frontiers | Frontiers in Blockchain

TYPE Original Research
PUBLISHED 02 October 2024
DOI 10.3389/fbloc.2024.1413840

OPEN ACCESS

EDITED BY
Akihiro Fujihara,
Chiba Institute of Technology, Japan

REVIEWED BY
Qin Wang,
Commonwealth Scientific and Industrial Research Organisation (CSIRO), Australia
Rameez Asif,
University of East Anglia, United Kingdom

*CORRESPONDENCE
Viktor Valaštín,
viktor.valastin@stuba.sk
Dušan Morháč,
xmorhac@stuba.sk
Kristián Košťál,
kristian.kostal@stuba.sk

RECEIVED 07 April 2024
ACCEPTED 16 September 2024
PUBLISHED 02 October 2024

CITATION
Valaštín V, Morháč D, Košťál K and Kotuliak I (2024) Protocol for unifying cross-chain liquidity on polkadot.
Front. Blockchain 7:1413840.
doi: 10.3389/fbloc.2024.1413840

Protocol for unifying cross-chain liquidity on polkadot

Viktor Valaštín*, Dušan Morháč*, Kristián Košťál* and Ivan Kotuliak

Faculty of Informatics and Information Technologies, Slovak University of Technology in Bratislava, Bratislava, Slovakia

Liquidity is critical for a healthy and thriving blockchain ecosystem, enabling value exchange between participants. However, achieving unified liquidity across heterogeneous blockchain platforms remains challenging due to disparities in architecture, virtual machines, and asset management logic. These disparities force assets to be wrapped into other formats to ensure compatibility with underlying systems, thus fragmenting liquidity into multiple pools. This paper proposes LiquiSpell, a novel protocol that aims to unify liquidity across multiple parachains within the Polkadot ecosystem. By leveraging the cross-chain message passing (XCMP), LiquiSpell introduces the concept of a universal transaction that can be constructed to be compatible with any parachain, regardless of its underlying architecture or asset management pallet. This approach overcomes the obstacles posed by the diverse nature of parachains, enabling seamless asset sharing and enhancing cross-chain interoperability. The proposed solution mitigates liquidity fragmentation within the Polkadot ecosystem. It presents a framework that can be extended to other multichain environments outside Polkadot. Ultimately, LiquiSpell aims to foster a thriving ecosystem by facilitating the introduction of new assets and increasing overall liquidity, thereby driving innovation and adoption within the decentralized finance (DeFi) landscape.

KEYWORDS

cross-chain, liquidity, ASSETS, XCM, XCMP, interoperability, Polkadot

1 Introduction

Achieving unified liquidity across diverse blockchain platforms is a critical challenge for fostering a thriving decentralized economy and most ledgers were not designed with interoperability in mind (Qasse et al., 2019). While many researchers strive for interoperability in the inherent diversity of architectures, virtual machines and asset management logic often create limitations in interoperability protocols that can safely integrate support.

The idea of designing a protocol to unify liquidity compatible with any chain is only a future dream for now. Existing interoperability solutions such as Polkadot (Wood, 2016) or Cosmos (Kwon and Buchman) fail to address the inherent heterogeneity of multichain ecosystems, which they are trying to unify using native protocols. Taking Polkadot as an example, we can observe that Parachains, as the storage of assets, provide a different way of accessing liquidity, and there is no unified way to access them. Polkadot-secured chains can communicate and exchange messages using an established communication standard consisting of unique instructions within a native interoperability protocol called cross-chain message passing (XCMP) (Abbas et al., 2022; Burdges et al., 2020). This protocol

sends specifically formatted messages through open channels between Parachains. The message format used by XCMP is the cross-consensus message format (XCM) (Abbas et al., 2022). Other cross-chain solutions, such as the Inter-Blockchain Communication (IBC) protocol used in the Cosmos ecosystem or bridges (Augusto et al., 2024) contribute to fragmentation of liquidity because they create bridge-bound assets that are not compatible with native assets of the chain (Pupyshev et al., 2020). Bridges are currently the most popular way to transfer assets between chains, but they must be fixed (Lee et al., 2023). However, they can help us bring liquidity and partially fix the problem of fragmentation of liquidity (Wan and Adams, 2022; Capponi et al., 2024). This new term is used to describe a problem of liquidity being spread across multiple chains and pools, creating a problem of low liquidity in each of them, making trading assets harder and leading to a higher slippage (Lehar et al., 2024).

Resolving liquidity fragmentation and improving cross-chain asset transferability can yield numerous benefits, including enhanced security, scalability, and connectivity within the ecosystem. The originality of this paper lies in the fact that the current state of DeFi in the Polkadot ecosystem is fragmented through multiple parachains across many liquidity pools (Jakub Gregus/HydraDX, 2022). Parachains, however, can communicate with each other, yet liquidity stays isolated, e.g., Hydration[1] has $27M total value locked (TVL), Moonbeam[2] has more than $45M TVL. Solving liquidity fragmentation issues on the chain level (Whitton, 2021) introduces another storage where liquidity will be stored. Therefore, we are proposing an interoperable solution on the protocol level that could be reused in other ecosystems. This paper presents a novel solution that helps minimize interoperability and liquidity integration problems on the Polkadot network. By leveraging the intrinsic capabilities of XCMP, LiquiSpell introduces the concept of a universal transaction that can be constructed to be compatible with any parachain, regardless of its underlying architecture or asset management pallet. The solution was created by extensive research of cross-chain abilities on each chain in the Polkadot ecosystem and inspired by our continuous development of the common good in multichain ecosystems, where we observed that these problems are more often than expected.

The contribution of this paper is organized as follows:

- Identifying and examining common liquidity fragmentation issues within multichain ecosystems, focusing on the Polkadot network.
- Comprehensive analysis of the architecture of the Polkadot ecosystem, security considerations, and native cross-chain protocols.
- Design of the LiquiSpell solution, aimed at resolving the diversity of interoperability implementations across parachains, thereby enhancing asset liquidity throughout the ecosystem.

The rest of the paper is sectioned in the following manner. Comprehensive interoperability problems, Polkadot, interoperability protocols, and related work are analyzed in Section 2. The design and architecture of the solution are summarized in Section 3. Design and implementation are then tested, and tests are evaluated in Section 4. The overall results of this paper's study and the proposed solution are then discussed in Section 5. Finally, everything is concluded, and the paper is summarized in Section 6. The list of abbrevations used through the paper is in Table 1.

2 State of the art

This section analyzes common cross-chain problems and protocols, focusing mainly on multichain ecosystems. It also goes through the infrastructure of the Polkadot network and related work, where we analyze UniswapX - the leading protocol on liquidity unification, and Axelar (Axelar Team, 2021) with their General Message Passing protocol. We also analyze the state of the SDKs present in the Polkadot ecosystem.

2.1 Background

The following paragraphs go through some common problems with interoperability, cross-chain sharing protocols designed for multichain systems, and Polkadot's infrastructure.

TABLE 1 Table of acronyms.

Acronym	Full form
DeFi	Decentralized Finance
DAOs	Decentralized Autonomous Organizations
NFTs	Non-Fungible Tokens
IBC	Inter-Blockchain Communication
dApp	Decentralized Application
SDK	Software Development Kit
XCMP	Cross-Chain Message Passing
XCM	Cross-Consensus Message Format
HRMP	Horizontally Relay-Routed Message Passing
EVM	Ethereum Virtual Machine
IBC	Inter-Blockchain Communication
MPC	Multi-Party Computation
AMMs	Automated Market Makers
CDK	Constrictor Development Kit (specific to Polygon)
GMP	General Message Passing (specific to Axelar)
RPC	Remote Procedure Call
KPIs	Key Performance Indicators
DDoS	Distributed Denial of Service
UI	User Interface

1 https://hydration.net

2 https://apps.moonbeam.network/moonbeam/app-dir

2.1.1 Blockchain interoperability and common problems

Fully secure interoperability between two ledgers is still nearly impossible to achieve. Some vulnerabilities will almost always exist, left to be discovered by malicious users. Implementing cryptographic protocols on top of blockchains necessitates fine-grained control over how individual transactions are constructed (Eizinger et al., 2021). There are also questions of trust involved. Trust in the protocol can also be achieved using zero-knowledge proofs, which can help us create trustless protocols such as zkBridge (Xie et al., 2022). Protocols that rely on trust can be easily manipulated if certain entities become malicious. Cross-chain technology is a connecting bridge that links different blockchains, whether homogeneous or heterogeneous (Cao and Song, 2021). As already known from past events, bridges are prone to hacks or malicious behavior.

Some of the common problems with interoperability are verification of transaction status on the origin and destination chain (Lin et al., 2021). The other issue is the verification of the token amount that is being transferred. Malicious chains can manipulate the transfer amount, saying they burned more tokens than the transferred amount, creating a supply mismatch between the bridge-connected chains.

2.1.2 Liquidity fragmentation and looking for a unification

Pursuing cross-chain interoperability and unified liquidity in multichain ecosystems is complex, with several common challenges contributing to liquidity fragmentation across the ecosystem.

However, what does liquidity indeed entail within the realm of interoperability refers to the ease and efficiency with which assets can be exchanged or transferred across different blockchain networks. It is a crucial aspect of enabling seamless cross-chain interactions and facilitating the flow of value among diverse ecosystems. Therefore, in the context of blockchain interoperability, liquidity entails several vital components: Asset Transferability, Market Depth, Pool availability, and Incentive Mechanisms (Qin et al., 2021). Inavery simple look, we can compare liquidity in blockchain networks to cash flow in the ecosystem of FIAT money.

- Asset Transferability: Ability to move assets from one blockchain to another without significant barriers or restrictions using secure and reliable cross-chain communication,
- Market Depth: The availability of sufficient liquidity across different chains to facilitate efficient asset trading and exchange without causing significant price slippage
- Pool Availability: The presence of interconnected liquidity pools across different chains to enable seamless asset swapping and trading
- Incentivization Mechanisms: The implementation of reward mechanisms to encourage users to provide liquidity, thereby increasing overall liquidity within the interoperable ecosystem.

Within the Polkadot network, these issues are particularly prevalent due to the diverse nature of its parachains.

One of the primary sources of liquidity fragmentation stems from the architectural heterogeneity inherent in multichain ecosystems like Polkadot. Each parachain is designed with its unique architecture, virtual machine, and asset management logic, resulting in disparate approaches to handle cross-chain transactions and asset transfers. This lack of standardization creates compatibility hurdles, hindering the seamless flow of liquidity across the ecosystem. Looking outside the ecosystem, Polygon (Kanani et al., 2021) is a good example of a network trying to unify liquidity across its ecosystem. Using the Polygon CDK, connected chains can share liquidity and assets, enabling developers to build applications that interact within the Polygon network (Polygon AggLayer). The Aggregation layer chains can submit transactions to the Ethereum network, allowing for the seamless transfer of assets between the two networks. A unique feature of the Aggregation layer is that new chains are exempt from building their bridges to Ethereum, as the Aggregation layer handles the transfer of assets between the two networks with the atomic guarantee (Brendan). Moreover, the Aggregation layer enables asynchronous cross-chain communication, calling contracts from one chain to another without finalizing Ethereum. However, the limitation of CDK and Aggregation layer seems obvious as AggLayer works only with homogeneous chains built using CDK. Therefore, liquidity is distributed only in multiple pools within multiple protocols across multiple homogeneous EVM networks. There are other ways to unify liquidity across the multichain ecosystem, like Cosmos (Kwon and Buchman). It stems from the nature of the Cosmos network, which is designed to be a network of independent chains that can communicate with each other via Inter-blockchain communication protocol (IBC). With more than 50 chains connected to the Cosmos Hub, the network is a prime example of a multichain ecosystem that can benefit from unified liquidity.

