SLOrion: Democratização das Redes 6G através de Contratos Inteligentes e Fatiamento de Rede como Serviço

Ariel Galante Dalla-Costa[1], **Abel Lisboa**[1],
Antônio Marcos Alberti[2], **Cristiano Bonato Both**[1]

[1] Universidade do Vale do Rio dos Sinos (UNISINOS), São Leopoldo – Brasil
[2]University of Leeds, Leeds – United Kingdom

{costaariel, abell}@edu.unisinos.br; a.m.alberti@leeds.ac.uk; cbboth@unisinos.br

Resumo. *A centralização e o monopólio global de operadoras de redes móveis representam barreiras significativas para a evolução das redes 6G, limitando a flexibilidade e a interoperabilidade necessárias para atender a serviços avançados de próxima geração. O conceito de fatiamento de rede surge como uma abordagem promissora para superar esses desafios, permitindo a criação de fatias configuráveis para diferentes requisitos. Este artigo apresenta SLOrion, uma solução descentralizada baseada em contratos inteligentes e Distributed Ledger Technology. A proposta automatiza a tradução de Service Level Agreements em parâmetros técnicos para fatiamento de rede como serviço, reduzindo a intervenção manual, promovendo a interoperabilidade e assegurando a integridade das configurações de rede. SLOrion endereça lacunas existentes na literatura ao conectar contratos inteligentes diretamente às configurações técnicas de redes móveis, contribuindo para a democratização das redes 6G. Os resultados demonstram que o SLOrion mantém mais de 90% de precisão e um tempo de resposta até mil vezes menor que o modelo da OpenAI, garantindo escalabilidade para redes 5G e 6G.*

Abstract. *The centralization and global monopoly of mobile network operators pose significant barriers to the advancement of 6G networks, limiting the flexibility and interoperability required to meet next-generation advanced service demands. The concept of network slicing emerges as a promising approach to address these challenges, enabling the creation of configurable slices tailored to different requirements. This paper introduces SLOrion, a decentralized solution based on smart contracts and Distributed Ledger Technology. The proposed solution automates the translation of Service Level Agreements into technical parameters for network slicing as a service, reducing manual intervention, promoting interoperability, and ensuring the integrity of network configurations. SLOrion addresses existing gaps in the literature by directly linking smart contracts to the technical configurations of mobile networks, contributing to the democratization of 6G networks. The results show that SLOrion maintains over 90% accuracy and a response time up to a thousand times faster than the OpenAI model, ensuring scalability for 5G and 6G networks.*

1. Introdução

O mercado digital desempenha um papel cada vez mais relevante na democratização das tecnologias emergentes, incluindo as redes móveis além da quinta geração (*beyond*

5G (B5G)) e as futuras redes 6G. Entretanto, o cenário atual é marcado por um monopólio global das operadoras de redes celulares, onde contratos de *Service Level Agreement*s (SLAs) são negociados manualmente e de forma centralizada [Moreno et al. 2024]. Essa abordagem tradicional dificulta a flexibilidade e a interoperabilidade necessárias para atender as demandas cada vez mais complexas, especialmente em cenários que exigem serviços avançados como *enhanced Mobile Broadband* (eMBB), *Ultra Reliable Low Latency Communications* (URLLC) e *massive Machine Type Communications* (mMTC) [Dai et al. 2025].

O conceito de fatiamento de rede surge como uma solução promissora nesse contexto [Cheng et al. 2024]. Ele permite que as redes móveis sejam particionadas logicamente em várias fatias, cada uma configurada para atender a requisitos específicos de desempenho, como largura de banda, latência ou confiabilidade. Esse particionamento possibilita a entrega de serviços sob medida para diferentes tipos de aplicações e usuários. Nesse contexto, a *Global System for Mobile Communication Association* (GSMA) define modelos genéricos de fatias de rede para a solicitação de fatias como serviços [GSMA 2024]. No entanto, a implementação efetiva de fatiamento de rede exige um gerenciamento inteligente, dinâmico e automatizado de SLAs, garantindo que os *Service Level Objective*s (SLOs), definidos como um conjunto de *Key Performance Indicator*ss (KPIs), sejam atendidos em cada fatia [Muntaha et al. 2025].

A Figura 1 ilustra, em alto nível, o fluxo desde a solicitação dos SLOs até a materialização do fatiamento de rede. Inicialmente, os usuários requisitam suas intenções às operadoras, que as traduzem em SLAs. Esses contratos são negociados entre diferentes operadoras, verificando a disponibilidade de recursos e acordos financeiros. Após o aceite dos contratos, os requisitos técnicos para o fatiamento de rede são identificados, considerando as infraestruturas das operadoras. Finalmente, as fatias de rede são implantadas com base nos SLAs, incluindo telemetria e monitoramento para evitar violações dos KPIs negociados.

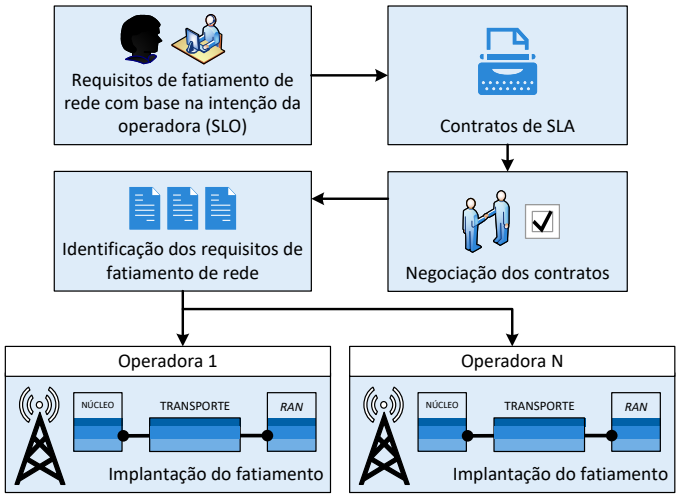

Figura 1. Fluxo de negociação de SLAs para fatiamento de rede.

