Proteção de Serviços URLLC para Indústria 4.0 Utilizando Aprendizagem por Reforço Multi-Agente

Cleverson Veloso Nahum, Weskley Vinicius Fernandes Mauricio, Maykon Renan Pereira da Silva, Francisco Raimundo Albuquerque Parente e Aldebaro Klautau

***Resumo*— As redes 5G/B5G são tecnologias habilitadoras de cenários da Indústria 4.0 tendo que lidar com toda a diversidade de aplicações e requisitos para garantir o funcionamento correto dos casos de usos industriais. Com esse intuito, os escalonadores de recursos de rádio precisam atender os requisitos definidos e proteger os *slices* prioritários quando os recursos disponíveis não são suficientes. Neste trabalho, nós propomos um escalonador de recursos de rádio baseado em aprendizado por reforço multi-agente para desempenhar as funções de escalonador *inter-slice* e *intra-slice* visando atender aos *slices* de eMBB, mMTC e URLLC (*slice* prioritário). Os resultados obtidos demonstram que o escalonador proposto obteve melhor desempenho na redução de violações e proteção do *slice* prioritário em relação ao *baseline*.**

***Palavras-Chave*— Aprendizado por reforço multi-agente, escalonamento de recurso de rádio, Indústria 4.0, *network slice*.**

***Abstract*— 5G/B5G networks are enabling technologies for Industry 4.0 scenarios, where mobile networks will have to handle a wide variety of applications and requirements to ensure the proper functioning of industrial use cases. Therefore, radio resource schedulers need to meet the defined requirements and protect high priority slices when the available resources are insufficient. In this work, we propose a radio resource scheduler based on a multi-agent reinforcement learning approach to perform inter-slice and intra-slice scheduling functions aimed at serving the slices of eMBB, mMTC, and URLLC (priority slice). The obtained results demonstrate that the proposed scheduler achieved better performance in reducing violations and protecting the priority slice compared to the baseline.**

***Keywords*— Industry 4.0, multi-agent reinforcement learning, network slice, radio resource scheduling.**

I. Introdução

Em sistemas de quinta geração (5G), o conceito de *Network Slicing* (NS) tem recebido especial interesse ao permitir o compartilhamento de recursos entre os diferentes nós da rede. Esse conceito surgiu do recente avanço nas tecnologias de computação e virtualização de funções e possibilita o fornecimento de serviços personalizados para cada cenário de aplicação distinto. O NS é comumente usado para alocação de recursos a fim de garantir o bom desempenho de serviços heterogêneos em 5G. Esses serviços incluem *enhanced Mobile Broadband* (eMBB), *Massive Machine Type Communication* (mMTC) e *Ultra Reliable Low Latency Communications* (URLLC) [1]. Nesses serviços, questões relacionadas a NS precisam ser analisadas, como orquestração *intra-slice* e *inter-slice*, bem como alocação dinâmica de recursos. Os principais desafios estão relacionados à largura de banda limitada e aos requisitos de *slices* heterogêneos, os quais exigem planejamento cuidadoso e técnicas de compartilhamento eficientes [2].

Uma alternativa que tem se mostrado bastante eficaz para melhorar a *Quality of Service* (QoS) em NS é o uso de algoritmos de aprendizagem de máquina. Em particular, o uso de agentes de *Reinforcement Learning* (RL) é capaz de escalonar os recursos de rede para atender a requisitos rigorosos de variados casos de uso em 5G [2]. Entretanto, algumas abordagens na literatura que utilizam algoritmos de RL sofrem com o provisionamento excessivo e/ou violações frequentes do *Service Level Agreements* (SLA), já que não garantem o cumprimento dos requisitos ou proteção de *slices* de rede que são essenciais para *Radio Resource Scheduling* (RRS) em NS da *Radio Access Network* (RAN) [3]. Em [4], os autores propõem um problema de *Multi-Agent Reinforcement Learning* (MARL) considerando um cenário de rede celular densa que contém vários *slices* de rede em múltiplas *Base Station*s (BSs) com o objetivo de gerenciar os recursos entre *slices* em tempo real. Em [5], os autores apresentam um novo algoritmo MARL para fatiamento de uma rede *Open* RAN, projetado para se adaptar a números de fatias variáveis e escalar efetivamente à medida que aumentam. Já em [6], é proposto um método baseado em RL hierárquico para alocar recursos de rádio para usuários móveis que utilizam serviços do tipo URLLC e eMBB. Considerando cenários da Indústria 4.0, Nahum *et al.* [7] introduzem um agente RL de RRS para NS capaz de orquestrar recursos de rádio entre vários *slices* para atender aos seus requisitos e dar prioridade à latência e à confiabilidade, resultando em menos violações de *slice* com foco na proteção de aplicações críticas (*slice* URLLC). Essa proteção no âmbito da Indústria 4.0 é uma demanda essencial para o RRS devido às variações de rede e canal do dispositivo, que podem causar instabilidade, um cenário que requer adaptações constantes do RRS para atender aos requisitos críticos da aplicação.