The most prominent chains in Cosmos from the perspective of decentralized finances are Injective (Yousaf et al., 2024), which has the highest trading volume, and Osmosis. However, from the IBC perspective, most liquidity is routed through Osmosis (Lucas García De Viedma Pérez, 2023). Osmosis is considered the primary DEX on the Cosmos Hub and the main gateway for cross-chain liquidity (Lagadamane Dinesha and Patil, 2023). Usually, liquidity from other ecosystems is routed through wormhole-wrapped tokens or channel-bound tokens, which fragment the token liquidity across the network, which is a problem. Suppose we bridge 1000 USDC from Injective to Osmosis and 1000 USDC from Cosmos Hub to Osmosis. In that case, our balance will not be 2000 USDC but two different tokens with a balance of 1,000 each.

We can take a simple example of Bitcoin and Ethereum. To use Bitcoin on the Ethereum network, users must wrap their Bitcoin into a wrapped token, such as WBTC. The fact is that WBTC is not the only Bitcoin wrapper on Ethereum. Some notable mentions are RenBTC and HBTC (Giulio, 2021). Multiple wrappers over one asset is an on-point example of how the liquidity of one asset is split on one network. The problem becomes more prominent with bridges that bring their wrappers for the tokens. Having one wrapper per token is somewhat solved on AggLayer, as it will be the go-to bridge from Ethereum to Polygon. However, the problem of liquidity fragmentation will persist in the Polygon network, as the AggLayer will only work with chains built using the CDK. In the case of Cosmos, the problem of liquidity fragmentation is complex as

TABLE 2 Overview of multichain ecosystems.

	Polygon CDK	Cosmos	Polkadot
SDK	CDK	Cosmos SDK	Polkadot SDK
Supported VMs	EVM homogeneous	Heterogeneous	Heterogeneous
Bridged tokens	Wrapped	Wrapped	Native
Wrapper types	One per token	Each channel has custom wrapper	—
Unifier	AggLayer	Osmosis chain	Native chain
Number of chains	1	>50	50

tokens can come from various sources - EVM (injective), CosmWasm, or native IBC tokens. To unify liquidity, the naive way is to transfer all via IBC to Osmosis and swap all tokens to the desired one. However, there are more efficient ways to unify liquidity, as it will result in a significant loss of value due to high slippage and fees. Furthermore, many additional steps are needed to ensure that tokens are in one network.

The state of liquidity in the Polkadot network looks alike. We can see that most liquidity is concentrated in the Polkadot Relay Chain, with a significant portion distributed across the parachains. Due to the diverse nature of parachains, liquidity can be created and originate from various sources. For example, we can illustrate the journey of stablecoin USDT into Polkadot Network. We can use multiple bridges and centralized exchanges to transfer USDT to the Polkadot network. The most common way is to deposit USDT on a specialized system parachain called AssetHub[3]. As the name states, it is a common good parachain that should be a hub for all tokens (Abbas et al., 2022). Reality, however, shows that AssetHub is one of many parachains that can handle tokens. For example, USDT can be deposited on the Moonbeam parachain, a parachain fully compatible with the Ethereum network. USDT represents an excellent example of how liquidity can be fragmented across the Polkadot network.

And still, the Moonbeam USDT and AssetHub USDT are the same. Using the XCMP protocol allows for seamless transfer of assets between parachains. Thanks to the multilocations in the XCM protocol, the Moonbeam chain is aware that the asset of id *1984* coming from AssetHub should be translated as USDT. However, the problem of liquidity fragmentation persists in the Polkadot network, as most of the liquidity is concentrated in the Polkadot Relay Chain. Few users know they can use XCM to transfer assets between parachains, as Polkadot is known for its complexity.

The comparison Table 2 provides a concise overview of the key characteristics and differences among three prominent blockchain ecosystems: Polygon CDK, Cosmos, and Polkadot. Polygon CDK focuses on a homogeneous architecture, utilizing the Ethereum virtual machine (EVM) throughout its network. This approach simplifies development and interoperability within the ecosystem but may limit the diversity of use cases. In contrast, Cosmos and Polkadot embrace heterogeneous architectures, allowing for a broader range of specialized chains and virtual machines. AggLayer facilitates cross-chain communication in Polygon CDK, IBC in Cosmos, and XCMP in Polkadot. While AggLayer and IBC rely on wrapped token representations for asset transfers, Polkadot's XCMP enables the exchange of native tokens across its parachains, reducing the need for additional token wrapping and potential liquidity fragmentation. Liquidity unification is addressed differently in each ecosystem. Polygon CDK utilizes AggLayer to aggregate liquidity across its chains. Cosmos relies on the Osmosis DEX as a central liquidity hub, and Polkadot leverages its Relay Chain to facilitate asset transfers and liquidity provision. All three ecosystems have a significant presence, with Cosmos and Polkadot each boasting more than 50 connected chains or parachains, while Polygon CDK is rapidly growing.

Pursuing cross-chain interoperability and unified liquidity in multichain ecosystems is a problematic task plagued by several common challenges. Lack of standardization, architectural heterogeneity, and awareness are primary sources of liquidity fragmentation that hinder the seamless flow of liquidity across the ecosystem. While networks each have their solution for liquidity unification, Polkadot, with an XCM multilocation feature, has the advantage of becoming the hub of liquidity in the multichain ecosystem. The following section will discuss the potential solutions of the Polkadot and compare them with state-of-the-art solutions from other networks (Wood, 2023; Zhang et al., 2023; Siniscalchi et al., 2023). liquidity is arguably the most significant metric in the economic landscape of blockchain networks, particularly in the context of blockchain interoperability within the Polkadot. Consider a hypothetical scenario where the trading market for a specific parachain is grappling with liquidity issues due to a myriad of problems.

Faced with the industry's volatility and uncertainty, traders within this sector have opted to withdraw their participation. Consequently, the trading market now comprises half of its original participants, leading to broader spreads, diminished turnover, and a reduced volume of trades. This situation adversely impacts users within this industry, as they cannot issue additional shares or derive benefits from existing ones. The likely outcome is a depreciation in their market price, which, in certain circumstances, could precipitate the end of a parachain. Therefore, a decline in liquidity has profound implications not only within the trading sphere but also on Polkadot's economic stage. Thus, understanding and addressing liquidity issues is crucial for the sustainability and growth of the Polkadot blockchain.

2.1.3 Polkadot's infrastructure

In the Polkadot ecosystem, a heterogeneous multiverse of blockchains coexists, each with its unique architectural design

3 Statemint was the original name.

and purpose. As outlined in the Polkadot white paper (Wood, 2016) by founder Gavin Wood, this scalable multichain network serves as a launching pad for diverse services, known as parachains, interconnected through the central Relay Chain connected via the message channel. Polkadot uses nominated proof of stake with more than 1,000 validators to ensure security.

The architecture overview in Figure 1 (Polkadot architecture overview) shows that Polkadot contains a Relay Chain, which is the leading chain responsible for network security and validating blocks. It runs an "OpenGov" model, allowing token holders to decide about funding new projects and runtime upgrades of the chain. Parachains are independent chains with their use cases, governance, and tokens connected to the Relay Chain all the time. Parathreads are similar to parachains but only connect when they need to fetch data or submit new transactions. The ecosystem also contains collators, which collect transactions and extrinsic on parachains or parathreads, and validators, which validate blocks received from parachains and blocks produced by the relay-chain (Wood, 2016; Abbas et al., 2022; Burdges et al., 2020).

2.1.4 Cross-chain protocols in multichain ecosystems

There are multiple well-known multichain ecosystems. One of them is Cosmos (Kwon and Buchman), which offers mentioned IBC. Inter-blockchain communication is an end-to-end protocol for reliable, ordered, and authenticated cross-chain communication between heterogeneous ledgers that are either part of the Cosmos ecosystem or implement protocol support manually. IBC's operation is similar to sending internet packets, ensuring safety through the presence of at least two Relayer entities. The packet flow adheres to a specific process, accounting for successful deliveries and scenarios where messages fail due to timeouts or unreachable destinations (Kim et al., 2022).

As Figure 2 (Cosmos IBC documentation) shows, there can be two different situations. One is where IBC works as expected, and the other is where the message fails to be delivered due to a timeout or an unreachable destination.

The other example of the multichain ecosystem is Polkadot (Wood, 2016), which offers mentioned XCMP. XCMP as a protocol ensures four key aspects (Burdges et al., 2020):

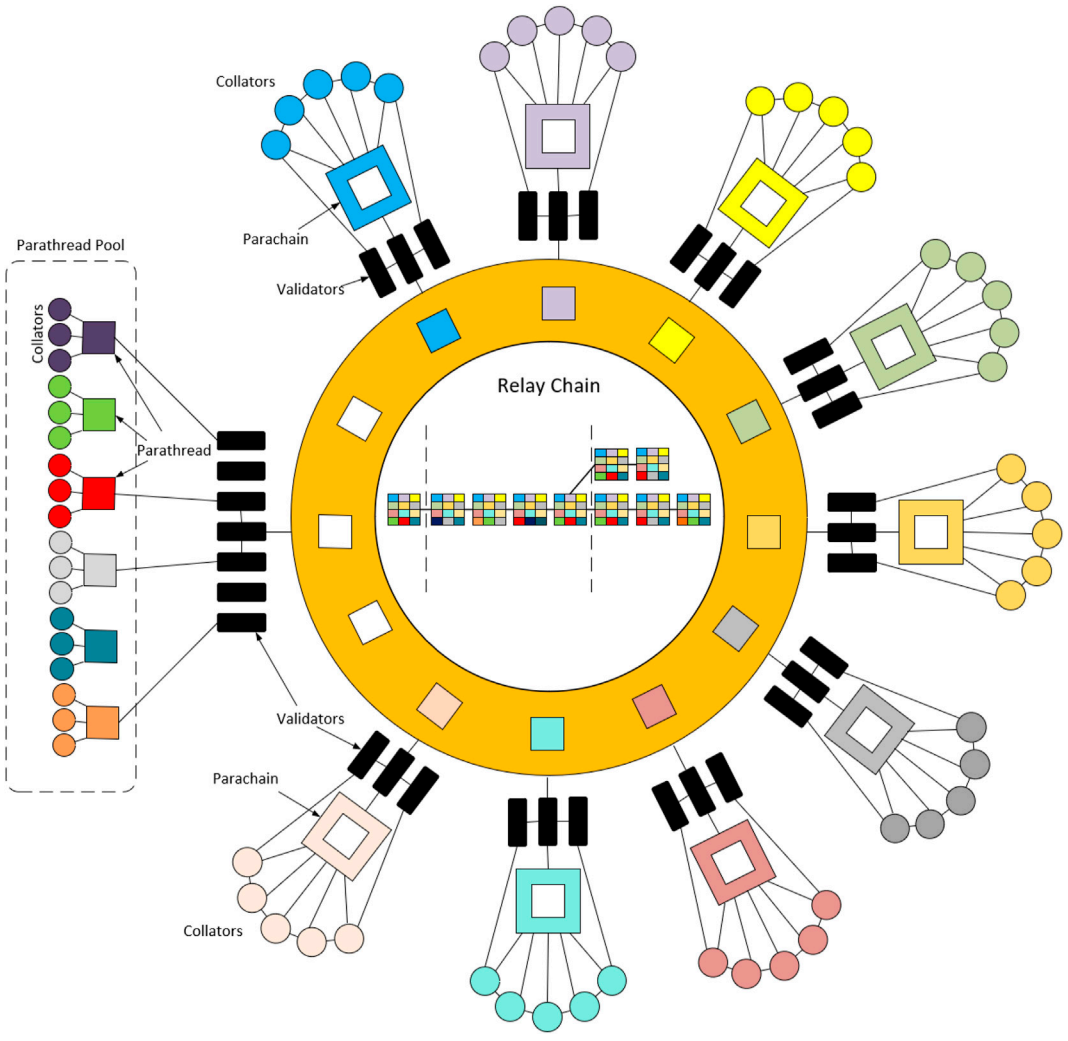

FIGURE 1
Polkadot's high level architecture overview (Polkadot architecture overview).