Apesar dos avanços recentes, as abordagens atuais ainda enfrentam desafios significativos. O processo de tradução das intenções de SLOs para parâmetros técnicos de fatiamento de rede continua sendo um problema crítico, devido à ambiguidade e à he-

terogeneidade das infraestruturas modernas. Além disso, a centralização dos processos limita a eficiência, escalabilidade e transparência, como evidenciado em estudos como [Samdanis et al. 2016], que propõe uma arquitetura de *5G Network Slice Broker* para gerenciar recursos de rede de forma dinâmica, mas permanece dependente de uma abordagem centralizada. Trabalhos como [Yang and Lei 2019] introduzem interpretadores formais para contratos inteligentes, mas se concentram na verificação de segurança e consistência, sem aplicar esses conceitos ao fatiamento de rede. Estudos mais recentes, como [Upadhyay et al. 2021], exploram contratos inteligentes para substituir SLAs ambíguos por cláusulas autoexecutáveis, enquanto [Gonçalves et al. 2022] propõem uma solução baseada em *Blockchain* para verificar a integridade das configurações de fatias de rede. Entretanto, essas abordagens não investigam a tradução programática de parâmetros técnicos. Além disso, [Javed et al. 2022, Javed et al. 2023] analisam o uso de tecnologias de registro distribuído para negociações automatizadas, mas não detalham a interpretação de parâmetros específicos de fatias. A ausência de um mecanismo de tradução robusto entre linguagens naturais, utilizadas na descrição de SLAs, e as linguagens de baixo nível, necessárias para a implementação técnica do fatiamento de rede em diferentes domínios administrativos, impõe às operadoras a necessidade de um conhecimento detalhado e exaustivo dos parâmetros técnicos de cada infraestrutura. Este trabalho preenche essas lacunas ao propor um interpretador especializado que conecta contratos inteligentes às demandas específicas de redes B5G, promovendo interoperabilidade e eficiência no gerenciamento de fatias de rede.

Para superar as limitações da literatura e avançar rumo à democratização das redes 6G, este trabalho apresenta o **SLOrion**: uma abordagem descentralizada que utiliza contratos inteligentes, *Blockchain* e *Distributed Ledger Technology* (DLT). SLOrion automatiza a tradução de SLAs em parâmetros técnicos para uma plataforma de fatiamento de rede como serviço, promovendo interoperabilidade entre operadoras, reduzindo a necessidade de intervenção manual e garantindo maior integridade e transparência nos processos. As principais contribuições deste artigo são:

- Uma abordagem descentralizada para tradução de SLAs em fatiamento de rede utilizando tecnologias como contratos inteligentes e *Blockchain*;
- Um mecanismo automatizado para negociação, identificação e implantação de SLOs com base em intenções dos usuários;
- Garantia de interoperabilidade entre operadoras com infraestruturas heterogêneas através do uso de DLT;
- Redução da sobrecarga manual nos processos de configuração de redes, otimizando a eficiência operacional, através da integração do SLOrion a uma plataforma de fatiamento de rede como serviço.

Este trabalho contribui para a construção de um ecossistema transparente e democrático, alinhado às necessidades de inovação e escalabilidade das redes futuras. Os resultados demonstram que SLOrion é escalável para garantir os serviços de redes B5G, mantendo uma taxa média de acerto acima dos 90% e um tempo de resposta três ordens de grandeza menor que o modelo de identificação da OpenAI, utilizado como comparação.

O restante deste artigo está organizado da seguinte maneira: Na Seção 2 é descrita a fundamentação teórica. Na Seção 3 são discutidos os trabalhos relacionados. Na Seção 4 é apresentada a arquitetura e as técnicas de decodificação do SLOrion. Na Seção 5, é detalhada a implementação do algoritmo, com avaliação e comparação de desempenho. Finalmente, na Seção 6 são apresentadas as conclusões e os trabalhos futuros.

2. Fundamentação Teórica

Esta seção apresentada os principais conceitos e terminologias relacionados à SLAs. Além disso, a seção discute as funções, serviços e modelo de referência definidos pela *3rd Generation Partnership Project* (3GPP) para análise de dados que podem ser utilizados no contexto de SLAs inteligentes, dinâmicos e automatizados.

2.1. Contrato de níveis de serviços

SLA é um acordo formal que estabelece os níveis de serviço que um provedor se compromete a oferecer a um cliente, onde esse acordo define as métricas, responsabilidades, garantias, penalidades e outros aspectos relacionados à prestação de serviços na relação de consumo [Kapassa et al. 2018]. Na prática, um SLA atua como um contrato entre provedor e consumidores. Desta forma, o SLO é uma declaração clara e mensurável das metas de desempenho ou *Qualidade de Serviço* (QoS) que um provedor se compromete a alcançar, i.e., monitorando periodicamente dados da sua infraestrutura e serviços [Touloupou et al. 2019].

Os *SLA Descriptor*s (SLADs) referem-se à descrição detalhada dos elementos e parâmetros que compõem um SLA, incluindo os SLOs e outras especificações de desempenho [Touloupou et al. 2019]. Além disso, os modelos de SLAs são exemplos pré-definidos que podem ser usados como base para a criação de um SLA personalizado. Os modelos geralmente incluem seções e cláusulas padrões que são comuns a muitos SLAs, como a definição dos serviços a serem prestados, os padrões de desempenho esperados, as responsabilidades do provedor de serviços e do cliente, os procedimentos para monitoramento, relatórios e resolução de problemas, entre outros [Touloupou et al. 2019].

No contexto de redes B5G, a negociação de serviços deve ser dinâmica, i.e., os SLAs precisam se adaptar à essas características de negociação flexível, utilizando KPIs de diferentes origens para fornecer um serviço inteligente. Para tal, faz-se necessário que o SLA tenha dinamicidade para garantir a conformidade com as metas de desempenho e os requisitos de QoS em ambientes B5G, onde as condições da rede e as demandas dos usuários podem variar dinamicamente. Essas características também são essenciais para garantir a adaptação contínua das metas de desempenho e dos parâmetros de QoS, ambas com base nas condições em tempo real da rede, nas necessidades dos usuários e nas demandas de tráfego. Isso permite uma resposta ágil a mudanças nas condições operacionais, garantindo uma experiência consistente e de alta qualidade para os usuários finais. Além disso, um SLA com características dinâmicas é vital para garantir a conformidade com as expectativas dos usuários e para otimizar o desempenho da rede, permitindo ajustes automáticos com base em análises em tempo real e em dados operacionais [Muntaha et al. 2025].