Neste artigo, introduzimos um novo escalonador MARL utilizando o método *Soft Actor-Critic* (SAC). A proposta estende o agente introduzido em [7] a fim de dar suporte a RL nas funções *inter-slice* e *intra-slice* do escalonador.

C. V. Nahum e A. Klautau fazem parte da Universidade Federal do Pará (UFPA). W. V. F. Mauricio, M. R. P. da Silva e F. R. A. Parente fazem parte da Fundação Centro de Pesquisa e Desenvolvimento em Telecomunicações (CPQD), Campinas-SP, Brasil. E-mail: {cleverson, aldebaro}@ufpa.br,{wmauricio, mpsilva, fparente}@cpqd.com.br. Este trabalho foi financiado parcialmente pela RNP, com recursos do MCTIC, Outorga nº 01245.010604/2020-14, no âmbito do Projeto Brasil 6G do Instituto Nacional de Telecomunicações (Inatel), Brasil; Projeto Universal (CNPq grant 405111/2021-5); OpenRAN Brazil - Fase 2 (MCTI grant Nº A01245.014203/2021-14), Projeto Smart 5G Core And MUltiRAn Integration (SAMURAI) (MCTIC/CGI.br/FAPESP under Grant 2020/05127-2) e, pelo projeto ForELERAN apoiado pelo Centro de Excelência em Redes Abertas EMBRAPII/CPQD, com recursos financeiros do PPI IoT/Manufatura 4.0/PPI HardwareBR do edital MCTI número 51/2023, firmado com a EMBRAPII.

Como resultado, conseguimos aperfeiçoar a proteção aos *slices* críticos em aplicações da Indústria 4.0, para os quais o RRS é priorizado, melhorando portanto os requisitos de QoS.

II. Modelo de Sistema

Neste trabalho, consideramos o aspecto de *downlink* de um sistema Múltipla Entrada Múltipla Saída (MIMO) industrial *indoor* que utiliza a tecnologia *Frequency Division Duplex* (FDD) *Orthogonal Frequency Division Multiple Access* (OFDMA), como em um galpão de fábrica. Esse sistema é composto por uma BS equipada com um modelo de antena 3D $n_v \times n_h$ [8], servindo U *User Equipment*s (UEs), cada um equipado com antenas únicas omnidirecionais, onde n_v e n_h são o número de antenas verticais e horizontais, respectivamente. Dentro deste contexto, identificamos três tipos de *slices* distintos, cada um adaptado para casos de uso específicos, com requisitos únicos de atraso e taxas de dados alcançáveis. Para simplificar nossa análise, assumimos que cada UE pertence exclusivamente a um desses tipos de *slices*. Além disso, consideramos R *Resource Block*s (RBs) disponíveis para cada *Transmission Time Interval* (TTI). Um RB representa a menor unidade de recursos que pode ser alocada para uma UE. Cada RB é subdividida em N_{sc} subportadoras OFDMA adjacentes e N_{symb} símbolos *Orthogonal Frequency Division Multiplexing* (OFDM) consecutivos.

Em particular, utilizamos a técnica de precodificação em diversidade *Maximum Ratio Transmission* (MRT), que pode ser definida como

$$\boldsymbol{x} = \frac{\boldsymbol{h}^{\mathrm{H}}}{\|\boldsymbol{h}\|_{\mathrm{F}}}, \tag{1}$$

onde $\boldsymbol{x}$ é o vetor de precodificação $n_v \cdot n_h \times 1$, $\boldsymbol{h}$ é o vetor de canal *downlink* $1 \times n_v \cdot n_h$ entre BS e UE, e $\|.\|_{\mathrm{F}}$ é a norma de Frobenius. Assim, definimos o vetor de canal efetivo do *dowlink* como $\boldsymbol{h}_u^{\mathrm{eff}}$ do UE u após a aplicação do precoder MRT como

$$\boldsymbol{h}_u^{\mathrm{eff}} = \boldsymbol{h}_u \boldsymbol{x}_u^{\mathrm{H}}, \tag{2}$$

onde $\boldsymbol{h}_u$ é o vetor de canal *downlink* $1 \times n_v \cdot n_h$ do UE u. Dado que a BS aloca o r-th RB para o u-th UE, a *Signal-to-Noise Ratio* (SNR) percebida pelo UE pode ser definida como

