587–1599. (Cited on page 70.)

[Haarnoja et al., 2018] Haarnoja, T., Zhou, A., Abbeel, P., and Levine, S. (2018). Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International conference on machine learning*, pages 1861–1870. PMLR. (Cited on page 30.)

[Hägerling et al., 2014] Hägerling, C., Ide, C., and Wietfeld, C. (2014). Coverage and capacity analysis of wireless m2m technologies for smart distribution grid services. In *2014 IEEE International Conference on Smart Grid Communications (SmartGridComm)*, pages 368–373. IEEE. (Cited on page 2.)

[Hanzo et al., 2005] Hanzo, L., Choi, B., Keller, T., et al. (2005). *OFDM and MC-CDMA for broadband multi-user communications, WLANs and broadcasting*. John Wiley & Sons. (Cited on page 2.)

[Hausknecht and Stone, 2015] Hausknecht, M. and Stone, P. (2015). Deep Recurrent Q-learning for Partially Observable MDPs. In *AAAI Fall Symposeum*. (Cited on pages 41, 46, 58, 82, 83, 97 and 114.)

[Hochreiter and Schmidhuber, 1997] Hochreiter, S. and Schmidhuber, J. (1997). Long short-term memory. *Neural computation*, 9(8):1735–1780. (Cited on pages 17 and 46.)

[Hou and Kumar, 2013] Hou, I.-H. and Kumar, P. R. (2013). Packets with deadlines: A framework for real-time wireless networks. *Synth. Lect. Commun*, 6(1):1–116. (Cited on pages 10, 41, 66, 95 and 110.)

[Hribar et al., 2019] Hribar, J., Marinescu, A., Ropokis, G. A., and DaSilva, L. A. (2019). Using deep q-learning to prolong the lifetime of correlated internet of things devices. In *2019 IEEE Int. Conf. Commun. Workshops ICC Workshops 2019*, pages 1–6. IEEE. (Cited on page 40.)

[Huang et al., 2003] Huang, M., Caines, P. E., and Malhamé, R. P. (2003). Individual and mass behaviour in large population stochastic wireless power con-

trol problems: centralized and nash equilibrium solutions. In *42nd IEEE International Conference on Decision and Control (IEEE Cat. No. 03CH37475)*, volume 1, pages 98–103. IEEE. (Cited on page 127.)

[Huang and Ontañón, 2020] Huang, S. and Ontañón, S. (2020). A closer look at invalid action masking in policy gradient algorithms. *arXiv preprint arXiv:2006.14171*. (Cited on pages 11, 41, 46 and 47.)

[Igl et al., 2018] Igl, M., Zintgraf, L., Le, T. A., Wood, F., and Whiteson, S. (2018). Deep variational reinforcement learning for pomdps. In *International Conference on Machine Learning*, pages 2117–2126. PMLR. (Cited on page 58.)

[Jakes and Cox, 1994] Jakes, W. C. and Cox, D. C. (1994). *Microwave mobile communications*. Wiley-IEEE press. (Cited on pages 62 and 63.)

[Jiang et al., 2018] Jiang, J., Dun, C., Huang, T., and Lu, Z. (2018). Graph convolutional reinforcement learning. *arXiv preprint arXiv:1810.09202*. (Cited on page 36.)

[Kaelbling et al., 1998] Kaelbling, L. P., Littman, M. L., and Cassandra, A. R. (1998). Planning and acting in partially observable stochastic domains. *Artificial intelligence*, 101(1-2):99–134. (Cited on pages 21 and 58.)

[Karlik and Olgac, 2011] Karlik, B. and Olgac, A. V. (2011). Performance analysis of various activation functions in generalized mlp architectures of neural networks. *International Journal of Artificial Intelligence and Expert Systems*, 1(4):111–122. (Cited on page 17.)

[Kassab et al., 2020] Kassab, R. et al. (2020). Multi-agent deep stochastic policy gradient for event based dynamic spectrum access. In *IEEE PIMRC*. (Cited on pages 9 and 93.)