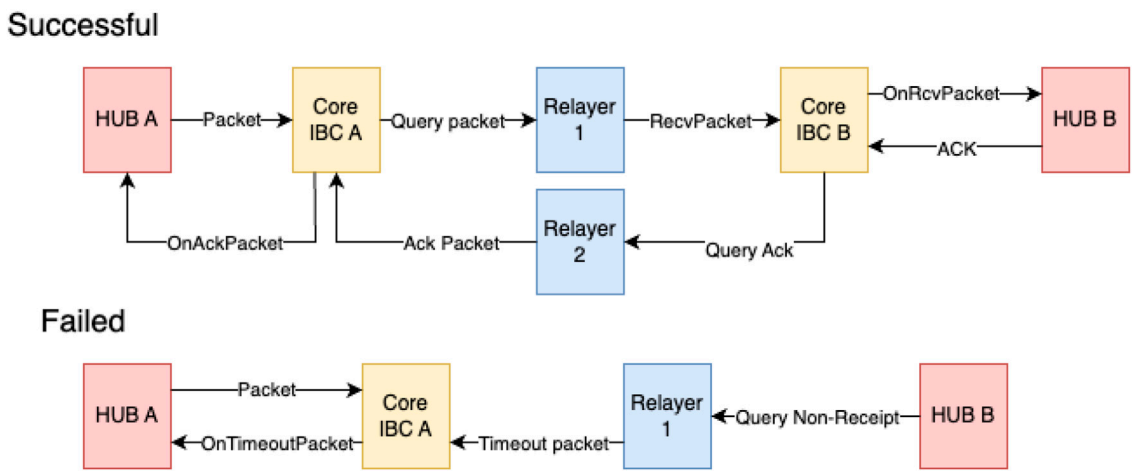

FIGURE 2
Cosmos IBC packet flow (Cosmos IBC documentation).

```
api.tx.xcmPallet.reserveTransferAssets
(
  {
      "V3": {
          "parents": 0,
          "interior": {
              "X1": {
                  "Parachain": 2006
              }
          }
       }
  },
  {
      "V3": {
          "parents": 0,
          "interior": {
              "X1": {
                  "AccountId32": {
                      "id": "0x84fc49ce30071ea611731838cc7736113c1ec68fbc47119be8a0805066df9b2b"
                  }
              }
          }
      }
  },
  {
      "V3": [{
          "id": {
              "Concrete": {
                  "parents": 0,
                  "interior": "Here"
              }
          },
          "fun": {
              "Fungible": "1000000000000"
          }
      }]
  },
  0
);
```

FIGURE 3
Classic XCM message construction through raw javascript.

- Swift delivery of messages from the source chain to the target chain.
- Ensuring the sequential arrival of messages.
- Assure that messages reaching the destination are formally acknowledged on the source chain.
- Equitable distribution of messages to recipients to prevent senders from waiting indefinitely for message visibility.

To implement dApp support for XCMP, developers must install SDK packages called PolkadotJS Apps (PolkadotJS Apps package repository). PolkadotJS SDKs allow developers to construct messages in the way shown in Figure 3. The call *reserveTransferAssets* takes four parameters: destination, beneficiary, assets, and asset that will be used to pay the fee. As we can observe, each parameter is described in the XCM language in version 3 with the

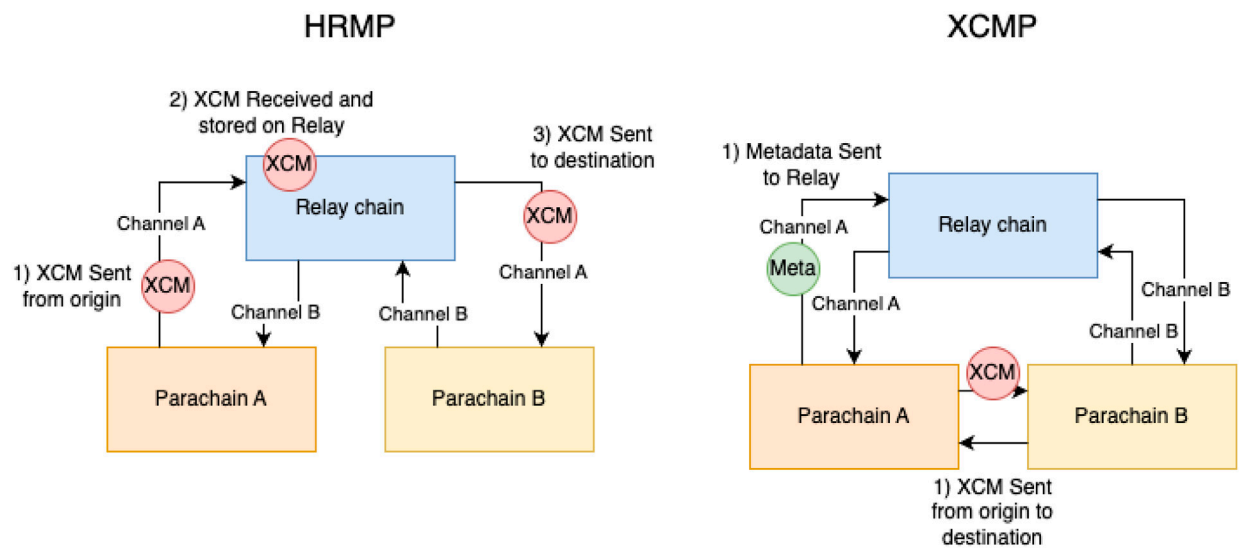

FIGURE 4
XCMP compared to HRMP.

so-called multilocation. Since we want to transfer the DOT token to the Astar parachain, the destination will be the Astar parachain ID. The second parameter is the account used to receive an asset. The third parameter is an array of assets we want to teleport to the parachain. In this case, the only asset on Polkadot is DOT, a native token to the chain, therefore marked as *Here*. The last parameter is the asset that will be used to pay fees. As before, we can only use DOT.

For cross-chain messages to pass between chains, they have to be connected by channels. These channels are not bidirectional by default, so each end should open a one-way channel to the destination. Once opened, these channels remain open for cross-chain message passing until manually closed.

While XCMP is still under development, the Horizontally Relay Routed Message Passing (HRMP) protocol is a substitute, albeit slower and more resource-intensive. HRMP, often referred to as XCMP light, routes messages through the Relay Chain, where we e the entire XCM message, in contrast to XCMP's direct routing between chains without needing whole message storage on the Relay Chain. Routing messages through Relay Chain adds overhead for Relay Chain validators. The way protocols work can be seen in Figure 4, which compares them. The critical difference between XCMP and HRMP lies in the channels. The channels between the Relay Chain and Parachains serve only for communication between the Relay Chain and Parachains in XCMP. In contrast, compared to HRMP, Parachain to Parachain communication is handled directly between chains in XCMP.

As Figure 4 shows, XCMP only stores the necessary metadata on the Relay Chain while the message passes directly to the destination chain. On the other hand, HRMP has to go layer up to the Relay Chain, which has to store the entire message and then redirect the message to the correct destination Parachain in the lower layer.

Both protocols utilize the cross-chain message format (XCM) to ensure compatibility and understanding between chains with different cross-chain modules. Although initially limited to the Substrate framework, the XCM format is expected to expand and become compatible with other ecosystems as support grows. The XCM format can be observed in Figure 3.

2.2 Related work

Despite the Polkadot ecosystem's vision of facilitating cross-chain communication and interoperability, existing tools and protocols fall short in addressing the intrinsic diversity of parachains and their varying implementations of cross-chain integration. While the native XCMP protocol aims to provide a common language for cross-parachain messaging, the heterogeneity in architectures, virtual machines, and asset management pallets employed by individual parachains poses significant challenges to achieving seamless liquidity unification.

This section discusses state-of-the-art projects trying to unify liquidity across different blockchains. From the vast number of projects aggregating liquidity, we chose the best candidates from different ecosystems as they applied various approaches to reach the same goal.

2.2.1 UniswapX

UniswapX is a proposed novel, non-custodial, decentralized trading protocol designed to unify liquidity across blockchain networks. It aims to achieve this through a few key mechanisms.

- Signed Orders and Dutch Auctions - Instead of sending transactions directly, users sign off-chain orders specifying the trade parameters like input/output tokens and amounts. These orders use a Dutch auction model where the price decays over time until filled, incentivizing liquidity providers ("fillers") to provide the best price
- Aggregating On-Chain and Off-Chain Liquidity - Fillers can source liquidity from various venues - on-chain DEXs like

Uniswap, off-chain order books, or even other UniswapX orders.

- Cross-Chain Trading - UniswapX supports cross-chain trading, allowing users to trade assets on one chain for assets on another. It uses light client bridges or canonical bridges to enable this cross-chain communication in a trustless manner
- Optimistic Cross-Chain Orders - UniswapX uses an optimistic model to circumvent slow bridges for cross-chain trades. Fillers execute on the destination side and then submit proof of execution. This enables near-instant cross-chain trades.

Figure 5 shows the sequence of interactions in the UniswapX protocol to facilitate cross-chain swaps.

The process begins with the *Swapper* signing an off-chain order that specifies the trade parameters, such as the input and output tokens, amounts, and other conditions. This signed order is spread over a network of *Fillers*, incentivized to provide the best possible execution price through a competitive Dutch auction model. Upon receiving the order, a *Filler* submits it to the *OriginReactor* contract, along with the *Swapper* funds and a bond, initiating the cross-chain swap process. The *OriginReactor* then interacts with the *DestReactor* contract on the destination chain, facilitating the transfer of the output tokens to the *Swapper* address by the *Filler*. To ensure the integrity of the cross-chain transaction, *DestReactor* records the order as fulfilled and relays a confirmation message through an *Oracle* bridge back to *OriginReactor* on the origin chain. This Oracle acts as a trustless intermediary, enabling secure cross-chain communication. In the optimistic case, where the fill is not challenged within a predetermined time frame, *OriginReactor* releases the *Swapper* funds and the *Filler* bond back to *Filler*, completing the swap. However, suppose that the fill is challenged during this period. In that case, *Filler* must provide valid proof of execution to the Oracle. If the evidence is valid, *Filler* receives the funds, the bond, and the challenger's bond. Conversely, if the proof is invalid, *Swapper* funds are returned, and the challenger receives a portion of the Filler's bond as a reward for identifying the invalid fill. This optimistic approach to cross-chain order settlement enables UniswapX to construct a fast and inexpensive bridging mechanism on top of any existing bridge infrastructure. Combining signed orders, Dutch auctions, liquidity aggregation, and optimistic cross-chain mechanisms, UniswapX presents a compelling solution to the liquidity fragmentation challenges, fostering a more efficient and interconnected decentralized economy (Austin Adams).

