Group-aware Resource and Path Discovery for Collaborative Ultra-Low Latency Applications

Hendrik Mahrt, Martina Zitterbart
Institute of Telematics, Karlsruhe Institute of Technology (KIT)
{hendrik.mahrt, martina.zitterbart}@kit.edu

***Abstract*—Support for applications like collaborative extended reality and multi-sensory communication in 6G demands ultra-low end-to-end latency guarantees for a group of interacting users. Placement of such applications requires discovering appropriate edge resources *and* paths across mobile networks of different operators or Internet-based edge clouds. This poster outlines *Sherpa*, a novel distributed resource discovery mechanism that especially addresses finding latency-constrained, cross-provider paths to all users of a group.**

***Index Terms*—Resource Discovery, Service Placement, 6G, QoS, Collaborative XR, Multi-sensory communication**

I. Introduction

Next-generation applications like collaborative extended reality and multi-sensory communication form part of the use cases planned for the sixth generation of mobile networks [1], [2]. The collaborative aspect and high level of interactivity (e.g., tactile feedback) between users of these applications requires ultra-low latency end-to-end, i.e., between all users of a group in an application. Assuming the application is executed on some edge resource and employs a client-server architecture, communication follows a *user - resource - user* pattern (see fig. 1). To achieve ultra-low latency between a group of users in that case requires placement of the application such that propagation delay to all users is low enough to fulfill the end-to-end latency. Discovery of a suitable resource therefore needs some form of awareness for group-wide latency constraints. While existing, centralized solutions could be adapted to support awareness for group-wide latency constraints, we identify another requirement in the 6G context: To enable customers to interact with friends, family or colleagues that are with different mobile network operators (MNO), those would need to cooperate. Additionally, there is future potential for some form of convergence with edge clouds located on the Internet [3]. To accomplish scalable discovery across operators and networks we argue in favor of a decentralized approach.

In this poster we outline *Sherpa*, a novel group-aware resource and path discovery scheme especially designed to support collaborative ultra-low latency applications in 6G.

II. Sherpa

Sherpa discovers a set of resources that meets the application's requirements while also holding the end-to-end latency within a group. This includes corresponding paths between

ISBN 978-3-903176-72-0 © 2025 IFIP

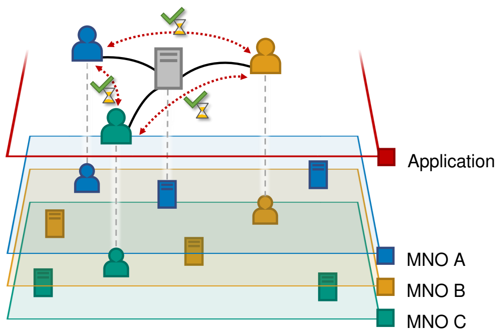

Fig. 1. Users and resources spread across MNO. Group-aware resource discovery considers end-to-end latency constraints between all pairs of users.

users and resource. Each *path* is constructed from a sequence of *path segments* that can be used to set up a tunnel or allows for segment routing. Sherpa uses an overlay to efficiently discover resources. The construction of this overlay is not in the scope of this poster. However, the next section briefly outlines its structure.

A. Hierarchical Overlay Abstraction

Sherpa assumes a hierarchical overlay spanning across all networks that participate in Sherpa. We first assume each participating MNO network to be partitioned into some form of areas on the granularity of routers and links. We call these lowest areas *physical areas*. Figure 2 shows how each network is split up into such physical areas (e.g., a, b and c).

Next, adjacent physical areas are clustered into *super areas* (we further simply write *area* if we do not differentiate between physical and super area). This clustering can be done using existing algorithms, e.g. as used in mobile adhoc networks. These neighboring areas do not necessarily belong to a single MNO. In our example the physical areas a, b and c of different MNOs form the super area $S_{1,1}$. Adjacent super areas in turn form another super area, e.g., $S_{1,1}$ forms $S_{2,1}$ with other super areas (not shown in the figure). This step is repeated until a single, all-encompassing super area emerges. The result is a hierarchy as shown in the inset in fig. 2. The number of levels in this hierarchy directly depends on the number of physical areas and how many areas are aggregated into a super area in each step.