[Kesava and Mehta, 2022] Kesava, G. S. and Mehta, N. B. (2022). Multi-connectivity for urllc and coexistence with embb in time-varying and frequency-selective fading channels. *IEEE Transactions on Wireless Communications*. (Cited on page 7.)

[Kingma and Ba, 2014] Kingma, D. P. and Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*. (Cited on pages 79 and 116.)

[Kobayashi and Caire, 2007] Kobayashi, M. and Caire, G. (2007). Joint beamforming and scheduling for a multi-antenna downlink with imperfect transmitter channel knowledge. *IEEE Journal on Selected Areas in Communications*, 25(7):1468–1477. (Cited on pages 62 and 63.)

[Konda and Tsitsiklis, 1999] Konda, V. and Tsitsiklis, J. (1999). Actor-critic algorithms. *Advances in neural information processing systems*, 12. (Cited on page 30.)

[Kuba et al., 2021] Kuba, J. G., Chen, R., Wen, M., Wen, Y., Sun, F., Wang, J., and Yang, Y. (2021). Trust region policy optimisation in multi-agent reinforcement learning. *arXiv preprint arXiv:2109.11251*. (Cited on pages 107, 112 and 113.)

[Kullback and Leibler, 1951] Kullback, S. and Leibler, R. A. (1951). On information and sufficiency. *The annals of mathematical statistics*, 22(1):79–86. (Cited on page 28.)

[Lillicrap et al., 2015] Lillicrap, T. P., Hunt, J. J., Pritzel, A., Heess, N., Erez, T., Tassa, Y., Silver, D., and Wierstra, D. (2015). Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971*. (Cited on page 57.)

[Lin et al., 2020] Lin, Y., Qu, G., Huang, L., and Wierman, A. (2020). Distributed reinforcement learning in multi-agent networked systems. *arXiv*. (Cited on pages 9 and 127.)

[Littman, 1994] Littman, M. L. (1994). Markov games as a framework for multi-agent reinforcement learning. In *Machine learning proceedings 1994*, pages 157–163. Elsevier. (Cited on page 31.)

[Liu et al., 2020] Liu, Y., Deng, Y., Elkashlan, M., Nallanathan, A., and Karagiannidis, G. K. (2020). Analyzing grant-free access for urllc service. *IEEE Journal on Selected Areas in Communications*, 39(3):741–755. (Cited on page 6.)

[Liu et al., 2021] Liu, Y., Deng, Y., Zhou, H., Elkashlan, M., and Nallanathan, A. (2021). A general deep reinforcement learning framework for grant-free noma optimization in murllc. *arXiv preprint arXiv:2101.00515*. (Cited on pages 9 and 56.)

[Lowe et al., 2017] Lowe, R. et al. (2017). Multi-agent actor-critic for mixed cooperative-competitive environments. In *NeurIPS*. (Cited on pages 36, 93, 97 and 102.)

[Mahmood et al., 2019a] Mahmood, N. H., Abreu, R., Böhnke, R., Schubert, M., Berardinelli, G., and Jacobsen, T. H. (2019a). Uplink grant-free access solutions for urllc services in 5g new radio. In *2019 16th International Symposium on Wireless Communication Systems (ISWCS)*, pages 607–612. IEEE. (Cited on pages 5, 6, 82 and 99.)

[Mahmood et al., 2019b] Mahmood, N. H., Karimi, A., Berardinelli, G., Pedersen, K. I., and Laselva, D. (2019b). On the resource utilization of multi-connectivity transmission for urllc services in 5g new radio. In *2019 IEEE Wireless Communications and Networking Conference Workshop (WCNCW)*, pages 1–6. IEEE. (Cited on page 7.)

[MB et al., 2020] MB, Y., Shivashetty, D., et al. (2020). D2d communication in internet of things: Conventional communication protocols, challenges and open issues. *Institute of Scholars (InSc)*. (Cited on page 127.)

[Metz et al., 2017] Metz, L., Ibarz, J., Jaitly, N., and Davidson, J. (2017). Discrete sequential prediction of continuous actions for deep rl. *arXiv preprint arXiv:1705.05035*. (Cited on page 57.)