$$\gamma_{u,r} = \frac{\alpha_j \; p_r \; |h_{u,r}^{\mathrm{eff}}|^2}{\sigma_u^2}, \tag{3}$$

onde p_r é a potência de transmissão alocada para o r-th RB, α_u é o efeito do ganho de percurso e sombreamento sofrido pelo u-th UE, e σ_u é a potência do ruído sofrido pelo u-th UE. Desta forma, a eficiência espectral $\mathsf{S}_{u,n}(n)$ do RB r e UE u é definida como

$$\mathsf{S}_{u,r}(n) = \log_2(1 + \gamma_{u,r}). \tag{4}$$

Além disso, usamos o modelo de canal *Quasi Deterministic Radio Channel Generator* (QuaDRiGa) considerando *Line Of Sight* (LOS) e *Non-Line Of Sight* (NLOS), como documentado na referência [9]. Utilizamos um cenário especificamente feito para aplicações industriais *indoor*, tais quais galpões de fábricas, no contexto de ambientes da Indústria 4.0. Para uma compreensão abrangente dos parâmetros de canal usados para criar este cenário, tais como detalhes sobre *shadow fading*, *small fading* e *angular spread*, o leitor pode consultar a referência [9].

A. Slicing de rede e requisitos do slice

Cada *slice* de rede, denominado como s, engloba um grupo de UEs U_s que possuem requisitos idênticos de QoS e comportamento de tráfego semelhante. O vetor $\boldsymbol{R}_n$ contém o número de *Resource Block Group*s (RBGs) alocados para cada *slice* pelo escalonador *inter-slice* no passo de simulação n, que pode ser definido como

$$\boldsymbol{R}_n = [R_1(n), R_2(n), \ldots, R_S(n)], \tag{5}$$

onde $R_s(n)$ representa o número de RBGs allocados para o *slice* s no passo n.

O princípio RRS adere a um processo dado por

$$\sum_{s=1}^{S} R_s(n) = R, \tag{6}$$

garantindo que a alocação agregada dos RBGs em todos os *slices* corresponda consistentemente ao total de RBGs disponíveis, designado como R. Consequentemente, o papel primário do RRS em um cenário que envolve a segmentação da RAN é determinar a alocação do $R_s(n)$ para cada *slice* s em um passo n específico, alinhando-a com as condições da rede para cumprir os requisitos específicos do *slice*. Depois que os RBs foram alocados entre os vários *slices*, o escalonamento *intra-slice* assume a responsabilidade de distribuir esses RBs entre os UEs individuais dentro de cada *slice*.

As exigências dos *slices* estabelecem valores desejados específicos para várias métricas de rede monitoradas. Essas exigências são formuladas com foco em duas métricas-chave: taxa de transferência atendida e atraso de *buffer*. Neste trabalho, calculamos as métricas de rede de acordo com [10] considerando a taxa de transferência atendida (taxa de dados alcançada) $r_u(n)$, a taxa de ocupação do *buffer* $b_u^{\mathrm{occ}}(n)$ e o atraso do *buffer* $d_u(n)$.

A taxa de transferência atendida para cada UE é a taxa máxima em bits por passo que um UE pode obter, considerando o número de RBGs alocado a ele e sua eficiência espectral

$$r_u(n) = \left\lfloor \frac{(R_s^u(n)/R)\mathrm{BS}_u(n)}{P} \right\rfloor P, \tag{7}$$

onde $R_s^u(n)$ representa o número de RBGs alocados ao u-th UE pelo escalonador *intra-slice*, B é a largura de banda total disponível na BS, $S_u(n)$ é a eficiência espectral para o u-th UE no passo n, e P é o tamanho do pacote. Já a taxa de ocupação do *buffer* é definida como

$$b_u^{\mathrm{occ}}(n) = \frac{b_u(n)}{b_{\max}}, \tag{8}$$

onde $b_u(n)$ representa a quantidade de dados disponível no *buffer* do u-th UE no passo de simulação n e $b_{\max}$ é a capacidade máxima do *buffer* do UE. Pacotes são perdidos toda vez que o *buffer* está cheio ou o atraso do pacote excede o atraso máximo $d_{\max}$ permitido pelo *buffer*. O atraso do *buffer*

representa o tempo médio que cada pacote esperou antes de ser enviado ou perdido e é definido como

$$d_u(n) = \frac{\sum_{i=0}^{d_{\max}} i d_n^u(i)}{\sum_{i=0}^{d_{\max}} d_n^u(i)}, \tag{9}$$

onde $d_u(n)$ é um vetor de dimensão $d_{\max} + 1$ representando o atraso dos pacotes no *buffer* do u-th UE no passo n.