Similarly to Polygon CDK, UniswapX is limited to homogeneous EVM networks such as Polygon, Base, and others. Cross-chain bridges

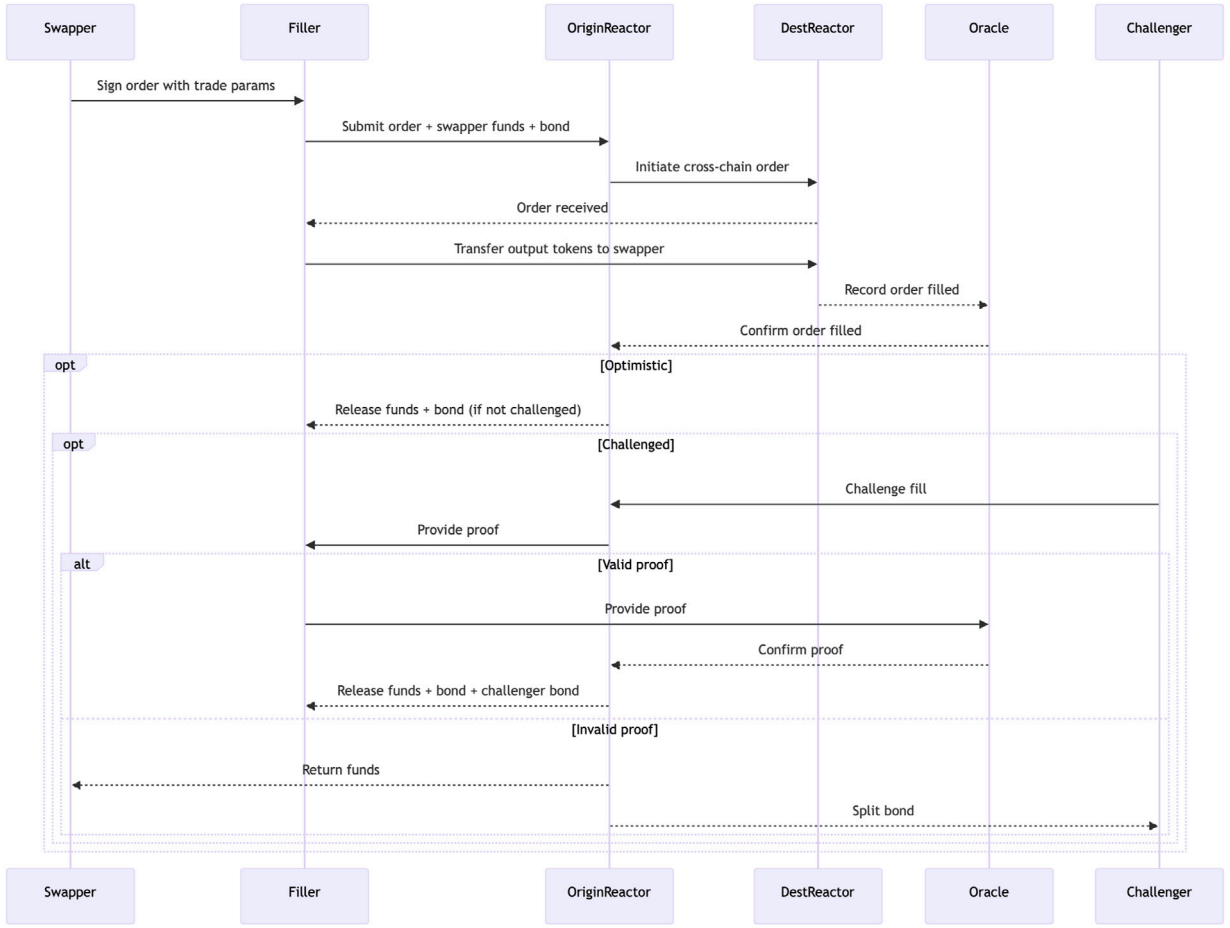

FIGURE 5
UniswapX approach to cross-chain swaps.

such as Wormhole would have the same effect on liquidity fragmentation as tokens would become wrapped in bridge-specific contracts.

2.2.2 Axelar and GMP

Axelar is a decentralized network that aims to enable frictionless communication and asset transfer across different blockchain ecosystems. It provides a protocol suite and APIs that allow applications to perform cross-chain requests and operations efficiently.

Axelar's approach towards unifying liquidity across blockchain ecosystems can be summarized into four key components:

- Cross-Chain Gateway Protocol (CGP) - enables routing and discovery of addresses/applications across different blockchain networks. It allows synchronizing state and transferring assets back and forth between any pair of blockchains, even those that do not natively support smart contracts like Bitcoin. The CGP is analogous to the Border Gateway Protocol (BGP) on the Internet, facilitating cross-chain routing and communication.
- Cross-Chain Transfer Protocol (CTP) is an application-level protocol that makes it easy for decentralized applications to leverage Axelar's cross-chain capabilities. Dapp developers can integrate their smart contracts with Axelar's "threshold bridge accounts" to execute queries to deposit, withdraw, and transfer assets seamlessly across chains. The CTP is analogous to application-level protocols such as HTTP/HTTPS on the Internet.
- network more than a protocol - Axelar is a Cosmos SDK-based blockchain network providing a decentralized cross-chain communication infrastructure. It has a set of validators that run a Byzantine consensus protocol to collectively control the threshold accounts on each bridged blockchain using secure multi-party computation (MPC). Using MPC enables high-security cross-chain transfers without any single point of control.
- Plug-and-Play Integration - Blockchain platforms only need to set up a threshold account controlled by Axelar validators to get plugged into Axelar's cross-chain network without custom integration work. This plug-and-play approach simplifies the process of bridging different blockchain ecosystems.

The main component for liquidity unification is a General Message Passing (GMP) protocol that allows one to call a contract from one chain to another. Alternatively, we can attach tokens with the contract call. This protocol is implemented as a set of smart contracts on each blockchain controlled by a set of validators. The current implementation supports EVM and Cosmos SDK-based blockchains. However, the protocol can also be extended to support other blockchain platforms.

Looking at Figure 6, we can see a sequential diagram that describes the workflow of Axelar's General Message Passing (GMP) protocol. The process is initiated when the user triggers a cross-chain call through the source contract on chain A. Subsequently, the source contract invokes the *callContract* function on the Axelar Gateway contract of chain A, specifying the destination chain, the address of the destination contract, and the payload containing the call data of the function. Upon receiving the call, the Axelar Gateway contract on chain A emits a *ContractCall* event monitored by the Axelar network. The Axelar network, comprising a set of decentralized validators, then uses a consensus protocol to validate the *ContractCall* event content through a voting process. Once the validation is successful, the Axelar network prepares a signed batch of *ContractCall* approved payloads, including the validated call, and submits it to the Axelar Gateway contract on the destination chain B. Upon receiving the signed batch, this contract records the approval of the payload hash and emits a *ContractCallApproved* event. A trustless relayer service monitoring the *ContractCallApproved* event invokes the *IAxelarExecutable.execute()* function on the destination contract, passing along the payload and other relevant data as parameters. The destination contract, in turn, calls the *validateContractCall()* function on the Axelar Gateway contract to verify the authenticity of the call's approval by the Axelar validators. When the Axelar Gateway contract confirms the approval, the destination contract executes its logic using the payload received from the cross-chain call, effectively enabling the cross-chain function invocation facilitated by the Axelar network and its decentralized consensus mechanism (Axelar GMP overview).

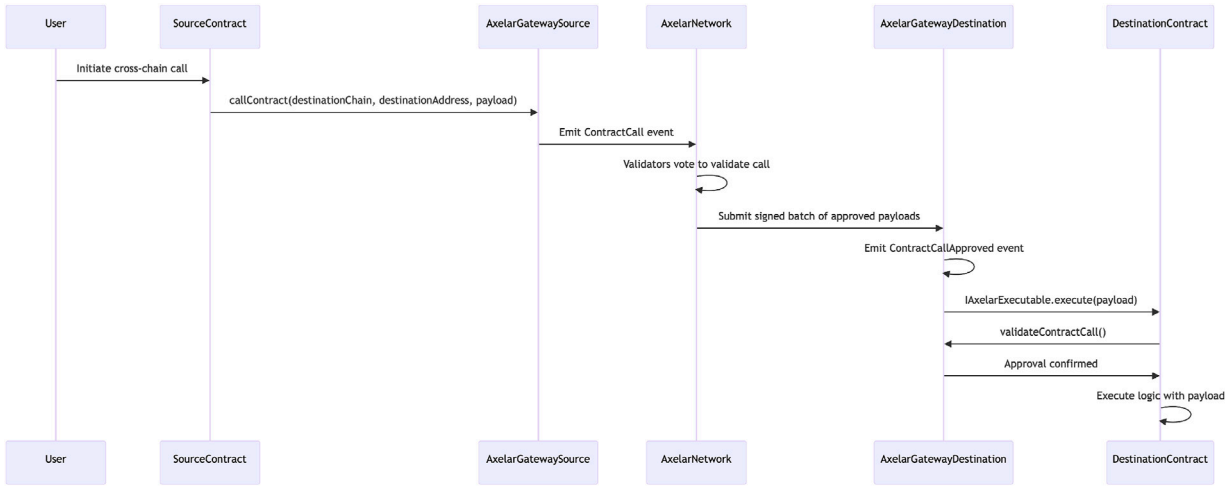

FIGURE 6
Sequential diagram of Axelar General Message Passing protocol.

Axelar team solved one of the friction points of bridging tokens via the ecosystem. As we previously mentioned, the reason that bridges are fragmenting liquidity is that they are wrapping tokens into bridge-specific tokens. Observing Listing 1, we can unwrap tokens directly using AxelarJS. In the example, we are trying to unify our AVAX liquidity wrapped on Polygon and bring it back into the Avalanche. Following this pattern, we can get a native AVAX token without any need to unwrap it on Avalanche with more transactions.

```
const sdk = new AxelarAssetTransfer({
environment: "testnet" });
const fromChain = CHAINS.TESTNET.POLYGON,
    toChain = CHAINS.TESTNET.AVALANCHE,
    destinationAddress = "0xF16DfB26e
    1FEc993E085092563ECFAEaDa7eD7fD",
    asset = "wavax-wei" // Send wrapped AVAX (WAVAX)
    from Polygon
const depositAddress = await sdk.getDepositAddress({
    fromChain,
    toChain,
    destinationAddress,
    asset,
    options: {
      shouldUnwrapIntoNative: true // unwraps into
      AVAX token
    }
});
```

Listing 1. Depositing and unwrapping AVAX token using GMP (Send a wrapped native token).

2.2.3 XCM SDK in polkadot ecosystem

At the time of writing, the Polkadot ecosystem does not offer any protocol to unify liquidity, as the current state of the art is considered two SDKs to create XCM calls. First is the official Parity Asset transfer API (Parity asset transfer API Documentation; Parity asset transfer API repository; Parity asset transfer API Registry). Built as a Typescript package, we can integrate it into our decentralized application. Moreover, Asset transfer API is doing additional work in the background, such as checking if the asset is registered on the chains, solving problems with multilocations, and ensuring that we use the correct amount of decimals. However, using Asset transfer API is still quite complex if we read the Listing 2. We must be aware of particular things, such as Parachain's ID, the ID of the token, and XCM versions.

```
api.createTransferTransaction(
      '2006', // Parachain ID of Astar
      '5F5586mfsnM6durWRLptYt3jSUs55KEmahdodQ5tQMr9iY96',
      ['0'], // Token ID of DOT
      ['1000000000000'],
      {
          format: 'call', isLimited: true, xcmVersion: 3,
          transferLiquidToken: true,
      }
);
```

Listing 2. Sending token to parachain via Parity Asset transfer API.

On the other hand, this has already been solved by Moonbeam XCM SDK (Moonbeam XCM SDK Documentation; Moonbeam). SDK allows easy asset transfers in all three scenarios (UMP, DMP, HRMP). SDK also prevents developers from needing to address complex details like multiple locations or specific extrinsic on specific chains. Currently, SDK supports transfers to many ecosystem chains, which is a notable improvement since the studies we conducted before in (Morháč et al., 2022; Morháč et al., 2023a; Morháč et al., 2023b). Earlier versions of SDK focused mainly on Moonbeam network (Moonbeam official website) use cases rather than general and overall ecosystem improvement. Compared to Asset transfer API Mo,nbeam XCM SDK is much friendlier. We have rewritten the code from Listing 2 into Moonbeam SDK as shown in Listing 3. We know that we plan to send a DOT token from the Polkadot Relay Chain into the Astar parachain, where the destination address is in the EVM format.

```
const data = await Sdk()
.assets()
.asset('dot')
.source('polkadot')
.destination('astar')
.accounts(pair.address, evmSigner.address, { pair });
const hash = await data.transfer(1000000000000);
```

Listing 3. Sending token to parachain via Moonbeam XCM SDK.