Como SLAs dinâmicos em redes B5G necessitam de uma maior flexibilidade, adaptabilidade e mudanças rápidas, algumas propostas para automatizar as negociações de níveis de serviço surgiram utilizando DLT, uma tecnologia de contabilidade distribuída que incorpora diferentes recursos para oferecer suporte de negociação confiável, inteligente e automatizada [Dai et al. 2025]. Na próxima subseção são apresentados os principais conceitos de DLT.

2.2. DLT, *Blockchain* e Contratos Inteligentes

A DLT surgiu como uma solução inovadora para a gestão de informações em redes sem a dependência de uma autoridade central. Por meio de algoritmos de consenso, vários

participantes compartilham e validam dados de forma colaborativa, tornando o registro resistente a falhas e fraudes. Essa abordagem reforça a confiabilidade das transações e distribui a responsabilidade pela verificação entre todos os nós da rede. Um dos principais benefícios da DLT é a redução de custos operacionais, uma vez que dispensa intermediários para a validação das informações. Além disso, a transparência inerente ao modelo de registro distribuído possibilita maior eficiência em processos de auditoria, garantindo um histórico completo e imutável das transações. Esse potencial tem incentivado diversos setores, como finanças, logística e saúde, a explorar novos casos de uso, promovendo inovação em produtos e serviços [Zheng et al. 2018].

Dentro desse universo, *Blockchain* aparece como uma das implementações mais conhecidas de DLT, ao estruturar blocos criptografados encadeados que garantem a imutabilidade do histórico de transações. Em *Blockchain*, podem ser desenvolvidos como contratos inteligentes, que são programas que atuam em plataformas de *Blockchain* e têm a capacidade de executar automaticamente termos pré-definidos de um acordo entre partes. A Figura 2 apresenta a integração do uso de DLTs, em diferentes nós *Blockchain* para negociar contratos inteligentes para o fatiamento de rede como serviço entre operadoras de redes móveis. A lógica desses contratos é estabelecida em linhas de código, de modo que, uma vez cumpridas as condições estabelecidas, a execução se torna inevitável. Essa automação elimina a necessidade de intermediários para monitorar e garantir que as partes cumpram suas obrigações, reduzindo custos e prazos de validação de transações [Bellaj et al. 2024].

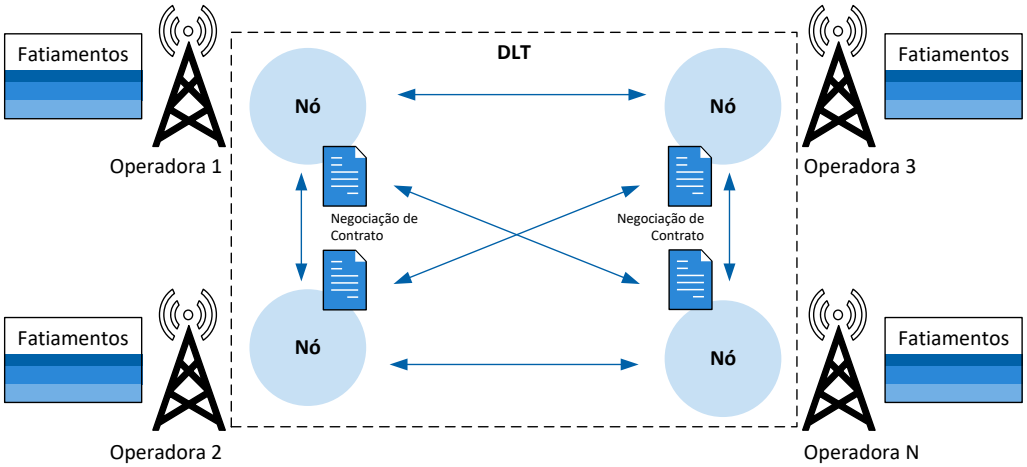

Figura 2. DLT, *Blockchain* e contratos inteligentes para fatiamento de rede.

A flexibilidade dos contratos inteligentes permite que um vasto conjunto de operações seja automatizado, como pagamentos, transferências de ativos digitais e até mesmo processos mais complexos de governança. Por serem executados de forma descentralizada, esses contratos operam sem qualquer ponto único de falha ou controle, trazendo uma camada adicional de robustez às transações. Isso é particularmente vantajoso em ambientes onde a confiança entre as partes é limitada, pois o código do contrato determina todo o fluxo de execução, minimizando a necessidade de supervisão externa [Singh et al. 2024].

Um dos grandes diferenciais dos contratos inteligentes é a segurança proporcionada pelo ambiente de *Blockchain*. Como as informações são registradas e verificadas de forma distribuída em diversos nós da rede, torna-se muito mais difícil adulterar ou inter-

romper a execução do contrato. Além disso, o histórico das transações e execuções contratuais fica gravado de forma imutável, permitindo maior auditabilidade e transparência [Zheng et al. 2018]. Essa combinação de automação, segurança e rastreabilidade abre espaço para inúmeras aplicações em áreas que envolvem acordos e transações de valor, como por exemplo, as redes B5G, beneficiando-as com maior eficiência, confiabilidade e redução de fraudes.

Essas características dos contratos inteligentes, i.e., automação, segurança e rastreabilidade, têm despertado grande interesse na literatura, especialmente em cenários que exigem alta confiabilidade e transparência, como as redes B5G. Diversos trabalhos relacionados têm investigado como essas tecnologias podem ser aplicadas para aprimorar a eficiência, reduzir fraudes e garantir maior auditabilidade em ambientes distribuídos. Na próxima seção, são discutidos os principais avanços e abordagens propostas nessa área.