[Miao et al., 2016] Miao, G., Zander, J., Sung, K. W., and Slimane, S. B. (2016). *Fundamentals of mobile data networks*. Cambridge University Press. (Cited on page 2.)

[Militano et al., 2015] Militano, L., Araniti, G., Condoluci, M., Farris, I., and Iera, A. (2015). Device-to-device communications for 5g internet of things. *EAI Endorsed Transactions on Internet of Things*, 1(1). (Cited on page 127.)

[Mills-Tettey et al., 2007] Mills-Tettey, G. A., Stentz, A., and Dias, M. B. (2007). The dynamic hungarian algorithm for the assignment problem with

changing costs. *Robotics Institute, Pittsburgh, PA, Tech. Rep. CMU-RI-TR-07-27*. (Cited on page 49.)

[Mnih et al., 2015] Mnih, V. et al. (2015). Human-level control through deep reinforcement learning. *Nature*, 518(7540):529–533. (Cited on pages 21, 22, 34 and 56.)

[Mo and Walrand, 2000] Mo, J. and Walrand, J. (2000). Fair end-to-end window-based congestion control. *IEEE/ACM Transactions on networking*, 8(5):556–567. (Cited on page 3.)

[Moyal, 2013] Moyal, P. (2013). On queues with impatience: stability, and the optimality of earliest deadline first. *Queueing Systems*, 75:211–242. (Cited on page 75.)

[Mozaffari et al., 2016] Mozaffari, M., Saad, W., Bennis, M., and Debbah, M. (2016). Unmanned aerial vehicle with underlaid device-to-device communications: Performance and tradeoffs. *IEEE Transactions on Wireless Communications*, 15(6):3949–3963. (Cited on page 1.)

[Naparstek and Cohen, 2019] Naparstek, O. and Cohen, K. (2019). Deep multi-user reinforcement learning for distributed dynamic spectrum access. *IEEE Transactions on Wireless Communications*, 18(1):310–323. (Cited on page 107.)

[Nash, 1951] Nash, J. (1951). Non-cooperative games. *Annals of mathematics*, pages 286–295. (Cited on page 32.)

[Neto et al., 2021] Neto, F. H. C., Araújo, D. C., Mota, M. P., Maciel, T. F., and de Almeida, A. L. (2021). Uplink power control framework based on reinforcement learning for 5g networks. *IEEE Transactions on Vehicular Technology*, 70(6):5734–5748. (Cited on page 9.)

[Nomeir et al., 2021] Nomeir, M. W., Gadallah, Y., and Seddik, K. G. (2021). Uplink scheduling for mixed grant-based embb and grant-free urllc traffic in 5g networks. In *2021 17th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob)*, pages 187–192. (Cited on page 4.)

[Oliehoek, 2012] Oliehoek, F. A. (2012). *Decentralized POMDPs*, pages 471–503. Springer Berlin Heidelberg, Berlin, Heidelberg. (Cited on pages 33, 93, 96 and 111.)

[Papadimitriou and Tsitsiklis, 1987] Papadimitriou, C. H. and Tsitsiklis, J. N. (1987). The complexity of markov decision processes. *Mathematics of operations research*, 12(3):441–450. (Cited on page 70.)

[Park and Saad, 2019] Park, T. and Saad, W. (2019). Distributed learning for low latency machine type communication in a massive internet of things. *IEEE Internet of Things Journal*, 6(3):5562–5576. (Cited on page 127.)

[Peters and Schaal, 2008] Peters, J. and Schaal, S. (2008). Natural actor-critic. *Neurocomputing*, 71(7-9):1180–1190. (Cited on page 30.)

[Polyanskiy et al., 2010] Polyanskiy, Y., Poor, H. V., and Verdú, S. (2010). Channel coding rate in the finite blocklength regime. *IEEE Transactions on Information Theory*, 56(5):2307–2359. (Cited on pages 64 and 65.)

[Popovski et al., 2019] Popovski, P., Stefanovic, C., Nielsen, J. J., de Carvalho, E., Angjelichinoski, M., Trillingsgaard, K. F., and Bana, A.-S. (2019). Wireless access in ultra-reliable low-latency communication (urllc). *IEEE Transactions on Communications*, 67:5783–5801. (Cited on page 79.)