In short, all state-of-the-art solutions have pros and cons in unifying and transferring liquidity across chains. We have summarized the state-of-the-art in Table 3. UniswapX handles the fees perfectly, and trying to find the best possible price using Dutch auctions, though limited to EVM networks, needs to be connected via bridges. However, at the time of writing, it is available only on Ethereum[4], aggregating other decentralized exchanges, and access to it is granted[5]. Moreover, it is the only proto-intent solution. Therefore, we do not know how tokens will be unified; we must know what we want to achieve. Axelar, with the GMP protocol, perfectly handles the edge cases of bridge wrapping and seamless token transfers through EVM and IBC (when we are using Cosmos); however, to unify tokens from one currency to another, we would need to employ additional swapping and routing, for example, via the Osmosis chain. It is essential to mention that GMP is also dependent on the finality of the chains and that some of them can take up to 80 min to complete the transaction (Understanding Interchain Transaction Time). Lastly, the Polkadot ecosystem needs a usable solution. The ecosystem offers a few SDKs that can amazingly manage multilocation, multi-address format, and translations from heterogeneous VM systems. However, complex use cases need to be built from scratch.

4 https://support.uniswap.org/hc/en-us/articles/17542724911501-What-networks-are-supported-by-UniswapX

5 https://docs.uniswap.org/contracts/uniswapx/guides/becomequoter

TABLE 3 Comparison of state-of-the-art solutions.

	UniswapX	Axelar	Polkadot XCM SDK
VM support	EVM	EVM, Cosmos	Homogeneous
Transaction type	Signed order	Transaction	Transaction
Unique feature	Dutch Auctions	Plug-and-play architecture	Quick finalization
Message protocol	REST API	GMP	XCM
Limitation	Only EVM networks/relying on API	High finalization time	Complexity of XCM
Finality	2 epochs (~13 min)	Up to 80 min	2 blocks (~30 s)
Liquidity sourcing	Automatic	Manual	Manual
Launched	Only on Ethereum	Yes	Yes
Processing party	Server with API	Validators	Validators

3 Solution design

This section provides an overview of the solution's design. We review its core functionality and describe each component. We also discuss all design decisions and the solution's goals.

3.1 Solution architecture overview

The main goal of this paper was to create a protocol to unify liquidity. We have taken a slightly different approach as the related projects try to simplify transferring and swapping tokens. Observing users in the DeFi space, we have noticed that users are not only interested in swapping tokens but usually intend to use the token in a specific way. For example, the user swaps fungible tokens on Uniswap and wants to buy non-fungible tokens on OpenSea. Therefore, the common route for the user is as follows:

- 1. Find the non-fungible token on OpenSea they would like to buy.
- 2. Check the balances on the wallet they would like to use.
- 3. Find the best exchange rate for the token they would like to swap.
- 4. Swap the token on the exchange.
- 5. Transfer the token to the wallet and chain they want to use.
- 6. Buy the non-fungible token on OpenSea.

As we can see, this process is fairly complex, and non-technical users will need help completing it. Additional complexity is added when the user bridges some tokens to the different chains. Onboarding users into the blockchain ecosystem should be smooth. Therefore, we would like to present our solution for unifying liquidity, LiquiSpell. The core of our protocol lies in a few main components:

- Liquidity sourcing - find the balances of appropriate tokens across the whole Polkadot ecosystem,
- Liquidity rebalancing - find the optimal way of getting desired liquidity,
- Asset Routing - in case of swapping tokens, find the most optimal route,
- AutoTeleportation - aggregating liquidity from chain(s) into desired destination.

The protocol is designed to work on top of existing cross-chain protocols and principles without interference with how the networks and their protocols operate. This means that our protocol mainly sources liquidity on other chains and swaps the currency through the existing chain interface to then send it to the end user through the mentioned existing cross-chain protocols. Liquidity pools are handled by sovereign chains and their policies, and they build their own asset risk management of asset pool liquidity. This is what makes LiquiSpell universal and special compared to other protocols.

To explain the principles of all components, let's create an intent to buy a non-fungible token for DOT on AssetHub. The journey starts with the case where we do not have a desired balance of DOT on AssetHub. Therefore, we should teleport the assets from the origin chain to AssetHub; in this case, the origin chain for DOT is Polkadot. If we want liquidity sourcing for different assets (like GLMR), look to the Moonbeam chain. Suppose Liquidity Sourcing needs to find more balance in the

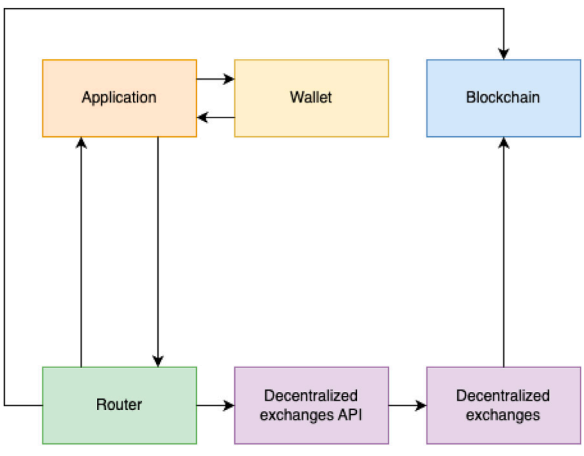

FIGURE 7
High-level overview of solution architecture.

10.3389/fbloc.2024.1413840

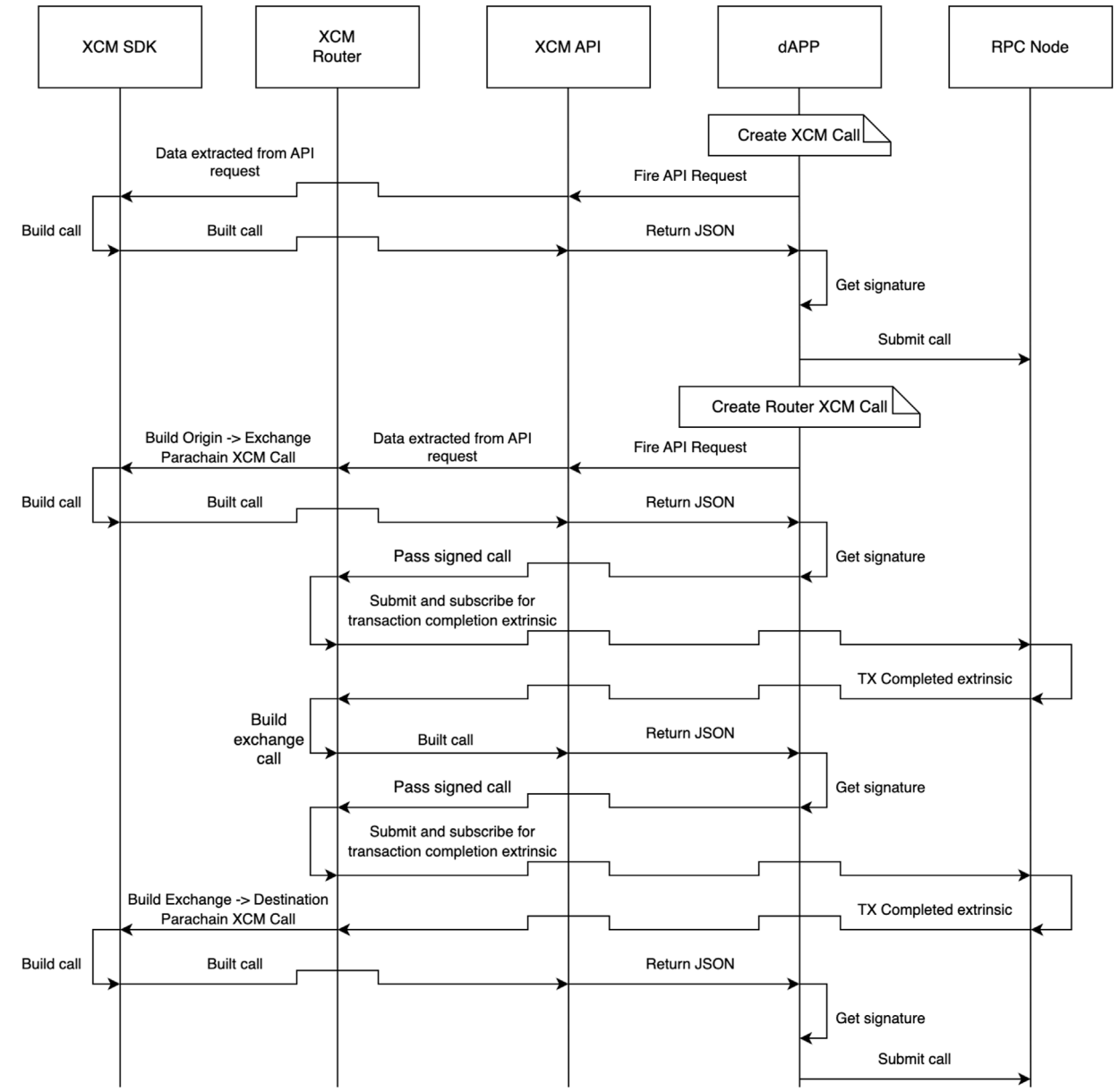

FIGURE 8
Sequential functionality of solution.

origin chain. In that case, it extends the search on other parachains that have DOT registered[6]. If we find the desired amount, we will take the DOT from other chains to AssetHub. Our optimal strategy is using the least amount of transactions[7]. Using the auto teleport feature, we take the DOT from other chains, and in a few minutes and a few signatures, we have finally reached the desired amount on AssetHub to buy an NFT. If we fail to have DOT, we have developed an XCM-based asset routing to buy non-fungible assets on different chains with different currencies. It acts as an adapter for cross-chain asset exchange. The difference from other adapters is that the solution does not require multiple steps to execute the operation of the cross-chain exchange.

As visible in the high-level architecture overview in Figure 7. As we operate with the logged-in wallet within the application and the liquidity souring decides to swap tokens into DOT, the Asset router finds the best possible exchange price using the API of decentralized exchanges within the Polkadot ecosystem. Once we sign the transactions, we will swap tokens, and through AutoTeleport, it will be delivered to the destination chain. Tue solution benefits from a simple design that can be easily upgraded, as it works directly with blockchain and decentralized exchanges APIs. Direct blockchain utilization

6 The standard practice is that DOT is registered with id 0.

7 With respect to Existential deposit.

happens when sending cross-chain messages, while API usage occurs when asset exchange happens or during best-price outcome queries.

The functionality can also be observed sequentially in Figure 8. As explained in the sequential diagram, the routing happens in the following order:

- Building cross-chain message from origin chain to exchange chain.
- Exchanging asset.
- Building cross-chain message from exchange chain to destination chain.

The Asset Router finds and selects the best exchange. First, it checks for exchanges that offer an asset pool with the requested assets. Then, the asset router checks the pool to see if it contains enough liquidity and gives the best output amount with minimal slippage. Then, the selected pool is used as a route using the exchange chosen chain.

Exchange selection is possible by implementing open-source libraries provided by decentralized exchanges. These libraries allow for on-chain exchange execution, best-price, and asset-pool queries. We have designed the Asset Router to allow us to add implementations of other libraries should new decentralized exchanges be effectively implemented.

Granted that all asset pools implemented in our solution are fully functional, we are searching for liquidity on 579 available asset pools, a number that is likely to increase in the future. With so many available pools, we are the largest liquidity aggregation solution in the Polkadot network, with the ability to expand to other ecosystems and heterogeneous chains.