3. Trabalhos Relacionados

O desenvolvimento de um interpretador para contratos inteligentes, com foco na interpretação de parâmetros de fatias de rede, apresenta lacunas não exploradas nos trabalhos existentes. Embora haja diversas contribuições relevantes em áreas correlatas, nenhuma atende de maneira específica a proposta de um interpretador voltado para a tradução de parâmetros de fatias de rede. A Tabela 1 sumariza os trabalhos relacionados, destacando as características principais de cada abordagem em termos de tipo de arquitetura, uso de contratos inteligentes, suporte a SLAs, suporte a fatiamento de rede e interpretação de contratos inteligentes para fatiamento de rede. Cada uma dessas características foi utilizada para identificar como cada artigo apresentado na literatura aborda aspectos relevantes ao tema deste trabalho, evidenciando lacunas que essa pesquisa preenche.

Tabela 1. Trabalhos relacionados

Trabalho	Arquitetura	Contratos inteligentes	SLA	Fatiamento de rede	Interpretador de intenções de SLA
[Samdanis et al. 2016]	*Broker*	○	◐	●	○
[Yang and Lei 2019]	*Blockchain*	●	○	○	○
[Upadhyay et al. 2021]	*Blockchain*	●	●	○	○
[Gonçalves et al. 2022]	*Blockchain*	●	◐	●	○
[Javed et al. 2022]	*Blockchain*	●	●	●	○
[Javed et al. 2023]	*Blockchain*	●	●	●	○
[Bellaj et al. 2024]	DLT	●	○	○	○
[Krijnen et al. 2024]	*Blockchain*	●	○	○	○
Este trabalho	*Blockchain*	●	●	●	●

Atributos: ● Completamente atendido ◐ Parcialmente atendido ○ Não atendido

[Samdanis et al. 2016] foram os primeiros autores a introduzem o conceito do *5G Network Slice Broker*, através de uma arquitetura para gerenciar recursos de rede de forma dinâmica. Apesar de explorar a alocação de fatias de rede, o trabalho não aborda o uso de contratos inteligentes nem a interpretação de parâmetros de fatias, sugerindo para tal, uma abordagem centralizada em *Broker*. Sua proposta está voltada para gestão operacional e não para a manipulação programática dos parâmetros técnicos dessas fatias. Posteriormente, [Yang and Lei 2019] desenvolvem FEther, um interpretador formal para contratos

inteligentes, combinando execução simbólica e provas formais em Coq. Embora forneça uma base sólida para verificação de contratos, o trabalho não se aplica ao contexto de fatias de rede, nem à interpretação de parâmetros relacionados. Sua abordagem é centrada na segurança e consistência de contratos genéricos, sem considerar as especificidades de redes B5G, como por exemplo suporte a fatiamento de rede.

Ethereum é uma plataforma descentralizada capaz de executar contratos inteligentes e aplicações descentralizadas usando a tecnologia blockchain. [Upadhyay et al. 2021] exploram o uso de contratos inteligentes para substituir SLAs ambíguos por cláusulas autoexecutáveis na plataforma Ethereum[1]. Embora o trabalho seja relevante no contexto de simplificação e automação de SLAs, os autores não contemplam cenários de fatiamento de rede, nem a necessidade de interpretação específica de parâmetros técnicos. A abordagem os autores está limitada à formalização de contratos legais e não inclui a complexidade inerente às configurações de rede multi-inquilinos. De uma forma experimental, [Gonçalves et al. 2022] propõem uma solução baseada em *Blockchain* para verificar a integridade de configurações em fatias de rede. Apesar de estar alinhado ao contexto de redes B5G, o foco dos autores está relacionado a garantia de integridade por meio de oráculos, e não na interpretação de parâmetros de fatias em contratos inteligentes. Assim, o trabalho não oferece uma solução que permita compreender e manipular esses parâmetros de maneira programática e adaptável.

[Javed et al. 2022, Javed et al. 2023] analisam como tecnologias de registro distribuído podem ser aplicadas em fatias de rede, destacando a interoperabilidade e a descentralização. No entanto, o foco é mais amplo, com ênfase na automatização de negociações entre partes interessadas, e não na interpretação específica de parâmetros de fatias. Nesse contexto, os trabalhos carecem de uma abordagem detalhada sobre como processar e interpretar os parâmetros técnicos necessários para configurar fatias de rede.

[Krijnen et al. 2024] apresenta um modelo formal para certificação de tradução de contratos inteligentes, garantindo que o código-fonte seja corretamente traduzido para o código compilado. Apesar de abordar questões de confiabilidade e consistência, o trabalho não considera a interpretação de parâmetros de fatias de rede, nem o contexto de redes B5G. Sua abordagem é focada em garantir a integridade do processo de compilação e não aborda a dinâmica ou as necessidades específicas de fatias de rede, como as interações entre múltiplas partes interessadas. [Bellaj et al. 2024] apresentam outro interessante trabalho, onde os autores introduzem uma taxonomia para classificar tecnologias de DLT, organizando-as em camadas como dados, consenso, execução e aplicação. Embora útil para estruturar sistemas baseados em DLTs, a proposta não aborda diretamente o problema de interpretar parâmetros técnicos em contratos inteligentes, especialmente no contexto de fatiamento de rede. A ausência de foco em redes B5G limita sua aplicabilidade ao cenário proposto.

A falta de um mecanismo de robustez linguística obriga as operadoras a manterem um conhecimento detalhado e atualizado dos parâmetros de fatiamento, o que é inviável em cenários reais devido à constante evolução das terminologias. Este trabalho preenche essa lacuna ao desenvolver um interpretador especializado, que alia intenções de fatiamento de rede às demandas específicas de redes B5G. Para lidar com esse desafio, na próxima seção é apresentado o SLOrion, uma solução projetada para superar essas limitações e garantir maior adaptabilidade e precisão no reconhecimento de parâmetros.