[Puterman, 1990] Puterman, M. L. (1990). Markov decision processes. *Handbooks in operations research and management science*, 2:331–434. (Cited on pages 19, 20 and 21.)

[Qi et al., 2020] Qi, R., Chi, X., Zhao, L., and Yang, W. (2020). Martingales-based aloha-type grant-free access algorithms for multi-channel networks with mmtc/urllc terminals co-existence. *IEEE access*, 8:37608–37620. (Cited on page 106.)

[Qu et al., 2020] Qu, G., Lin, Y., Wierman, A., and Li, N. (2020). Scalable multi-agent reinforcement learning for networked systems with average reward. *Advances in Neural Information Processing Systems*, 33:2074–2086. (Cited on page 36.)

[Rajagopalan et al., 2009] Rajagopalan, S., Shah, D., and Shin, J. (2009). Network adiabatic theorem: An efficient randomized protocol for contention resolution. In *Proceedings of the eleventh international joint conference on Measurement and modeling of computer systems*, pages 133–144. (Cited on page 3.)

[Rashid et al., 2018] Rashid, T. et al. (2018). Qmix: Monotonic value function factorisation for deep multi-agent reinforcement learning. In *ICML*. (Cited on pages 12, 35 and 97.)

[Ren et al., 2020] Ren, H., Pan, C., Deng, Y., Elkashlan, M., and Nallanathan, A. (2020). Joint power and blocklength optimization for urllc in a factory automation scenario. *IEEE Transactions on Wireless Communications*, 19(3):1786–1801. (Cited on pages 10 and 61.)

[Roberts, 1975] Roberts, L. G. (1975). ALOHA packet system with and without slots and capture. *ACM SIGCOMM Comput. Commun. Rev.*, 5(2):28–42. (Cited on pages 3 and 42.)

[Rumelhart et al., 1986] Rumelhart, D. E., Hinton, G. E., and Williams, R. J. (1986). Learning representations by back-propagating errors. *nature*, 323(6088):533–536. (Cited on page 17.)

[Saito et al., 2013] Saito, Y., Kishiyama, Y., Benjebbour, A., Nakamura, T., Li, A., and Higuchi, K. (2013). Non-orthogonal multiple access (noma) for cellular future radio access. In *2013 IEEE 77th vehicular technology conference (VTC Spring)*, pages 1–5. IEEE. (Cited on pages 6 and 61.)

[Salaün et al., 2020] Salaün, L., Coupechoux, M., and Chen, C. S. (2020). Joint subcarrier and power allocation in noma: Optimal and approximate algorithms. *IEEE Transactions on Signal Processing*, 68:2215–2230. (Cited on pages 10 and 61.)

[Saxe et al., 2013] Saxe, A. M., McClelland, J. L., and Ganguli, S. (2013). Exact solutions to the nonlinear dynamics of learning in deep linear neural networks. *arXiv preprint arXiv:1312.6120*. (Cited on page 50.)

[Schulman et al., 2015a] Schulman, J., Levine, S., Abbeel, P., Jordan, M., and Moritz, P. (2015a). Trust region policy optimization. In *International confer-*

ence on machine learning, pages 1889–1897. PMLR. (Cited on pages 27, 28 and 112.)

[Schulman et al., 2015b] Schulman, J., Moritz, P., Levine, S., Jordan, M., and Abbeel, P. (2015b). High-dimensional continuous control using generalized advantage estimation. *arXiv preprint arXiv:1506.02438*. (Cited on pages 28, 46, 74 and 114.)

[Schulman et al., 2017] Schulman, J., Wolski, F., Dhariwal, P., Radford, A., and Klimov, O. (2017). Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*. (Cited on pages 11, 28, 34, 41, 46, 59, 113 and 114.)

[Segura et al., 2022] Segura, D., Khatib, E. J., and Barco, R. (2022). Dynamic packet duplication for industrial urllc. *Sensors*, 22(2):587. (Cited on pages 7 and 121.)