The main goal is to provide a simple, unified way of bridging liquidity cross-chain across diverse ecosystems. We further abstract cross-chain communication by providing asset swap functionality during cross-chain message passing. Similarly to Moonbeam SDK mentioned in Section 2 2 we have implemented asset teleportation in all three ways (Abbas et al., 2022): From Parachain to Relay Chain (UMP), From Relay Chain to Parachain (DMP) and from Parachain to Parachain (HRMP).

However, the universal transaction module plays a major role in our system. Currently, different parachain teams utilize many assets and XCM pallets in various combinations. The major pallets are xTokens, polkadotXCM, orlmXTokens, and palletXCM. In addition, many ecosystem teams are tweaking the modules to meet their unique needs. While this shows the Substrate framework's great modularity, it unnecessarily adds another complexity vector for developers integrating support for multiple chains.

The universal transaction works as follows. We see a typescript-based call on our Asset Router in the Listing 4. To compose a swap from Astar's USDT to Polkadot AssetHub and receive DOT, we need to call *RouterBuilder*. Using the Builder pattern, which we found to be the most flexible to compose the calls, we need to specify *from, to, currencyFrom, currencyTo* additionally with *amount* and *slippage*. Now, we need to sign a few transactions and wait for finalizations, and we are ready to use the DOT on AssetHub.

```
await RouterBuilder
      .from('Astar') //Origin Parachain/Relay Chain
      .to('AssetHubPolkadot')          //Destination
      Parachain/Relay Chain
      .currencyFrom('USDT') // Currency to send
      .currencyTo('DOT') // Currency to receive
      .amount('1000000') // Amount to send
      .slippagePct('1') // Max slippage percentage
      .injectorAddress(selectedAccount.
      address) //Injector address
      .recipientAddress(recipientAddress)
      //Recipient address
      .signer(injector.signer) //Signer
      .onStatusChange((({status,   hashes,   type}):
      TTxProgressInfo)    =>    console.log(status,
      hashes, type))
      .buildAndSend()
```

Listing 4. Constructing call for Asset Router.

The universality question remains unanswered. Astar uses both WASM and EVM virtual machines, Moonbeam is completely EVM-based, and other parachains use custom variations of XCM and asset pallets, but the call will still look the same. To encapsulate the whole idea, when we intend to make

TABLE 4 Comparison of LiquiSpell with other solutions.

	UniswapX	Axelar GMP	Polkadot XCM	LiquiSpell
Usable in multiple ecosystems	No	Yes	No	Yes
Bridged tokens	Wrapped	Wrapped	Native	Native
Implemented liquidity sourcing	Yes	No	No	Yes
Ability to prioritize coins	No	No	No	Yes
Plug and play architecture	No	Yes	Yes	Yes
Routing strategy	Automatic	Manual	Manual	Automatic
Select best exchange	Yes	No	No	Yes
Route tokens from other ecosystems	No	Yes	No	Yes
Required trusted third party	Yes	No	No	No

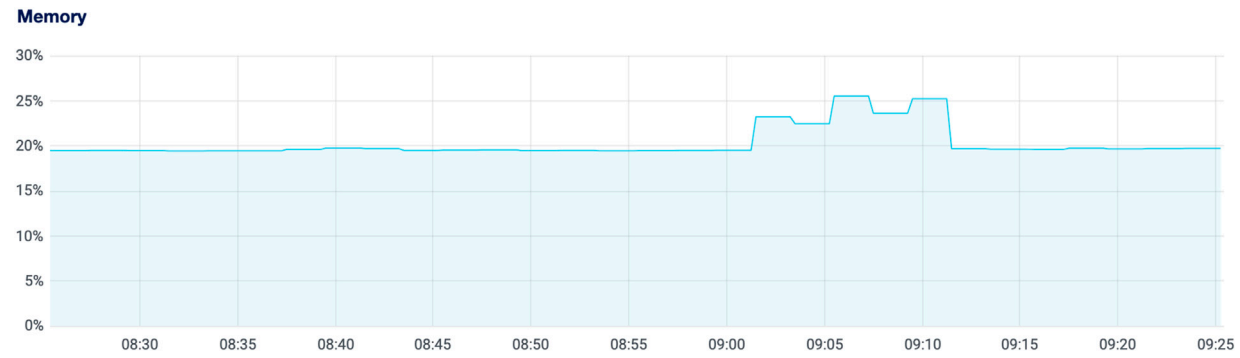

FIGURE 9
Router API memory management during first request batch.

an operation that would need to unify liquidity on one chain, we are sourcing the tokens from the origin and other chains in the ecosystem. After that, we try to find an optimal way to bring them to the destination, and if we fail to accomplish this, we can always use Asset Router to swap and route tokens from one chain to another. Therefore, from the insane complexity of liquidity unification and XCM, we got a solution that is easy to use for novel use cases or the use cases that would be missing particular liquidity logic on the specific chain.

To address the advantages of the LiquiSpell, we should observe the comparison in Table 4. As we can see, the LiquiSpell is the only solution usable in multiple ecosystems, thus not limited only to EVM or its native Polkadot system, but can also be reused in the Cosmos ecosystem along with IBC. We have also designed liquidity sourcing with token prioritization; this way, existing decentralized applications are not forced to implement multi-currency support; for instance, NFT marketplaces can use their preferred currency (DOT), and if we own only USDT, the system, thanks to the automatic routing strategy, will automatically give us the best way to obtain DOT on the background. Thanks to the modular design and universal transactions, we do not see a limitation.

4 Evaluation

The solution was evaluated using experiments that were carried out with:

- SDK tests: Macbook PRO 15 2017 running the MacOS 13.6 Ventura system and equipped with 16 GB RAM and a 4-core Intel core I7-7820HQ CPU running at 2.8 GHz.
- API tests: Server deployed on Digital Ocean running Linux with an intel CPU and 8 GB of RAM.

This setup closely simulates the server power most projects implementing XCM API or XCM SDK must allocate for smoother performance. Both the XCM SDK and API are written in TypeScript. As we mentioned before, the XCM API uses a framework called NestJS. Our work, mainly SDK (ParaSpell XCM SDK GitHub repository) and API (ParaSpell XCM API GitHub repository), is fully open source with an MIT license, and their repositories are accessible on GitHub.

During testing, we focused on four leading key performance indicators (KPIs): scalability, availability, functionality, and performance of the proposed solution. The tests also analyzed the solution's throughput if it was under heavy use. We wanted to test our solutions and selected three testing scenarios thoroughly.

4.1 Evaluation of high load situation

The first scenario involved sending 1,000 requests to the Router API quickly. This test was performed ten times; therefore, we created 10,000 requests. The API's process of generating calls can be observed in Figure 10 while the complexity of each request is shown in Equation 1. This test also analyzed how API performed in a high-load situation and whether memory was efficient enough to meet demands.

$$t = \frac{tb + delay}{proc.power} \tag{1}$$

where:

- t - time required,
- tb - time required for PolkadotJS library tasks (creating allet address to hex conversion, for example),
- delay - time delay between receiving and sending the request,
- proc. power - Processing power that can increase or decrease the final time required.

Memory usage during Router API testing was very stable, and memory usage of the first 1,000 request batch can be observed in Figure 9. Requests were processed between 9:06 and 9:11. The result of each batch can be seen in Table 5.

The second scenario involved continuous requests until the counter reached 10,000. The goal of this scenario was to analyze overall stability and availability. The time complexity of this test was the same as in the previous test, so Equation 1 for the complexity of the request also applies to this test. The request process can be found in Figure 10. The result grouped into four batches of 2,500 requests is visually available in Table 6.

TABLE 5 Batch of test 1 requests.

Batch no.	Failed	Median (ms)	Min (ms)	Max (ms)	Avg. size (Bytes) (B)
Batch 1	0	558	450	1,193	354
Batch 2	0	602	447	952	354
Batch 3	0	838	515	1,403	354
Batch 4	0	639	348	881	354
Batch 5	0	931	500	1703	354
Batch 6	0	742	501	901	354
Batch 7	0	768	468	1,300	354
Batch 8	0	872	403	1,526	354
Batch 9	0	734	399	1,387	354
Batch 10	0	867	487	1873	354

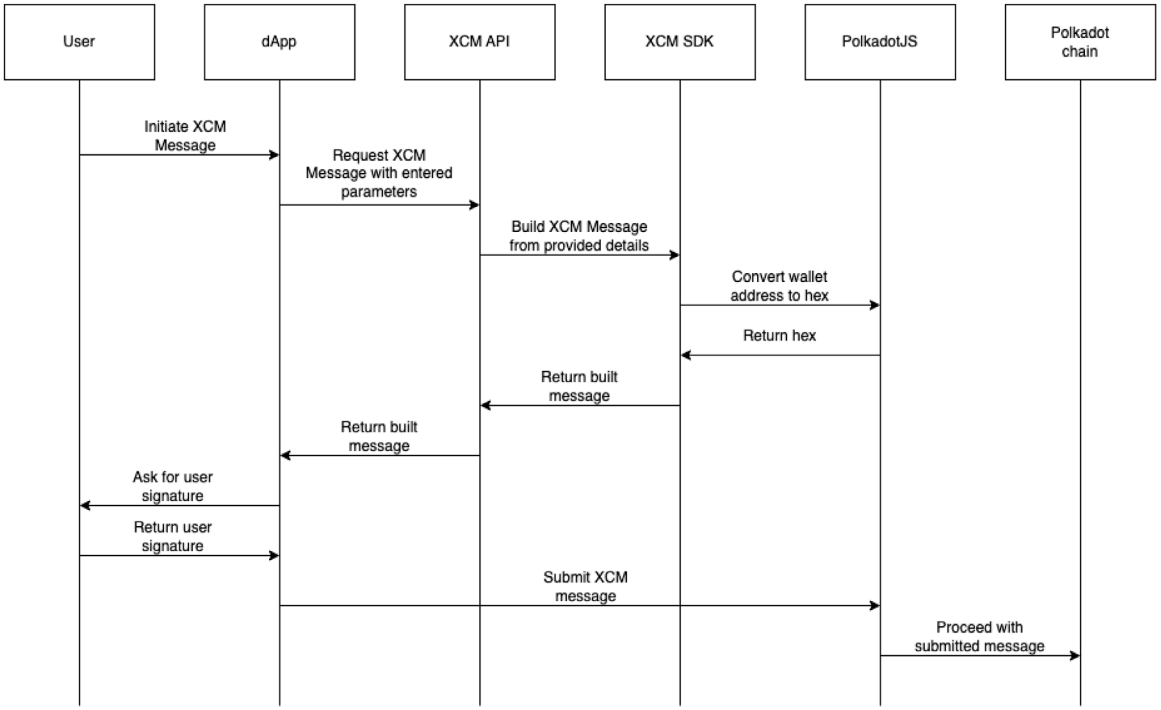

FIGURE 10
Process of XCM message within our solution.

4.2 Evaluation of feasability

To evaluate our state-of-the-art solution, we have decided to implement it into the most significant open-source application in the Polkadot ecosystem, KodaDot (Teleport section; Sonia, 2023), which is an NFT marketplace that allows cross-chain transfers and utilizes the advantages of liquidity unification by using NFTs on AssetHub without owning a DOT. Figure 11 shows the KodaDot interface to mint an NFT. Because we do not have enough balance, the app found out that we have enough DOT liquidity on the Relay Chain, and it is the most optimal route to get and use it.

After that, we sign the first step of teleportation as shown in Figure 11. We will need to wait for the finalization a few blocks after the funds arrive at AssetHub. The application can automatically follow with a function to mint a non-fungible token.

The test objective was to create successful cross-chain transfers from two Parachains with different XCM pallets implemented to compare the ability of abstraction in interfaces. The tests were concluded with ten users who work with IT technologies professionally and previously used

TABLE 6 Batch of test 2 requests.