[1]https://ethereum.org/

4. SLOrion

SLOrion foi projetado para ajudar no processo de democratização de redes B5G e 6G através de contratos inteligenstes e fatiamento de rede como serviço. O ambiente de funcionamento do SLOrion pode ser observado na Figura 3. Iniciando pelo lado superior esquerdo, o processo de implantação tem início com as operadoras criando contratos inteligentes. Por se tratar de um contrato que possui liberdade de negociação, as operadoras descrevem as intenções de parâmetros de fatiamento de rede em parâmetros mais abrangentes, se assemelhando à linguagem natural. Esses contratos inteligentes são negociados em uma infraestrutura de DLT, conhecida como *On-chain*. A infraestrutura *On-chain* aproveita todos os recursos e garantias oferecidas pela DLT, ao custo de ser fechada e não possuir acesso à informações do mundo externo de forma nativa. Para suprir tal necessidade de comunicação com o mundo externo, surgiram os oráculos, que são softwares que possuem a capacidade de interagir com os contratos inteligentes da DLT, bem como, dados e informações do mundo externo, conhecidos como *Off-chain*. O oráculo pode executar tarefas e consumir serviços, como por exemplo requisições *Representational State Transfer* (REST) à serviços da infraestrutura das operadoras, a qual não seria possível nativamente na DLT, por se tratar de um ciclo fechado. Essa maleabilidade é fornecida através de *Jobs*, que são descritores de tarefas que se comunicam com os protocolos externos. Neste contexto, o oráculo requisita ao SLOrion a identificação das intenções expressas em textos, palavras-chave e comentários, e o SLOrion cria, identifica e traduz as intenções descritas para os parâmetros técnicos de fatiamento de rede, encontrados nos *NEtwork Slice Template* (NEST). Como resultado final, a configuração efetiva da infraestrutura é implantada às cláusulas do contrato, viabilizando uma orquestração automatizada e eficiente.

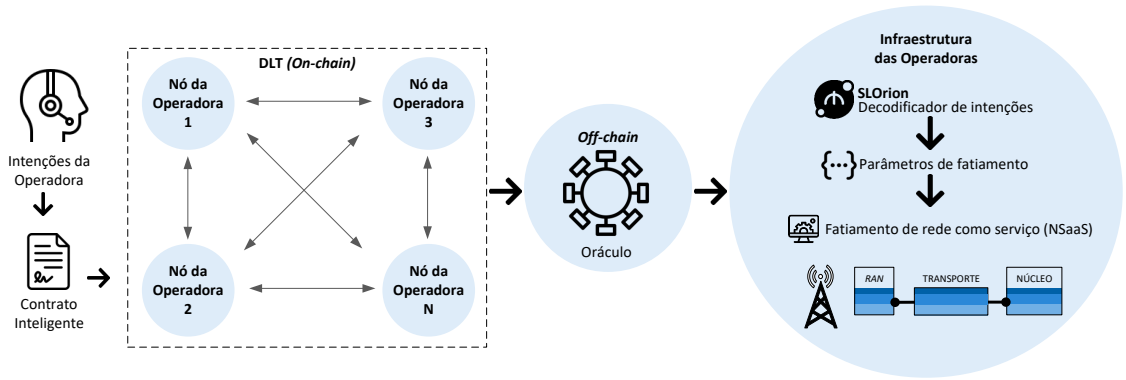

Figura 3. Ambiente de funcionamento do SLOrion

Sem um mecanismo de robustez linguística, a identificação das intenções das operadoras exigiria que elas conhecessem detalhadamente todos os nomes dos parâmetros no arquivo de fatiamento, o que é pouco viável em operações reais, onde terminologias podem mudar e surgir novas nomenclaturas ao longo do tempo. Para enfrentar esse desafio, o SLOrion emprega o coeficiente de similaridade de Dice [Dice 1945], que utiliza bigramas para medir a similaridade entre duas palavras. Bigramas são sequências de dois caracteres consecutivos extraídas de uma string, que quando comparadas, resultam em um coeficiente entre 0 e 1, onde valores mais altos indicam maior semelhança.

O Algoritmo 1 representa o funcionamento do cálculo do coeficiente de similari-

Algorithm 1 Calcula o coeficiente de Dice entre duas strings

```
1:  function CoefDice(string1, string2)
2:      string1 ← ConverterParaMaiusculas(string1)
3:      string2 ← ConverterParaMaiusculas(string2)
4:      bigramas1 ← {}
5:      bigramas2 ← {}                          ▷ Gerar bigramas para string1
6:      for i ← 1 até comprimento(string1) - 1 do
7:          bigrama ← string1[i:i + 1]
8:          Adicionar bigrama em bigramas1
9:      end for                                 ▷ Gerar bigramas para string2
10:     for i ← 1 até comprimento(string2) - 1 do
11:         bigrama ← string2[i:i + 1]
12:         Adicionar bigrama em bigramas2
13:     end for
14:     interseccao ← |bigramas1 ∩ bigramas2|
15:     total_bigramas ← |bigramas1| + |bigramas2|
16:     if total_bigramas > 0 then
17:         coefDice ← (2× interseccao)/ total_bigramas
18:     else
19:         coefDice ← 0.0
20:     end if
21:     retornar coefDice
22: end function
```

dade de Dice. As linhas 2 e 3 convertem as strings para maiúsculas, garantindo uniformidade na comparação. Em seguida, nas linhas 4 e 5, são inicializados conjuntos vazios para armazenar os bigramas de cada string. As linhas 6 a 10 percorrem a primeira string e extraem pares de caracteres consecutivos, inserindo-os no conjunto bigramas1, enquanto as linhas 11 a 15 repetem esse processo para a segunda string, armazenando os bigramas no conjunto bigramas2. Na linha 16, calcula-se a interseção entre os conjuntos de bigramas para determinar a quantidade de bigramas em comum, e, na linha 17, obtém-se o total de bigramas das duas strings. Por fim, nas linhas 18 a 21, verifica-se se há bigramas antes de calcular o coeficiente de Dice, aplicando sua equação caso existam, ou retornando zero se não houver. O coeficiente de Dice é calculado pela Equação 1.

$$D(A, B) = \frac{2 \times |A \cap B|}{|A| + |B|} \tag{1}$$

onde, A e B são os conjuntos de bigramas das strings, $|A \cap B|$ é a quantidade de bigramas em comum. Além disso, $|A|$ e $|B|$ são os tamanhos dos conjuntos de bigramas.