[Shahab et al., 2020] Shahab, M. B., Abbas, R., Shirvanimoghaddam, M., and Johnson, S. J. (2020). Grant-free non-orthogonal multiple access for iot: A survey. *IEEE Communications Surveys & Tutorials*, 22(3):1805–1838. (Cited on page 6.)

[Shapley, 1953] Shapley, L. S. (1953). Stochastic games. *Proceedings of the national academy of sciences*, 39(10):1095–1100. (Cited on page 31.)

[Simon et al., 1994] Simon, M. K., Omura, J. K., Scholtz, R. A., and Levitt, B. K. (1994). *Spread Spectrum Communications Handbook (Revised Ed.)*. McGraw-Hill, Inc., USA. (Cited on page 2.)

[Slivkins, 2019] Slivkins, A. (2019). Introduction to multi-armed bandits. *arXiv preprint arXiv:1904.07272*. (Cited on page 40.)

[Sohaib et al., 2021] Sohaib, M., Jeong, J., and Jeon, S.-W. (2021). Dynamic multichannel access via multi-agent reinforcement learning: Throughput and fairness guarantees. *IEEE Transactions on Wireless Communications*, 21(6):3994–4008. (Cited on page 106.)

[Sondik, 1971] Sondik, E. J. (1971). *The optimal control of partially observable Markov processes*. Stanford University. (Cited on pages 21 and 58.)

[Stankovic et al., 1998] Stankovic, J. A., Spuri, M., Ramamritham, K., and Buttazzo, G. (1998). *Deadline scheduling for real-time systems: EDF and related algorithms*, volume 460. Springer Science & Business Media. (Cited on page 75.)

[Subramanian and Mahajan, 2019] Subramanian, J. and Mahajan, A. (2019). Reinforcement learning in stationary mean-field games. In *Proceedings of the 18th International Conference on Autonomous Agents and MultiAgent Systems*, pages 251–259. (Cited on pages 37 and 127.)

[Subramanian et al., 2022] Subramanian, J., Sinha, A., Seraj, R., and Mahajan, A. (2022). Approximate information state for approximate planning and reinforcement learning in partially observed systems. *J. Mach. Learn. Res.*, 23:12–1. (Cited on page 58.)

[Sukhbaatar et al., 2016] Sukhbaatar, S., Fergus, R., et al. (2016). Learning multiagent communication with backpropagation. *Advances in neural information processing systems*, 29. (Cited on page 37.)

[Sunehag et al., 2018] Sunehag, P. et al. (2018). Value-decomposition networks for cooperative multi-agent learning. In *AAMAS*. (Cited on pages 35 and 97.)

[Sutton and Barto, 2018] Sutton, R. S. and Barto, A. G. (2018). *Reinforcement learning: An introduction*. MIT press. (Cited on pages 19, 21 and 24.)

[Sutton et al., 1999] Sutton, R. S., McAllester, D., Singh, S., and Mansour, Y. (1999). Policy gradient methods for reinforcement learning with function approximation. *Advances in neural information processing systems*, 12. (Cited on page 24.)

[Szepesvári and Littman, 1999] Szepesvári, C. and Littman, M. L. (1999). A unified analysis of value-function-based reinforcement-learning algorithms. *Neural computation*, 11(8):2017–2060. (Cited on page 22.)

[Tampuu et al., 2017] Tampuu, A., Matiisen, T., Kodelja, D., Kuzovkin, I., Korjus, K., Aru, J., Aru, J., and Vicente, R. (2017). Multiagent cooperation and competition with deep reinforcement learning. *PloS one*, 12(4):e0172395. (Cited on page 34.)

[Tan et al., 2020] Tan, J., Liang, Y.-C., Zhang, L., and Feng, G. (2020). Deep reinforcement learning for joint channel selection and power control in d2d networks. *IEEE Transactions on Wireless Communications*, 20(2):1363–1378. (Cited on page 126.)

[Tan, 1993] Tan, M. (1993). Multi-agent reinforcement learning: Independent vs. cooperative agents. In *ICML*. (Cited on pages 33, 34 and 106.)

[Tan et al., 2021] Tan, X. et al. (2021). Cooperative multi-agent reinforcement learning based distributed dynamic spectrum access in cognitive radio networks. *arXiv:2106.09274*. (Cited on pages 9, 12 and 93.)