Caracterizamos as métricas do *slice* da rede calculando a média das métricas para todos os UEs associados a um *slice* específico. Cada tipo de *slice* vem com conjuntos distintos de requisitos. Neste artigo, consideramos três tipos distintos de *slice*: eMBB, URLLC e mMTC.

1) slice eMBB: UEs alocados para o *slice* eMBB priorizam altas taxas de dados, independentemente das condições do canal, e tem requisitos menos restritos relativos a atraso e perda de pacotes. Estabelecemos dois requisitos principais para o *slice* eMBB, descritos abaixo.

Taxa de transferência $r_{\mathrm{embb}}(n)$: a taxa de transferência para o *slice* eMBB deve satisfazer ou exceder um limite mínimo especificado, denotado como $r_{\mathrm{embb}}^{\mathrm{req}}$.

Atraso de *buffer* $d_{\mathrm{embb}}(n)$: o atraso do *buffer* para o *slice* eMBB precisa ser mantido abaixo de um limite de atraso do *buffer* predeterminado, denominado como $d_{\mathrm{embb}}^{\mathrm{req}}$.

A taxa de transferência requisitada pelos UEs associados com o *slice* eMBB é denotada como $r_{\mathrm{u}}^{\mathrm{rqt}}(n)$ seguindo uma distribuição de Poisson com um valor médio de μ_{embb}.

2) slice URLLC: Os UEs designados para um *slice* URLLC priorizam a latência mínima e a comunicação altamente confiável, geralmente caracterizada por uma taxa de perda de pacotes extremamente baixa. Os *slices* URLLC geralmente servem aplicações para missões críticas, necessitando de priorização da rede para garantir a adesão aos seus requisitos específicos. As métricas avaliadas para URLLC são similares às usadas para o *slice* eMBB e estão descritas abaixo.

Taxa de transferência $r_{\mathrm{urllc}}(n)$: representa a taxa de transferência alcançada para o *slice* URLLC. Além disso, o $r_{\mathrm{urllc}}^{\mathrm{req}}$ denota a taxa de transferência alvo que o *slice* URLLC visa alcançar.

Atraso do *buffer* $d_{\mathrm{urllc}}(n)$: representa o atraso sofrido no *buffer* do *slice* URLLC. Além disso, $d_{\mathrm{urllc}}^{\mathrm{req}}$ denota o atraso de *buffer* desejado para o tráfego URLLC.

A taxa de transferência requisitada $r_{\mathrm{u}}^{\mathrm{rqt}}(n)$ dos UEs u URLLC é definida como uma distribuição de Poisson com média μ_{urllc}.

3) slice mMTC: O *slice* mMTC é projetado para facilitar conexões para um grande número de UEs. Para o mMTC, consideramos que o atraso do *buffer*, representado como $d_{\mathrm{mmtc}}(n)$, deve ser mantido igual ou abaixo de um limite especificado de atraso do *buffer*, denominado $d_{\mathrm{mmtc}}^{\mathrm{req}}$. Além disso, no *slice* mMTC, os UEs têm um padrão de ativação probabilístico. Em cada passo de simulação, há uma probabilidade de 50% de que os UEs mMTC sejam ativados ou desativados. A taxa de transferência requisitada para o UE u do *slice* mMTC $r_{\mathrm{u}}^{\mathrm{rqt}}(n)$ é definida como uma distribuição de Poisson com média μ_{mmtc} se o UE for ativado ou zero caso contrário.

Devido às flutuações nos canais dos UEs, pode haver casos em que os recursos de rádio da rede disponíveis são insuficientes para atender a todos os requisitos dos *slices* ao longo de toda a simulação. Portanto, dada a natureza crítica das aplicações URLLC, torna-se crucial priorizar seus requisitos acima dos requisitos das outras aplicações.

III. Escalonador MARL proposto

Nesta seção, propomos um escalonador MARL utilizando o método SAC para desempenhar as funções de escalonamento de *inter-slice* e *intra-slice*. Nós estendemos o agente proposto em [7] para dar suporte à utilização de RL nas funções do escalonador *inter-slice* e *intra-slice*, com o intuito de melhorar a proteção aos *slices* críticos em cenários industriais através da priorização da alocação de recursos para esses *slices* quando a quantidade de recursos não é suficiente para atender a todos os requisitos.

O agente de *inter-slice* é o responsável por definir a quantidade de RBs para cada um dos slices, enquanto os agentes de *intra-slice* são responsáveis por alocar os RBs disponibilizados pelo agente de *intra-slice* para os UEs. O agente de *inter-slice* implementa o método de RL SAC [11] tendo uma política própria e não compartilhada com outros agentes. Os agentes de *intra-slice* utilizam o compartilhamento de parâmetros com o método SAC permitindo que uma única política seja utilizada para todos os agentes de *intra-slice*.