Batch no.	Type	Name	Median (ms)	Average (ms)	Min (ms)	Max (ms)	Average request size (Bytes)	Failed
Batch 1	GET	/router?from = Polkadotand to = BifrostPolkadot& currencyFrom = DOT& currencyTo = BNC& amount = 1,000,000,000,000& recipientAddress = 5F5586 mfsnM6durWRLptYt3jSUs 55KEmahdodQ5tQMr9iY96& injectorAddress = 5F5 586mfsnM6durWRLptYt3jSU s55KEmahdodQ 5tQMr9iY96& slippagePct = 1	745	745	482	1,239	354	0
Batch 2	GET	Same as Batch 1	761	760	426	1,692	354	0
Batch 3	GET	Same as Batch 1	1,061	1,060	573	1,520	354	0
Batch 4	GET	Same as Batch 1	883	880	450	1,299	354	0

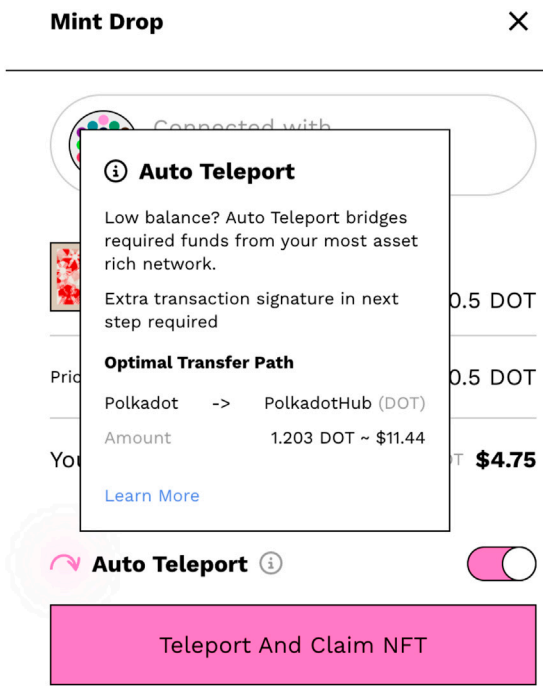

FIGURE 11
Kodadot user interface asking auto teleport DOT from Relay Chain into the AssetHub.

XCM to perform cross-chain transfers on Polkadot. Test results are shown in Table 7.

5 Discussion

We designed a solution that solves many problems with the diversity of cross-chain sharing protocols implemented by Parachains on the Polkadot network. Our study has shown that Parachains can use different or modified versions of cross-chain sharing modules. Moreover, with the wide variability of virtual machines in the ecosystem, creating one universal solution to solve liquidity fragmentation was a problem. Our solution is fully developed and ready to use. Research to develop this solution can be further used to solve similar problems with diversity in other multichain ecosystems. The solution is designed to group all XCM compatible Parachains in a unified matter, which allows for subtracting complex logic from developers that can benefit from:

- Saved time,
- Well-documented reliable solution for integrating XCM into their applications,
- Simple and abstract implementation,
- Completely free and open source nature of the solution[8].

One possible attack vector is running a DDoS attack on our API. Fortunately, the solution is also designed to support private deployment, greatly enhancing decentralization and providing transparency. Additionally, LiquiSpell uses direct RPC endpoints, so changing the provider is a matter of configuration. Therefore, we can boost network economics with new business models if we use incentives for users to build useful paid API add-ons or deploy solutions to high-performance servers for further renting of power to projects that need the fastest responses they can get. Aggregation and improvement solutions can also be refined using a built-in analytic tool that does not collect sensitive user data.

While creating this solution, our biggest concern was keeping it secure and users' privacy untouched. Security is vital for every blockchain application, as history shows that security breaches can lead to billions in losses. Our solution strictly uses only the network's native cross-chain protocol and modules. Liquidity pools are managed by chains themselves, meaning they implement their risk management strategies regarding liquidity.

Different blockchains have varying transaction throughput and block confirmation times. For example, Bitcoin's block time is around 10 min, while Solana processes transactions much faster. This disparity introduces challenges in synchronizing asset transfers

8 https://github.com/kodadot/nft-gallery/tree/main/composables/autoTeleport

TABLE 7 Test 3 comparing project implementing our solution to interface that is default to Polkadot

User	KodaDot with XCM SDK	PolkadotJS UI without SDK
1	32.08	216.27
2	35.78	152.31
3	36.84	142.61
4	30.96	225.21
5	39.43	232.68
6	43.34	138.35
7	44.16	137.22
8	41.07	224.46
9	37.71	208.86
10	38.27	235.03
Avg.	37.96	191.30
Mean	37.99	212.57

and maintaining liquidity pools. In the context of our LiquiSpell protocol, which operates within the Polkadot ecosystem, this challenge is somewhat mitigated due to the nature of Polkadot's parachains. All parachains in the Polkadot network produce blocks at the same rate, which helps maintain synchronization. However, we acknowledge that this becomes a more significant issue when considering potential future expansions to other blockchain ecosystems.

Currently, there are not that many solutions similar to ours. As we mentioned previously, projects like UniswapX or Axelar are doing an outstanding job of unifying liquidity; however, our state-of-the-art solution offers the whole experience of wrapping the hard-to-swallow logic of XCM into the SDK and API. There are few SDKs in the Polkadot ecosystem, so our all-in-one solutionis first in the ecosystem. Many developers consider cross-chain communication on Polkadot to be a complicated topic. We expect new versions of XCM will make the diversity gap even more significant. Therefore, we believe that grasping this problem as soon as possible is the key to creating a vibrant ecosystem. Furthermore, we must continue to investigate and improve the throughput and efficiency of LiquiSpell, as demand will increase with time. As our analysis from the previous section outlines, the solution delivers stable results, even though we forced it to process 10,000 requests quickly and did not return any failed requests. The results can be observed in Tables 5, 6. The response time is relatively standard for European servers, considering the median ranges from 558 to 1,061 ms, respectively. In Figure 9, we can see that the number of requests mentioned only slightly fills the server memory. We can also see a marginal difference in the cross-chain capabilities of users using applications that implement our solution compared to the standard PolkadotJS UI. Table 7 shows the difference in the amount of time users try to buy an NFT using cross-chain liquidity, unified in a way that allows them to seamlessly use the Kodadot mint UI 11, which implements the LiquiSpell SDK solution and was able to abstract a lot of complexity from users to allow them to quickly transfer their funds, as shown in Figure 12.

Some limitations are present. As we try to unify liquidity, some decentralized exchanges can become congested, creating higher fees. This can be solved by using alternate routes or distributing the swap amount to multiple exchanges. The economic impacts of our solution are undeniable. In the case of a sharing economy, e.g., peer-to-peer car sharing (Valastin et al., 2019) that uses platform-specific tokens to pay for rental services, we do not need to find the token, swap it, and then use it. Instead, the app can use our cross-chain liquidity to find the best exchange rate, and with a few signatures, we can create a smooth user experience.

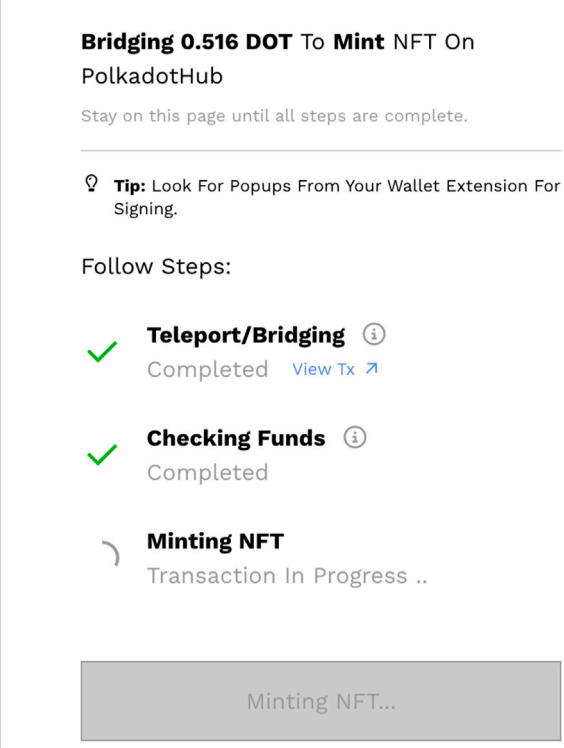

FIGURE 12
AutoTeleporting DOT from Relay Chain to mint NFTs.

5.1 Security

LiquiSpell solution inherently benefits from the robust security measures built into the Polkadot ecosystem, particularly its shared security model and the use of XCMP. Regarding double-spending attacks, Polkadot's relay chain provides a unified security layer for all connected parachains. This means that the security of cross-chain transactions is not solely dependent on individual chains but is upheld by the entire network of validators. The XCMP protocol ensures that messages (including asset transfers) between chains are processed only once and in the correct order, significantly mitigating the risk of double spending. For replay attacks, we utilize the inherent properties of XCM messages, which inc for replay attacks, including identifiers and versioning. This ensures that each cross-chain message can only be executed once. Additionally, our protocol implements nonce-based transaction signing, further preventing replay attacks.

6 Conclusion

In this paper, we introduced LiquiSpell, a groundbreaking protocol designed to address the challenges of liquidity fragmentation and interoperability in multichain ecosystems, particularly within the Polkadot network. LiquiSpell's novel approach, centered around a universal transaction module, enables seamless cross-chain asset transfers and liquidity unification across heterogeneous parachains. The universal transaction module is a significant advancement in cross-chain communication. It allows for constructing transactions compatible with any parachain, regardless of its underlying architecture or asset management pallet. This innovation abstracts the complexities of interacting with diverse parachains, providing developers a streamlined and efficient way to integrate XCM functionalities into their dApps. Compared to state-of-the-art solutions, we provide a protocol for unifying liquidity in the Polkadot ecosystem, leveraging native cross-chain communication and intents. As users, we do not need to know how tokens are unified, but we want to see the result. Our solution does not require any trusted third party, as anyone can replicate our API, and we do not introduce incompatible standards. Moreover, LiquiSpell incorporates cutting-edge liquidity aggregation and routing mechanisms, which intelligently source liquidity from multiple parachains, determine optimal transfer pathways, and facilitate the seamless movement of tokens across the Polkadot ecosystem. This advanced liquidity management system ensures the efficient allocation and availability of assets where they are needed most. We have demonstrated LiquiSpell's exceptional performance, scalability, and usability through extensive testing and evaluation. The protocol has proven capable of handling high transaction throughput, with 10,000 requests processed without any failures, showcasing its reliability under heavy load. Additionally, LiquiSpell achieves impressive response times, with median values ranging from 558 ms to 1,061 ms and average times spanning 745 ms to 1,060 ms across different test batches. Implementing LiquiSpell in the KodaDot NFT marketplace further highlights its practical applicability and potential to revolutionize cross-chain liquidity management. Users experienced significantly faster cross-chain message creation when utilizing the LiquiSpell-powered KodaDot teleport UI compared to the standard PolkadotJS UI, demonstrating the tangible benefits of our solution. Looking ahead, we aim to explore the potential of extending LiquiSpell to other multichain ecosystems, such as Cosmos, by leveraging the Inter-Blockchain Communication (IBC) protocol. While adapting to Cosmos' unique characteristics may present challenges, we are confident in designing a robust and compatible solution. Future work will also focus on optimizing the memory management of the LiquiSpell API, enhancing error analytics for improved developer debugging, and staying up-to-date with the latest XCM versions and parachain updates. Through ongoing collaboration with parachain teams and the broader community, we strive to refine and expand LiquiSpell's capabilities continuously, driving adoption and fostering innovation within the cross-chain ecosystem. In conclusion, LiquiSpell represents a significant leap forward in addressing liquidity fragmentation and interoperability challenges in multichain environments. By providing a universal and efficient solution for cross-chain asset transfers and liquidity unification, LiquiSpell has the potential to revolutionize the way decentralized applications interact and operate across heterogeneous blockchain networks. As we continue to push the boundaries of cross-chain technology, LiquiSpell serves as a foundation for a more connected, fluid, and vibrant decentralized future.