[Tan et al., 2022] Tan, X., Zhou, L., Wang, H., Sun, Y., Zhao, H., Seet, B.-C., Wei, J., and Leung, V. C. (2022). Cooperative multi-agent reinforcement-learning-based distributed dynamic spectrum access in cognitive radio networks. *IEEE Internet of Things Journal*, 9(19):19477–19488. (Cited on page 107.)

[Tang et al., 2021] Tang, W., Chen, X., Chen, M. Z., Dai, J. Y., Han, Y., Jin, S., Cheng, Q., Li, G. Y., and Cui, T. J. (2021). On channel reciprocity in reconfigurable intelligent surface assisted wireless networks. *IEEE Wireless Communications*, 28(6):94–101. (Cited on page 109.)

[Tavakoli et al., 2018] Tavakoli, A., Pardo, F., and Kormushev, P. (2018). Action branching architectures for deep reinforcement learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*. (Cited on pages 57, 59, 77 and 82.)

[Tegos et al., 2020] Tegos, S. A., Diamantoulakis, P. D., Lioumpas, A. S., Sarigiannidis, P. G., and Karagiannidis, G. K. (2020). Slotted aloha with noma for the next generation iot. *IEEE Transactions on Communications*, 68(10):6289–6301. (Cited on pages 6 and 82.)

[Titsias and Nikoloutsopoulos, 2018] Titsias, M. K. and Nikoloutsopoulos, S. (2018). Bayesian transfer reinforcement learning with prior knowledge rules. *arXiv preprint arXiv:1810.00468*. (Cited on pages 11, 59 and 76.)

[Tokgoz and Rao, 2006] Tokgoz, Y. and Rao, B. D. (2006). The effect of imperfect channel estimation on the performance of maximum ratio combining in the

presence of cochannel interference. *IEEE transactions on vehicular technology*, 55(5):1527–1534. (Cited on page 64.)

[Tse and Viswanath, 2005] Tse, D. and Viswanath, P. (2005). *Fundamentals of wireless communication*. Cambridge university press. (Cited on pages 63, 65, 77 and 78.)

[Vilà et al., 2023] Vilà, I., Sallent, O., and Pérez-Romero, J. (2023). On the design of a network digital twin for the radio access network in 5g and beyond. *Sensors*, 23(3):1197. (Cited on page 125.)

[Vishwakarma et al., 2019] Vishwakarma, S. K., Upadhyaya, P., Kumari, B., and Mishra, A. K. (2019). Smart energy efficient home automation system using iot. In *2019 4th international conference on internet of things: Smart innovation and usages (IoT-SIU)*, pages 1–4. Ieee. (Cited on page 1.)

[Wang et al., 2020] Wang, S., Lv, T., Zhang, X., Lin, Z., and Huang, P. (2020). Learning-based multi-channel access in 5g and beyond networks with fast time-varying channels. *IEEE Transactions on Vehicular Technology*, 69(5):5203–5218. (Cited on page 106.)

[Watkins and Dayan, 1992] Watkins, C. J. and Dayan, P. (1992). Q-learning. *Machine learning*, 8(3):279–292. (Cited on page 22.)

[Wiering and Van Otterlo, 2012] Wiering, M. A. and Van Otterlo, M. (2012). Reinforcement learning. *Adaptation, learning, and optimization*, 12(3):729. (Cited on page 58.)

[Williams, 1992] Williams, R. J. (1992). Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8(3):229–256. (Cited on page 24.)

[Xu et al., 2020] Xu, Y., Yu, J., and Buehrer, R. M. (2020). The application of deep reinforcement learning to distributed spectrum access in dynamic heterogeneous environments with partial observations. *IEEE Trans. Wireless Commun.*, 19(7):4494–4506. (Cited on pages 9, 70, 82 and 92.)

[Xu et al., 2018] Xu, Y., Yu, J., Headley, W. C., and Buehrer, R. M. (2018). Deep reinforcement learning for dynamic spectrum access in wireless networks. In *IEEE MILCOM*. (Cited on pages 9 and 92.)