SLOrion faz uma análise recursiva, comparando as intenções recebidas via texto com um modelo padrão de parâmetros fornecidos pela operadora do tipo NEST. Para tal análise, há quatro tipos de estratégias de comparação: (i) o nome completo do parâmetro, (ii) a comparação de palavra-a-palavra entre o parâmetro e a própria intenção, (iii) as palavras chave fornecidas pelo operador em relação ao próprio parâmetro, e finalmente, não encontrando similaridade suficiente nas opções anteriores, (iv) são analisados os comentários associados a cada parâmetro fornecidos pelo operador. Quando houver mais de uma palavra para comparação, o resultado final é a média ponderada das similaridades e

quantidades de palavras em cada arranjo. Esse procedimento aumenta significativamente as chances de sucesso na combinação, pois expande o escopo textual que o SLOrion pode utilizar para estabelecer equivalências. A partir dessa combinação entre as quatro diferentes abordagens, o maior valor de similaridade é extraído. Adicionalmente, o operador adiciona um valor de *threshold*, que tem por finalidade funcionar como um balizador para a identificação dos parâmetros. Na próxima seção é descrita a prototipação do SLOrion.

5. Experimentação e Resultados

SLOrion foi desenvolvido em Python, recebendo as requisições via *Application Programming Interface* (API) REST. A entrada do SLOrion é um texto com uma sequência de intenções declaradas no contrato inteligente. Em seguida, SLOrion analisa seu conteúdo e carrega um modelo padrão de *JavaScript Object Notation* (JSON) para acomodar os valores correspondentes. Conforme as intenções são decodificadas, SLOrion preenche ou ajusta campos no JSON, refletindo parâmetros de rede relevantes, como largura de banda, disponibilidade ou limites de sessão. Ao final desse processo, a saída do SLOrion é um JSON preparado e o monitor de cilo de vida do fatiamento de rede como serviço é acionado. A seguir, são detalhados os procedimentos adotados na metodologia de avaliação, incluindo os critérios e abordagens utilizadas para conduzir o estudo.

5.1. Metodologia de avaliação

A avaliação foi conduzida em duas etapas, a primeira delas foi analisar a execução e validação manual do algoritmo, obtendo evidências sobre o seu funcionamenteo e suas limitações. A segunda etapa, possui o objetivo de comparar com a solução de similaridade com a ferramenta da OpenAI, utilizando o método ChatGPT o1, como referência. A OpenAI foi escolhida como referência por representar uma abordagem consolidada em Inteligência Artificial. O tempo de execução foi medido entre o início e o término de cada execução dos algoritmos para ambas as abordagens. Os experimentos foram realizados em um ambiente controlado, utilizando um computador com processador Core i7 5500U e 8 GB de memória RAM. Todas as medições foram realizadas do início ao fim da execução dos algoritmos, assegurando a precisão e a consistência dos resultados obtidos.

Os testes foram executados usando o modelo NEST da GSMA, através de um JSON [GSMA 2024], com 47 parâmetros no total. Os experimentos contiveram 5, 10, 20 e 40 intenções de parâmetros como entrada. Esses parâmetros foram acumulativos, ou seja, os primeiros 5 parâmetros estão contidos nos 10 seguintes parâmetros, e os 10 seguintes parâmetros estão contidos nos 20 parâmetros subsequentes, e assim por diante. Os parâmetros utilizados foram configurados com 40% a 90% de similaridade com os parâmetros originais do NEST. Cada experimento foi repetido 50 vezes para minimizar variabilidades e garantir a reprodutibilidade dos resultados os modelos estão disponíveis no Github [2]. Além disso, os testes avaliaram diferentes *thresholds* de similaridade, variando entre 0.5 e 0.9, permitindo uma análise detalhada sobre o impacto dessa variação na eficiência do SLOrion.

Os testes incluíram a avaliação do tempo de processamento e execução de cada abordagem, a quantidade de erros e acertos, bem como a incidência de falsos positivos. Esses parâmetros forneceram uma visão abrangente sobre o desempenho e a precisão das soluções, permitindo identificar suas respectivas vantagens e limitações

[2]https://github.com/arieldll/SLOrion

5.2. Análise de precisão: SLOrion

Inicialmente, os resultados são analisados considerando a quantidade de acertos e quantidade de falsos negativos, que são os parâmetros que não foram, mas deveriam ser identificados. Em seguida, falsos positivos, que são os parâmetros que foram identificados de forma incorreta, são discutidos.

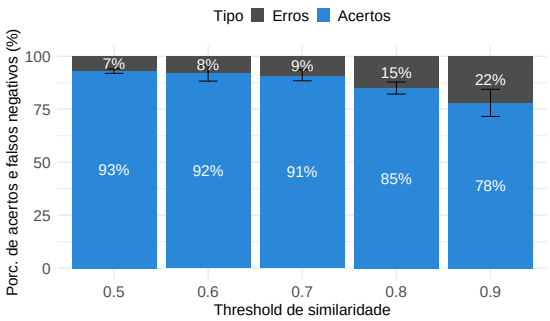

Figura 4. Porcentagem de acerto e falsos negativos

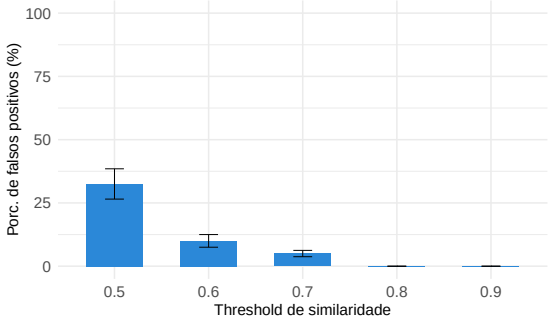

Figura 5. Porcentagem de falsos positivos

A Figura 4 apresenta a porcentagem de acertos e falsos negativos pelo valor do *threshold* no SLOrion. O eixo x representa o valor absoluto do *threshold* e o eixo y, representa a porcentagem em cada uma das situações, i.e., porcentagem de acertos (em azul) e falsos negativos (em cinza escuro). Complementarmente, a Figura 5 apresenta a porcentagem de falsos positivos em função do mesmo valor de *threshold*, onde o eixo x denota o *threshold* e o eixo y, o percentual de falsos positivos. A intenção dessas visualizações é fornecer uma análise detalhada de como diferentes *thresholds* afetam a eficácia do sistema na classificação de parâmetros relevantes.