A. Espaços de observação

Os espaços de observação representam as entradas dos métodos para os agentes de *inter-slice* e *intra-slice* tendo diferentes composições para cada tipo de agente de escalonamento. O espaço de observação do agente de *inter-slice* no passo de simulação n é definido como

$$\boldsymbol{O}_n^{\mathrm{inter}} = [\boldsymbol{q}_{\mathrm{embb}}, \boldsymbol{q}_{\mathrm{urllc}}, \boldsymbol{q}_{\mathrm{mmtc}}, \boldsymbol{s}_{\mathrm{embb}}, \boldsymbol{s}_{\mathrm{urllc}}, \boldsymbol{s}_{\mathrm{mmtc}}], \tag{10}$$

onde $\boldsymbol{q}_s = [r_{\mathrm{s}}^{\mathrm{req}}, d_{\mathrm{s}}^{\mathrm{req}}, U_s]$ representa os requisitos do *slice* para a taxa de transferência, latência e a quantidade de UEs associados ao *slice*. As métricas de taxa de transferência, ocupação de *buffer* e *atraso* no *buffer* para cada *slice* são representadas em $\boldsymbol{s}_s = [r_s(n), b_s^{\mathrm{occ}}(n), d_s(n)]$. Essas métricas de *slice* são os valores médios dos valores obtidos dos UEs associados ao *slice* s.

O espaço de observação para o agente de *intra-slice* no passo de simulação n é

$$\boldsymbol{O}_{n,s}^{\mathrm{intra}} = [R_s(n), \boldsymbol{q}_s, \boldsymbol{s}_s], \tag{11}$$

onde, além dos requisitos do slice analisado e as suas métricas, também é incluída a quantidade de RBs alocados pelo agente de *inter-slice* para o agente de *intra-slice* em questão.

B. Espaços de ação

O espaço de ação do agente de *inter-slice* no passo de simulação n é definido de maneira similar a [7] como $\boldsymbol{A}_n^{\mathrm{inter}} = [a_{\mathrm{embb}}^{\mathrm{inter}}, a_{\mathrm{urllc}}^{\mathrm{inter}}, a_{\mathrm{mmtc}}^{\mathrm{inter}}]$, onde a_s representa um fator de ação para o *slice* s com valor no intervalo $[-1, 1]$. A ação escolhida pelo agente $\boldsymbol{A}_n$ é mapeada para o número de RBs a serem alocados para cada slice usando

$$\boldsymbol{A}_n^{\mathrm{RB}} = \Big\lfloor \frac{R(\boldsymbol{A}_n + 1)}{\sum_{s \in S}(a_s + 1)} \Big\rfloor. \tag{12}$$

A ação gerada pelo agente de *inter-slice* define quantos RBs serão utilizados por cada *slice*. O agente de intra-slice tem o seu espaço de ação representado por uma única variável a_s^{inter} com valores inteiros no intervalo de $[0, 2]$, onde cada valor representa uma escolha de algoritmo entre as três opções *round-robin*, *maximum-throughput* e *proportional-fair*, que serão responsáveis por alocar os RBs para os UEs.

C. Funções de recompensa

De forma similar a [7], a função de recompensa $\mathsf{W}(n)$ considera a distância para cumprir os requisitos de cada *slice* com base na sua construção em conjunto com o uso de mecanismo de proteção ao *slice* prioritário, em que as recompensas dos *slices* de eMBB e URLLC são contabilizadas apenas quando os requisitos do *slice* de URLLC são cumpridos. A função de recompensa do agente de *inter-slice* é

$$\mathsf{W}^{\text{inter}}(n) = \begin{cases} \mathsf{W}_{\text{embb}}(n) + \mathsf{W}_{\text{mmtc}}(n), \text{if } \mathsf{W}_{\text{urllc}}(n) = 0 \\ \mathsf{W}_{\text{urllc}}(n), \text{ caso contrário} \end{cases}, \tag{13}$$

onde $\mathsf{W}_{\text{embb}}(n)$, $\mathsf{W}_{\text{urllc}}(n)$ e $\mathsf{W}_{\text{mmtc}}(n)$ representam a função de recompensa para os *slices* de eMBB, URLLC e mMTC no passo de simulação n, respectivamente. Dessa forma, o agente de *inter-slice* tem como prioridade atender aos requisitos do *slice* prioritário de URLLC e, após isso, atender aos requisitos dos *slices* restantes.