[Yang et al., 2020] Yang, H. et al. (2020). Deep reinforcement learning based massive access management for ultra-reliable low-latency communications. *IEEE Trans. Wireless Commun.*, 20(5):2977–2990. (Cited on pages 9, 56 and 127.)

[Yang et al., 2018] Yang, Y., Luo, R., Li, M., Zhou, M., Zhang, W., and Wang, J. (2018). Mean field multi-agent reinforcement learning. In *International Conference on Machine Learning*, pages 5571–5580. PMLR. (Cited on pages 37 and 127.)

[Ye et al., 2021] Ye, X., Yu, Y., and Fu, L. (2021). Multi-channel opportunistic access for heterogeneous networks based on deep reinforcement learning. *IEEE Transactions on Wireless Communications*, 21(2):794–807. (Cited on pages 107 and 118.)

[Yu et al., 2022] Yu, C., Velu, A., Vinitsky, E., Gao, J., Wang, Y., Bayen, A., and Wu, Y. (2022). The surprising effectiveness of ppo in cooperative multi-agent games. *Advances in Neural Information Processing Systems*, 35:24611–24624. (Cited on pages 36 and 107.)

[Yu et al., 2019] Yu, Y., Wang, T., and Liew, S. C. (2019). Deep-reinforcement learning multiple access for heterogeneous wireless networks. *IEEE J. Sel. Areas Commun*, 37(6):1277–1290. (Cited on page 93.)

[Yu et al., 2018] Yu, Z., Xu, Y., and Tong, L. (2018). Deadline scheduling as restless bandits. *IEEE Transactions on Automatic Control*, 63(8):2343–2358. (Cited on page 40.)

[Zhang et al., 2020] Zhang, H., Yang, N., Huangfu, W., Long, K., and Leung, V. C. (2020). Power control based on deep reinforcement learning for spectrum sharing. *IEEE Transactions on Wireless Communications*, 19(6):4209–4219. (Cited on page 126.)

[Zhang et al., 2018] Zhang, K., Yang, Z., Liu, H., Zhang, T., and Basar, T. (2018). Fully decentralized multi-agent reinforcement learning with networked agents. In *ICML*, pages 5872–5881. PMLR. (Cited on page 36.)

[Zhang et al., 2021] Zhang, K., Yang, Z., Liu, H., Zhang, T., and Başar, T. (2021). Finite-sample analysis for decentralized batch multiagent reinforcement learning with networked agents. *IEEE Transactions on Automatic Control*, 66(12):5925–5940. (Cited on page 36.)

[Zhang and Liang, 2020] Zhang, L. and Liang, Y.-C. (2020). Deep reinforcement learning for multi-agent power control in heterogeneous networks. *IEEE Transactions on Wireless Communications*, 20(4):2551–2564. (Cited on page 126.)

[Zhong et al., 2018] Zhong, C., Lu, Z., Gursoy, M. C., and Velipasalar, S. (2018). Actor-critic deep reinforcement learning for dynamic multichannel access. In *2018 IEEE Glob. Conf. Signal Inf. Process. (GlobalSIP)*, pages 599–603. IEEE. (Cited on page 40.)

[Zhong et al., 2019] Zhong, C., Lu, Z., Gursoy, M. C., and Velipasalar, S. (2019). A deep actor-critic reinforcement learning framework for dynamic multichannel access. *IEEE Transactions on Cognitive Communications and Networking*, 5(4):1125–1139. (Cited on page 106.)

ECOLE
DOCTORALE

Titre : Apprentissage par Renforcement pour l'Accès Multiple Non-Coordonné.

Mots clés : Accès multiple, Apprentissage par Renforcement, Systèmes Multi-Agents, Réseaux sans fil.