Observa-se que, à medida que o *threshold* aumenta, a porcentagem de acertos diminui, alcançando de 93% no *threshold* 0.5 para 78% no *threshold* 0.9. Entretanto, os falsos negativos aumentam de 7% para 22%, indicando que *thresholds* mais altos tornam o sistema mais seletivo, reduzindo sua capacidade de identificar parâmetros. Além disso, conforme o *threshold* aumenta, os falsos positivos diminuem de forma significativa, chegando a valores praticamente nulos nos *thresholds* 0.8 e 0.9, motivados pela diminuição do erro do coeficiente por *thresholds* menos permissivos. Isso demonstra que *thresholds* mais altos tornam o sistema mais conservador, quase eliminando falsos positivos, com o custo de deixar de identificar alguns parâmetros corretamente.

A relação entre as duas análises destaca uma dinâmica inversa entre falsos positivos e falsos negativos. *Threshold* mais baixos, como 0.5 e 0.6, apresentam menores taxas de falsos negativos, o que permite identificar um maior número de parâmetros. No entanto, esses *thresholds* também estão associados a uma maior proporção de falsos positivos, indicando uma maior propensão do SLOrion de identificar erroneamente parâmetros que não deveriam ser incluídos. Em contrapartida, *thresholds* mais altos, como 0.8 e 0.9, praticamente eliminam os falsos positivos, mas aumentam significativamente os falsos negativos, reduzindo a abrangência do sistema.

Entre os valores analisados, o *threshold* 0.7 se destaca como o mais equilibrado. Nesse ponto, SLOrion mantém uma alta taxa média de acertos (91%) e reduz os falsos negativos para 9%, i.e., o menor valor entre os *thresholds* com falsos positivos. Além disso,

os falsos positivos com *threshold* 0.7 são mínimos, representando uma melhora significativa em relação a *thresholds* mais baixos. Esse equilíbrio faz do *threshold* 0.7 uma excelente escolha para maximizar a identificação correta de parâmetros, sem comprometer a precisão. Dessa forma, para comparação efetiva com outras técnicas, o *threshold* de 0.7 foi utilizado como referência.

5.3. Comparativo: SLOrion e OpenAI

As seguintes análises comparam os métodos SLOrion e OpenAI em relação aos falsos positivos e ao tempo de processamento, avaliados conforme a quantidade de intenções de parâmetros. Cada análise possui uma métrica distinta, proporcionando uma clarificação sobre o desempenho e a eficiência de cada abordagem.

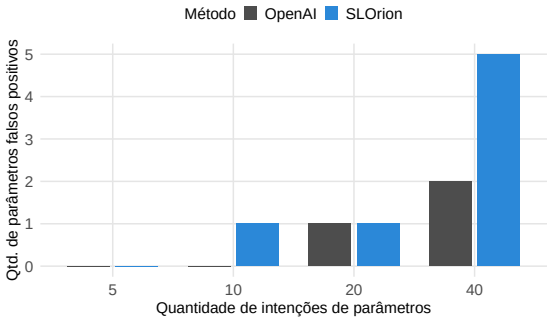

Figura 6. Comparação de falsos positivos

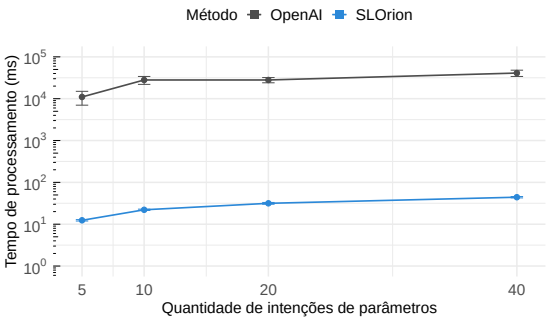

Figura 7. Comparação de tempo por intenção de parâmetros

A Figura 6 exibe a quantidade de intenções de parâmetros no eixo x e a quantidade de falsos positivos no eixo y. Os valores no eixo x variam entre 5, 10, 20 e 40 intenções de parâmetros da operadora. Observa-se que SLOrion apresenta um aumento considerável nos falsos positivos à medida que a quantidade de intenções cresce, atingindo um total de 5 falsos positivos no cenário mais complexo (40 intenções). Entretanto, OpenAI demonstra um comportamento mais consistente, com um crescimento mais moderado nos falsos positivos, alcançando um máximo de 2 falsos positivos no mesmo cenário. Esses resultados mostram que, enquanto SLOrion é competitivo em cenários de menor complexidade, ele se torna mais suscetível a erros em situações mais exigentes, enquanto o OpenAI mantém maior estabilidade na identificação correta dos parâmetros.

A Figura 7 exibe a quantidade de intenções de parâmetros no eixo x e o tempo de processamento em milissegundos no eixo y, que está representado em escala logarítmica. Os valores do eixo x variam entre 5, 10, 20 e 40 intenções de parâmetros. Neste gráfico, SLOrion destaca-se com tempos de processamento significativamente menores e um crescimento gradual, conforme o número de intenções aumenta. Por exemplo, mesmo no maior cenário analisado (40 intenções), o tempo de processamento do SLOrion permanece abaixo de 41 ms. Em contrapartida, OpenAI apresenta tempos de processamento consistentemente mais elevados, ultrapassando 44 segundos, no cenário mais complexo. Isso evidencia que SLOrion é mais eficiente em termos de tempo de processamento, especialmente em aplicações que requerem alta escalabilidade.

Ao analisar os dois gráficos conjuntamente, observa-se um claro *trade-off* entre as soluções. Enquanto, SLOrion apresenta uma notável vantagem no tempo de processamento, essa solução sofre com um aumento nos falsos positivos em cenários com maior

quantidade de intenções de parâmetros. Em contrapartida, OpenAI demonstra um desempenho mais robusto na redução de falsos positivos, mas com o custo de tempos de processamento significativamente mais elevados. Esse comportamento torna o OpenAI menos eficiente em aplicações que demandam alta escalabilidade ou respostas rápidas, mas mais confiável em termos de precisão.