Data availability statement

The original contributions presented in the study are included in the article/supplementary material, further inquiries can be directed to the corresponding authors.

Author contributions

VV: Data curation, Formal Analysis, Investigation, Methodology, Resources, Software, Validation, Visualization, Writing–original draft, Writing–review and editing. DM: Data curation, Investigation, Software, Validation, Visualization, Writing–original draft. KK: Conceptualization, Funding acquisition, Methodology, Project administration, Supervision, Writing–review and editing. IK: Funding acquisition, Project administration, Supervision, Writing–review and editing.

Funding

The author(s) declare that financial support was received for the research, authorship, and/or publication of this article. This work was supported by the Slovak Research and Development Agency under Contract No. APVV-20-0338.

Conflict of interest

The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Publisher's note

References

Abbas, H., Caprolu, M., and Pietro, R.Di (2022). "Analysis of polkadot: architecture, internals, and contradictions," in Proceedings of 2022 IEEE International Conference on Blockchain, 61–70. doi:10.1109/BLOCKCHAIN55522.2022.00019

Augusto, A., Belchior, R., Correia, M., Vasconcelos, A., Zhang, L., and Hardjono, T. (2024). "SoK: security and privacy of blockchain interoperability [extended version]," in *TechRxiv*. doi:10.36227/techrxiv.24595764.v4

Austin Adams UniswapX whitepaper. Available at: https://uniswap.org/whitepaper-uniswapx.pdf (Accessed February 13, 2024).

Axelar GMP overview. Available at: https://docs.axelar.dev/dev/gmp-overview (Accessed October/11/2023).

Axelar Team (2021). Axelar network: connecting applications with blockchain ecosystems. Available at: https://www.axelar.network/whitepaper (Accessed April 21, 2024).

Brendan, F. Aggregated blockchains. Available at: https://mirror.xyz/ (Accessed February 13, 2024).

Burdges, J., Cevallos, A., Czaban, P., Habermeier, R., Hosseini, S., Lama, F., et al. (2020). Overview of polkadot and its design considerations. arXiv: 2005.13456 [cs.CR]. doi:10.48550/arXiv.2005.13456

Cao, L., and Song, Bo. (2021). "Blockchain cross-chain protocol and platform research and development". In: 2021 International Conference on Electronics, Circuits and Information Engineering (ECIE). (2021). pp. 264–269. doi:10.1109/ECIE52353.2021.00063

Capponi, A., Jia, R., and Zhu, B. (2024). The paradox of just-in-time liquidity in decentralized exchanges: more providers can sometimes mean less liquidity. 2024. arXiv: 2311.18164 [q-fin.GN].

Cosmos IBC documentation. Available at: https://ibc.cosmos.network/main/ibc/apps/ibcmodule#packet-callbacks (Accessed February/22/2024).

Eizinger, T., Hoenisch, P., and Pino, L. S. del (2021). Open problems in cross-chain protocols. *arXiv 2101.12412 [cs.CR]*. doi:10.48550/arXiv.2101.12412

Giulio, C. (2021). Wrapping trust for interoperability: a preliminary study of wrapped tokens. *Inf. 2022* 13, 6–13. ISSN 2078-2489. doi:10.3390/INFO13010006

Jakub Gregus/HydraDX (2022). Economic efficiency as the killer feature of DotSama @ AmsterDOT conf hack 2022. Available at: https://www.youtube.com/watch?v=azwkJy6itqc (Accessed May 25, 2024).

Kanani, J., Nailwal, S., and Arjun, A. (2021). Matic whitepaper. Available at: https://www.allcryptowhitepapers.com/wp-content/uploads/2019/04/matic-litepaper.pdf (Accessed February 13, 2024).

Kim, J., Essaid, M., and Ju, H. (2022). "Inter-blockchain communication message relay time measurement and analysis in Cosmos," in *2022 23rd asia-pacific network operations and management symposium (APNOMS)*, 1–6. doi:10.23919/APNOMS56106.2022.9919970

Kwon, J., and Buchman, E. Cosmos whitepaper: a network of distributed ledgers. Available at: https://cosmos.network/resources/whitepaper (Accessed December 10, 2023).

Lagadamane Dinesha, D., and Patil, B. (2023). "Achieving interoperability in heterogeneous blockchain users through inter-blockchain communication protocol," in *TechRxiv*. doi:10.36227/TECHRXIV.21532953.V1

Lee, S.-S., Murashkin, A., Derka, M., and Gorzny, J. (2023). "SoK: not quite water under the bridge: review of cross-chain bridge hacks," in 2023 IEEE International Conference on Blockchain and Cryptocurrency (ICBC), Dubai, United Arab Emirates, 1–14. doi:10.1109/ICBC56567.2023.10174993

Lehar, A., Parlour, C., and Zoican, M. (2024). Fragmentation and optimal liquidity supply on decentralized exchanges. arXiv: 2307.13772 [q-fin.TR].

Lin, S., Kong, Y., Nie, S., Xie, W., and Du, J. (2021) "Research on cross-chain technology of blockchain". In: 2021 6th International Conference on Smart Grid and Electrical Automation (ICSGEA), Kunming, China, 405–408. doi:10.1109/ICSGEA53208.2021.00098

Lucas García De Viedma Pérez (2023). Development and optimization of ETL processes for blockchain data analystics. Available at: https://aaltodoc.aalto.fi/handle/123456789/122839 (Accessed February 13, 2024).

Moonbeam official website. Available at: https://moonbeam.network/ (Accessed October/12/2023).

Moonbeam XCM SDK documentation. Available at: https://docs.moonbeam.network/builders/interoperability/xcm/xcm-sdk/v1/xcm-sdk/ (Accessed October/11/2023).

Moonbeam, X. C. M. S. D. K. Available at: https://github.com/PureStake/xcm-sdk (Accessed October 11, 2023).

Morháč, D., Valaštín, V., Košťál, K., and Kotuliak, I. (2023a). "Enhancing XCMP interoperability across polkadot paraverse," in 2023 IEEE International Conference on Blockchain and Cryptocurrency (ICBC), Dubai, United Arab Emirates (IEEE), 1–3. doi:10.1109/ICBC56567.2023.10174872

Morháč, D., Valaštín, V., Košťál, K., and Kotuliak, I. (2023b). "ParaSpell XCM sdk: a new protocol for interoperability in polkadot paraverse," in 2023 Fifth International Conference on Blockchain Computing and Applications (BCCA), Kuwait, 569–576. doi:10.1109/BCCA58897.2023.10338906

Morháč, D., Valaštín, V., and Koštál, K. (2022). "Sharing fungible assets across polkadot paraverse," in 2022 International Conference on Electrical, Computer and Energy Technologies (ICECET), 1–7. doi:10.1109/ICECET55527.2022.9872938

ParaSpell XCM API GitHub repository. Available at: https://github.com/paraspell/xcm-api (Accessed September/28/2023).

ParaSpell XCM SDK GitHub repository. Available at: https://github.com/paraspell/xcm-sdk (Accessed September/28/2023).

Parity asset transfer API Documentation. Available at: https://paritytech.github.io/asset-transfer-api/ (Accessed October/12/2023).

Parity asset transfer API Registry repository. Available at: https://github.com/paritytech/asset-transfer-api-registry (Accessed October/12/2023).

Parity asset transfer API repository. Available at: https://github.com/paritytech/asset-transfer-api (Accessed October/12/2023).

Polkadot architecture overview. Available at: https://banklesspublishing.com/the-polkadot-primer/ (Accessed October/12/2023).

PolkadotJS Apps package repository. Available at: https://github.com/polkadot-js/apps (Accessed October/13/2023).

Polygon AggLayer - polygon knowledge layer. Available at: https://docs.polygon.technology/learn/agglayer/#proof-aggregation (Accessed February/13/2024).

Pupyshev, A., Sapranidi, I., Dzhafarov, E., Khalilov, S., Teterin, I., et al. (2020). Graviton: interchain swaps and wrapped tokens liquidity incentivisation solution. arXiv: 2009.05540 [cs.CR].

Qasse, I. A., Abu Talib, M., and Nasir, Q. (2019). "Inter blockchain communication: a survey," in Proceedings of the ArabWIC 6th Annual International Conference Research Track. ArabWIC 2019 (New York, NY, USA: Association for Computing Machinery), 1–6. doi:10.1145/3333165.3333167

Qin, K., Zhou, L., Gamito, P., Jovanovic, P., and Gervais, A. (2021). "An empirical study of DeFi liquidations: incentives, risks, and instabilities". In: Proceedings of the 21st ACM Internet Measurement Conference (IMC '21) (New York, NY: Association for Computing Machinery), 336–350. doi:10.1145/3487552.3487811

Send a wrapped native token from an EVM chain back to its home EVM chain. Available at: https://docs.axelar.dev/dev/axelarjs-sdk/token-transfer-dep-addr (Accessed February/13/2024).

Siniscalchi, A., Mateos, T., and Evans, A. (2023). Laos: vision for a scalable, truly non-custodial, dynamic NFT protocol with universal minting and evolution. Available at: https://spaces.gorengine.com/laos/laos_whitePaper.pdf (Accessed April 25, 2024).

Sonia, M. "Application of metaverse and its underlying challenges in the 21st century". In: ed *Bahaaeddin el khoury rim and alareeni*. Springer Nature Singapore, 2023, pp. 195–205. doi:10.1007/978-981-99-5126-0\text\17

Teleport section - KodaDot NFT marketplace. Available at: https://kodadot.xyz/ahk/teleport (Accessed September/28/2023).

Understanding interchain transaction time. Available at: https://docs.axelar.dev/learn/txduration (Accessed February/13/2024).

Valaštín, V., Košťál, K., Bencel, R., and Kotuliak, I. (2019). "Blockchain based car-sharing platform," in 2019 International Symposium ELMAR, Zadar, Croatia, 5–8. doi:10.1109/ELMAR.2019.8918650

Wan, X., and Adams, A. (2022). "Just-in-time liquidity on the Uniswap protocol," in *SSRN electronic journal*. doi:10.2139/SSRN.4382303

Whitton, J. (2021). EAVE parachain design. Available at: https://johnwhitton.com/assets/images/EAVEParachainDesign.pdf (Accessed May 25, 2024).

Wood, G. (2016). *Polkadot: vision for a heterogeneous multi-chain framework*. White paper, 21.

Wood, G. XCM: the cross-consensus message format. (2023). Available at: https://polkadot.network/blog/xcm-the-cross-consensus-message-format (Accessed February 13, 2024).

Xie, T., Zhang, J., Cheng, Z., Zhang, F., Zhang, Y., Jia, Y., et al. (2022). "ZkBridge: trustless cross-chain bridges made practical," in Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security. CCS '22 (Los Angeles, CA, USA: Association for Computing Machinery), 3003–3017. doi:10.1145/3548606.3560652

Yousaf, I., Pham, L., and Goodell, J. W. (2024). Dynamic spillovers between leading cryptocurrencies and derivatives tokens: insights from a quantile VAR approach. *Int. Rev. Financial Analysis* 94, 103156. doi:10.1016/J.IRFA.2024.103156

Zhang, P., Xiaoqi, H., and Zhu, H. (2023). Cross-chain digital asset system for secure trading and payment. *IEEE Trans. Comput. Soc. Syst.* 11 (2 Apr. 2024), 1654–1666. ISSN 2329924X. doi:10.1109/TCSS.2023.3241065