O cálculo das contribuições da recompensa de cada *slice* é baseado nos requisitos de taxa de transferência e atraso do *buffer*:

$$\mathsf{W}_{\text{embb}}(n) = -(w^r_{\text{embb}}\mathsf{W}^r_{\text{embb}}(n) + w^d_{\text{embb}}\mathsf{W}^d_{\text{embb}}(n)), \tag{14}$$

$$\mathsf{W}_{\text{urllc}}(n) = -(w^r_{\text{urllc}}\mathsf{W}^r_{\text{urllc}}(n) + w^r_{\text{urllc}}\mathsf{W}^d_{\text{urllc}}(n)), \tag{15}$$

$$\mathsf{W}_{\text{mmtc}}(n) = -w^d_{\text{mmtc}}\mathsf{W}^d_{\text{mmtc}}(n), \tag{16}$$

onde $\mathsf{W}^r_{s\in S}(n)$ e $\mathsf{W}^d_{s\in S}(n))$ representam as contribuições da taxa de transferência e o atraso do *buffer* no cálculo da recompensa. Os pesos w podem definir a prioridade das métricas de cada *slice* e também podem ser utilizados para normalizar os valores da função de recompensa. A contribuição da taxa de transferência é definida como

$$\mathsf{W}^r_s(n) = \begin{cases} \frac{r^{\text{req}}_{\text{s}} - r_{\text{s}}(n)}{r^{\text{req}}_{\text{s}}}, \text{ if } r_{\text{s}}(n) < r^{\text{req}}_{\text{s}} \\ 0, \qquad\qquad \text{if } r_{\text{s}}(n) \geq r^{\text{req}}_{\text{s}} \end{cases}, \tag{17}$$

e a contribuição do atraso no *buffer* é

$$\mathsf{W}^d_s(n) = \begin{cases} \frac{d_{\text{s}}(n) - d^{\text{req}}_{\text{s}}}{d_{\max} - d^{\text{req}}_{\text{s}}}, \text{ if } d_{\text{s}}(n) > d^{\text{req}}_{\text{s}} \\ 0, \qquad\qquad \text{if } d_{\text{s}}(n) \leq d^{\text{req}}_{\text{s}} \end{cases}. \tag{18}$$

A função de recompensa do agente de *intra-slice* é

$$\mathsf{W}^{\text{intra}}_s(n) = \mathsf{W}^r_s(n) + \mathsf{W}^d_s(n), \tag{19}$$

onde as distâncias para atingir os requisitos das taxa de transferência e do atraso do *buffer* são contabilizadas pelo agente que busca minimizar essas distâncias através da seleção do melhor algoritmo entre as opções *round-robin*, *maximum-throughput* e *proportional-fair* para cada passo de simulação n. No caso do *slice* de mMTC, o valor da contribuição da taxa de transferência no $\mathsf{W}^r_s(n)$ é sempre considerado zero, dado que o *slice* de mMTC não possui requisitos de taxa de transferência.

IV. Resultados Numéricos

Na Tabela I, encontram-se os principais parâmetros da rede de comunicação empregada na simulação de canal utilizando o simulador QuaDRiGa [12]. Utilizamos MARL com SAC para o escalonador de recursos de rádio utilizando a biblioteca Ray Rllib [13], tendo um agente de *inter-slice* e três agentes de *intra-slice* responsáveis por escolher entre os algoritmos de *round-robin*, *maximum-throughput* e *proportional-fair*. O método de *baseline* SLA *Satisfaction Rate* (SSR) usado para comparação está descrito em [7].

TABELA I: Parâmetros de simulação.

Parâmetro	**Valor**
Frequência da portadora	6 GHz
Número de RBs	100
Subcarrier spacing	15 kHz
Duração do TTI	1 ms
Potência de transmissão	35 dBm
Velocidade dos UEs	3 km/h
Número de UEs	100
Número de *slices*	3
Rodadas de simulação	100
TTI	1000
req. de taxa de transf.	$r^{\text{req}}_{\text{embb}}$=20, $r^{\text{req}}_{\text{urllc}}$=5 Mbps
req. de latência	$d^{\text{req}}_{\text{embb}} = 30$, $d^{\text{req}}_{\text{urllc}} = 1$, $d^{\text{req}}_{\text{mmtc}} = 50$ ms
Média Poisson	$\mu_{\text{embb}} = 20$, $\mu_{\text{urllc}} = 5$, $\mu_{\text{mmtc}} = 0.1$ Mbps
Peso das métricas	$w^r_{\text{embb}} = 0.5$, $w^d_{\text{embb}} = 0.3$, $w^r_{\text{urllc}} = 0.5$, $w^d_{\text{urllc}} = 0.5$, $w^r_{\text{mmtc}} = 0.2$