6. Conclusão

Os resultados demonstram que SLOrion é capaz de entregar respostas rápidas, com tempos de processamento na casa de 41 ms mesmo no pior cenário, quando há 40 intenções de parâmetros. Essa eficiência computacional destaca o método como uma solução viável para sistemas que exigem agilidade e alta escalabilidade, mantendo tempos de resposta consistentes mesmo em contextos de maior complexidade. Tal característica torna o SLOrion adequado para aplicações em tempo real, onde o desempenho é um fator crítico.

Embora o método seja eficiente em termos de tempo, os resultados mostram que ainda há espaço para melhorias na precisão. A necessidade de aprimorar a acurácia ressalta a importância de ajustes nos algoritmos de classificação e no balanceamento dos *thresholds* para reduzir os erros sem comprometer a eficiência. Como trabalho futuro, pretende-se focar na melhoria da acurácia do SLOrion, explorando novas abordagens que combinem o método com outras técnicas, a fim de garantir maior precisão na identificação dos parâmetros. O objetivo é desenvolver um sistema mais robusto, que uma a alta eficiência computacional com uma classificação mais confiável, atendendo às demandas de tempo necessárias para não comprometer o funcionamento das redes B5G.

7. Agradecimentos

Os autores agradecem à Fundação de Amparo à Pesquisa do Estado de São Paulo (FAPESP) por meio dos Projetos PORVIR-5G: Programabilidade, ORquestração e VIRtualização de Redes em 5G, concessão 2020/05182-3, e SAMURAI: núcleo 5G inteligente e integração de múltiplas redes de acesso, concessão 20/05127-2. A RNP/MCTIC também apoiou o trabalho sob número 01245.010604/2020-14, no âmbito do projeto 6G Brasil.

Referências

[Bellaj et al. 2024] Bellaj, B. et al. (2024). Drawing the Boundaries Between Blockchain and Blockchain-Like Systems: A Comprehensive Survey on Distributed Ledger Technologies. *Proceedings of the IEEE*, 112(3):247–299.

[Cheng et al. 2024] Cheng, H. et al. (2024). ORANSlice: An Open Source 5G Network Slicing Platform for O-RAN. In *Proceedings of the 30th Annual International Conference on Mobile Computing and Networking*, ACM MobiCom '24, page 2297–2302, New York, NY, USA. Association for Computing Machinery.

[Dai et al. 2025] Dai, J. et al. (2025). O-RAN-Enabled Intelligent Network Slicing to Meet Service-Level Agreement (SLA). *IEEE Transactions on Mobile Computing*, 24(2):890–906.

[Dice 1945] Dice, L. R. (1945). Measures of the amount of ecologic association between species. *Ecology*, 26(3):297–302.

[Gonçalves et al. 2022] Gonçalves, J. P. et al. (2022). Data integrity verification in network slicing using oracles and smart contracts. In *2022 IEEE International Conference on Blockchain (Blockchain)*, pages 476–481.

[GSMA 2024] GSMA (2024). NG.116 - Slice SLA Framework. Technical report, GSMA.

[Javed et al. 2022] Javed, F., Bafalluy, J., and Zeydan, E. (2022). Blockchain-based SLA Management for Inter-Provider Agreements. In *IEEE Conference on Network Function Virtualization and Software Defined Networks (NFV-SDN)*, pages 155–161.

[Javed et al. 2023] Javed, F., Bafalluy, J., and Zeydan, E. (2023). Blockchain-based SLA monitoring for 6G: Inter-Provider Agreements as a Use Case. In *IEEE International Conference on Blockchain and Cryptocurrency (ICBC)*, pages 1–3.

[Kapassa et al. 2018] Kapassa, E., Touloupou, M., and Kyriazis, D. (2018). SLAs in 5G: A Complete Framework Facilitating VNF- and NS- Tailored SLAs Management. In *32nd International Conference on Advanced Information Networking and Applications Workshops (WAINA)*, pages 469–474.

[Krijnen et al. 2024] Krijnen, J. O. et al. (2024). Translation certification for smart contracts. *Science of Computer Programming*, 233:103051.

[Moreno et al. 2024] Moreno, L. et al. (2024). Emergency Systems Democratization: Disability-Inclusive Mobile App Requirements. In *Proceedings of the XXIV International Conference on Human Computer Interaction*, New York, NY, USA. Association for Computing Machinery.

[Muntaha et al. 2025] Muntaha, S. T. et al. (2025). Hybrid Blockchain-Based Multi-Operator Resource Sharing and SLA Management. *IEEE Open Journal of the Communications Society*, 6:362–377.

[Samdanis et al. 2016] Samdanis, K., Costa-Perez, X., and Sciancalepore, V. (2016). From network sharing to multi-tenancy: The 5G network slice broker. *IEEE Communications Magazine*, 54(7):32–39.

[Singh et al. 2024] Singh, J., Rani, S., and Kumar, P. (2024). Blockchain and smart contracts: Evolution, challenges, and future directions. In *International Conference on Knowledge Engineering and Communication Systems (ICKECS)*, volume 1, pages 1–5.

[Touloupou et al. 2019] Touloupou, M. et al. (2019). An Integrated SLA Management Framework in a 5G Environment. In *Conference on Innovation in Clouds, Internet and Networks and Workshops (ICIN)*, pages 233–235.

[Upadhyay et al. 2021] Upadhyay, K. et al. (2021). Can't Understand SLAs? Use the Smart Contract. In *Third IEEE International Conference on Trust, Privacy and Security in Intelligent Systems and Applications (TPS-ISA)*, pages 129–136.

[Yang and Lei 2019] Yang, Z. and Lei, H. (2019). FEther: An Extensible Definitional Interpreter for Smart-Contract Verifications in Coq. *IEEE Access*, 7:37770–37791.

[Zheng et al. 2018] Zheng, Z. et al. (2018). Blockchain challenges and opportunities: A survey. *International Journal of Web and Grid Services*, 14(4):352–375.