Résumé : Les protocoles de contrôle d'accès au support (MAC) distribués sont fondamentaux dans la communication sans fil, mais les protocoles traditionnels basés sur l'accès aléatoire sont confrontés à des limitations importantes dans le cas d'utilisation de l'internet des objets (IoT). En effet, ils ont du mal à garantir la latence, ce qui les rend inadaptés aux communications ultra-fiables à faible latence (URLLC). Cette thèse aborde ces défis en exploitant le potentiel de l'apprentissage par renforcement profond (DRL), un paradigme dans lequel les agents optimisent leurs actions en interagissant avec un environnement.
Cette thèse aborde les principaux défis du problème de l'accès multiple (MA) pour les réseaux URLLC, incluant la latence des protocoles centralisés, les collisions et retransmissions des protocoles sans allocation (GF) ainsi que les complexités pour gérer l'hétérogénéité des appareils et les environnements dynamiques. En outre, la thèse explore l'intégration de nouvelles techniques de couche physique comme l'accès multiple non orthogonal (NOMA).
Notre méthodologie applique le DRL pour développer des protocoles intelligents, qui ont déjà montré leur efficacité dans les applications IoT. Dans un premier temps, nous modélisons le problème de l'URLLC dans un paradigme centralisé, où la station de base (BS) orchestre les transmissions des appareils. Cette configuration présente l'avantage d'assurer une communication sans collision, mais introduit une observabilité partielle, car la station de base n'a pas accès à la mémoire et à l'état du canal des utilisateurs. Nous nous attaquons à ce problème en introduisant deux algorithmes : FilteredPPO et NOMA-PPO. Alors que le premier surpasse les algorithmes de référence dans les scénarios avec trafic périodique, le second démontre une performance supérieure à l'état de l'art dans les scénarios avec trafic sporadique. Les troisième et quatrième contributions, SeqDQN et MCA-PPO, étudient l'application de l'apprentissage par renforcement multi-agents (MARL) pour l'URLLC où chaque appareil est équipé d'un algorithme DRL. Alors que SeqDQN explore une méthode pour réduire la non-stationnarité et améliore la scalabilité et l'apprentissage, MCA-PPO présente une solution théoriquement robuste pour le défi de l'accès dynamique multicanal (DMCA) permettant aux utilisateurs d'optimiser l'utilisation de la bande passante et donc d'améliorer les performances URLLC.

Title : Reinforcement Learning for Uncoordinated Multiple Access

Keywords : Multiple Access, Reinforcement Learning, Multi-Agent Systems, Wireless Networks.

Abstract : Distributed Medium Access Control (MAC) protocols are fundamental in wireless communication, yet traditional random access-based protocols face significant limitations dealing with the Internet-of-Things (IoT) use cases. Indeed, they struggle with latency guarantees, making them unsuitable for Ultra Reliable Low Latency Communications (URLLC). This thesis addresses these challenges by leveraging the potential of Deep Reinforcement Learning (DRL), a paradigm where decision-makers optimize actions by interacting with an environment.
This thesis tackles key challenges in the Medium Access (MA) problem for URLLC networks, including the latency in centralized protocols, the collision and retransmission issues in Grant-Free (GF) protocols, the complexities to handle device heterogeneity and dynamic environments. Furthermore, the thesis explores the integration of new physical layer techniques like Non-Orthogonal Multiple Access (NOMA).
Our methodology applies DRL to develop intelligent protocols, which has already shown effectiveness in addressing IoT applications. Initially, we model the URLLC problem within a centralized paradigm, where the Base Station (BS) orchestrates device transmissions. This setup has the benefit to ensure collision-free communication but introduces partial observability as the BS does not have access to the users' buffer and channel state. We tackle this problem by introducing two algorithms: FilteredPPO and NOMA-PPO. While the former outperforms the benchmarks in scenarios with periodic traffic patterns, the latter demonstrates superior performance over the state-of-the-art baselines on scenarios with sporadic traffic. The third and fourth contributions, SeqDQN and MCA-PPO, study the application of Multi-Agent Reinforcement Learning (MARL) for URLLC where each device is equipped by a DRL algorithm. While SeqDQN explores a method to reduce non-stationarity and enhances scalability and training efficiency, MCA-PPO presents a theoretically robust solution for the Dynamic Multi-Channel Access (DMCA) challenge allowing users to optimize bandwidth utilization, and thus enhancing the URLLC performance.

Institut Polytechnique de Paris
91120 Palaiseau, France