O cenário considerado neste artigo é um galpão de fábrica, com uma BS equipada com múltiplas antenas e posicionada no canto superior. A BS atende UEs em três casos de uso distintos: eMBB, URLLC e mMTC. Para aumentar o realismo do cenário, nós baseamos a simulação em medições reais feitas em um galpão de fábrica localizado em Nuremberg, Alemanha, que extraímos de [9]. Neste cenário, a BS fornece serviço para um total de 100 UEs, distribuídos uniformemente por todo o galpão da fábrica. Dentre esses UEs, 30 estão associados com eMBB, 40 com URLLC e 30 com aplicações mMTC. Consideramos 100 RBs para distribuição entre os *slices* e UEs, utilizando tanto escalonamento *inter-slice* quanto *intra-slice*, com potência alocada uniformemente entre os RBs. Além disso, assume-se que os UEs estão em movimento a uma velocidade de 3 km/h. O sistema apresenta três *slices*, sendo uma dedicada ao eMBB, uma ao URLLC e uma ao mMTC. Por fim, dividimos as 100 rodadas de simulação produzidas pelo QuaDRiGa em dois conjuntos distintos: um conjunto de treinamento e um conjunto de teste. O conjunto de treinamento compreende 70 rodadas, enquanto o conjunto de teste consiste em 30 rodadas.

A Figura 1 mostra os resultados para as taxas de transferência médias para cada slice. Considerando os requisitos para taxas de transferência $r^{\text{req}}_{\text{embb}} = 20$ e $r^{\text{req}}_{\text{urllc}} = 5$ Mbps, o agente proposto forneceu uma taxa de transferência maior para o slice de URLLC e uma taxa menor para o eMBB quando comparado com o método SSR. Esse comportamento é justificado pois o método proposto primeiramente atende os requisitos do *slice* prioritário e, posteriormente, aloca os recursos restantes para o *slice* de eMBB. Tanto o método proposto quanto o método SSR não conseguiram cumprir o

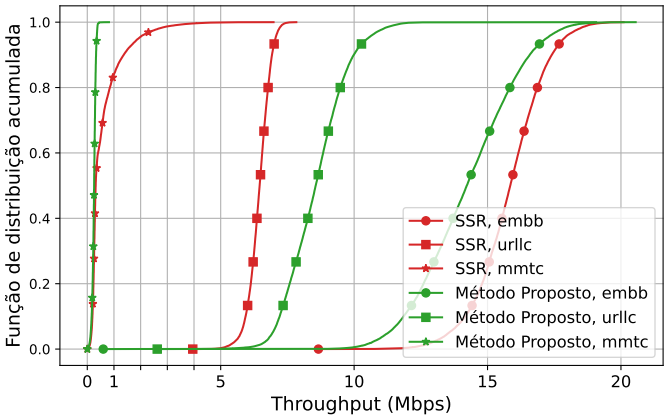

Fig. 1: Taxas de transferência média obtidas por cada *slice*.

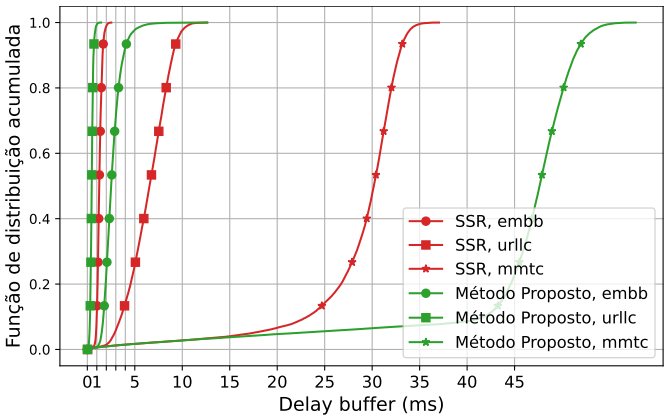

Fig. 2: Atraso médio do *buffer* experienciados por cada *slice*.

requisito de taxa de transferência para o *slice* de eMBB, pois a quantidade de recursos disponíveis na rede não era suficiente para atender a todos os requisitos.

Quando se avalia o atraso no *buffer*, a Figura 2 mostra que o método proposto conseguiu manter o atraso do *slice* de URLLC abaixo do requisito de $d^{\text{req}}_{\text{urllc}} = 1$ ms, enquanto o método SSR manteve atraso acima de 1 ms por pelo menos 50% dos passos de simulação. Ambos os métodos cumpriram o requisito do *slice* eMBB $d^{\text{req}}_{\text{embb}} = 30$ ms. O método proposto conseguiu cumprir os requisitos de latência para o mMTC de $d^{\text{req}}_{\text{mmtc}} = 50$ ms durante 80% do período da simulação, enquanto o método de SSR conseguiu cumprir durante toda a simulação.

Por fim, a Figura 3 mostra os valores médios de violações de requisitos totais e para o *slice* prioritário URLLC em 30 episódios de teste. O método proposto apresenta um menor número de violações considerando-se todos os *slices* e também o *slice* prioritário. O número de violações para o slice prioritário URLLC mantém valores próximos a zero durante toda a simulação e com um pequeno desvio padrão se comparado aos valores obtidos para o URLLC pelo SSR.

V. Conclusão

Nós apresentamos um método para alocação de recurso de rádio utilizando MARL executando as funções de escalonamento *inter-slice* e *intra-slice* focados em cenários da

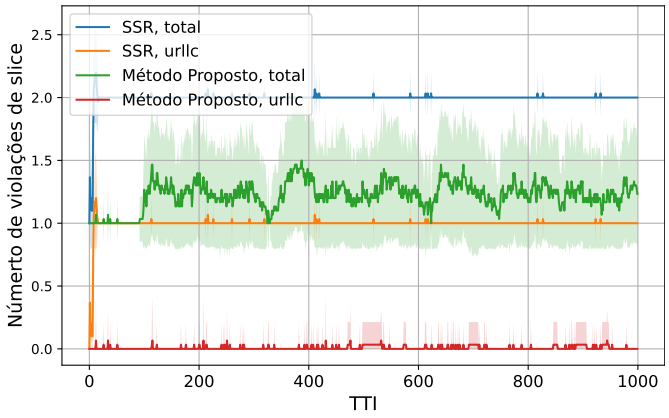

Fig. 3: Média de violações e desvio padrão nos dados de teste.

Indústria 4.0. O agente proposto cumpre os requisitos dos *slices* através da combinação de ações do escalonador de *inter-slice* e *intra-slice*, além de priorizar os *slices* de URLLC quando os recursos disponíveis na rede não são suficientes para atender os requisitos de todos os *slices*. Os resultados preliminares mostram que o método proposto obteve um número de violações significativamente menor em relação ao método de SSR nos *slices* gerais e prioritários.

Referências

[1] Y. Liu *et al.*, "Network slicing for eMBB, URLLC, and mMTC: An uplink rate-splitting multiple access approach," *IEEE Transactions on Wireless Communications*, vol. 23, no. 3, pp. 2140–2152, Mar. 2024.

[2] F. Debbabi *et al.*, "An overview of interslice and intraslice resource allocation in B5G telecommunication networks," *IEEE Transactions on Network and Service Management*, vol. 19, no. 4, pp. 5120–5132, Dec. 2022.

[3] M. Zangooei *et al.*, "Reinforcement learning for radio resource management in RAN slicing: A survey," *IEEE Communications Magazine*, vol. 61, no. 2, pp. 118–124, Feb. 2023.

[4] Y. Shao *et al.*, "Graph attention network-based multi-agent reinforcement learning for slicing resource management in dense cellular network," *IEEE Transactions on Vehicular Technology*, vol. 70, no. 10, pp. 10 792–10 803, Oct. 2021.

[5] M. Zangooei *et al.*, "Flexible RAN slicing in Open RAN with constrained multi-agent reinforcement learning," *IEEE Journal on Selected Areas in Communications*, vol. 42, no. 2, pp. 280–294, Feb. 2024.

[6] H. Anıl Akyıldız *et al.*, "Hierarchical reinforcement learning based resource allocation for RAN slicing," *IEEE Access*, vol. 12, pp. 75 818–75 831, May 2024.

[7] C. Nahum *et al.*, "Safeguard URLLC services in Industry 4.0 through reinforcement learning scheduling," in *2023 Workshop on Communication Networks and Power Systems (WCNPS)*. IEEE, 2023, pp. 1–7.

[8] 3GPP, "TS 38.873 v12.5.0," *Study on 3D channel model for LTE*, 2017.

[9] S. Jaeckel *et al.*, "Industrial indoor measurements from 2-6 GHz for the 3GPP-NR and QuaDRiGa channel model," in *IEEE 90th Vehicular Technology Conference*, 2019, pp. 1–7.

[10] C. V. Nahum *et al.*, "Intent-aware radio resource scheduling in a RAN slicing scenario using reinforcement learning," *IEEE Transactions on Wireless Communications*, vol. 23, no. 3, pp. 2253–2267, Mar. 2024.

[11] T. Haarnoja *et al.*, "Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor," in *Proc. of International conference on machine learning*. PMLR, 2018, pp. 1861–1870.

[12] S. Jaeckel *et al.*, "QuaDRiGa - quasi deterministic radio channel generator, user manual and documentations," 2021.

[13] E. Liang *et al.*, "Ray rllib: A composable and scalable reinforcement learning library," *arXiv preprint arXiv:1712.09381*, vol. 85, p. 245, 2017.