In each area a Sherpa instance is running on one node. We call these responsible sherpa nodes (RSN), as they process

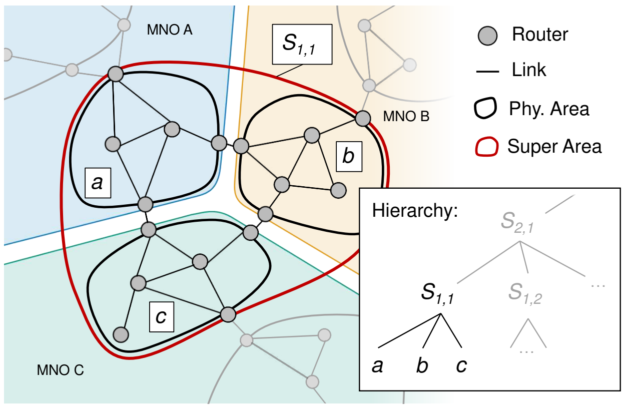

Fig. 2. MNO networks are partitioned into (physical) areas. Areas of different MNOs form super area. Super areas themselves can form super areas yielding a hierarchy.

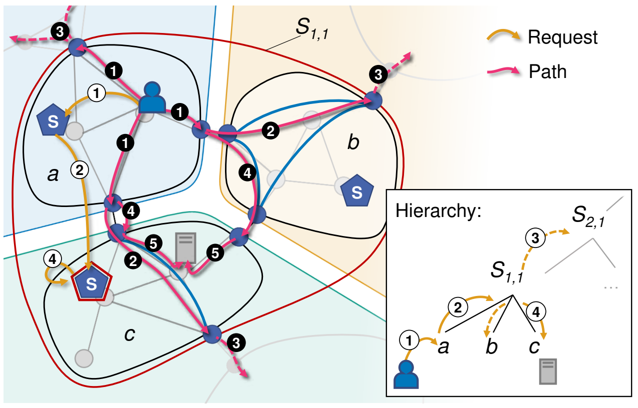

Fig. 4. Paths are constructed extending from group member while going upwards in hierarchy. Branching off and traveling downwards in the hierarchy extends paths towards resources.

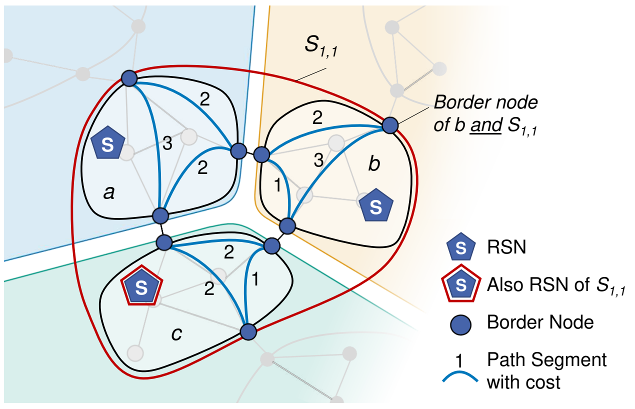

Fig. 3. On super area level: sub areas abstracted away leaving path segments between all pairs of sub areas border nodes.

requests for that area. While RSNs of physical areas need to hold a full view on the underlying topology as well as a list of resources within, super area RSNs only hold aggregated information about their sub areas. This aggregation is obtained in a next step: for each area their internal structure is abstracted away, as is shown in fig. 3 for the physical areas a, b, c, by pre-calculating a set of shortest paths between all pairs of their *border nodes* set B. The result is a set of path segments $PS_B = \{s(b_s, b_t) | b_s, b_t \in B, b_s \neq b_t\}$, which represents a fully connected graph of all border nodes. We refer to these pre-calculated paths as path segments as they are concatenated to form paths between users and resources later in the discovery process. A path segment s further is only represented by its two border nodes b_s and b_t and its propagation delay d when being communicated. Each RSN communicates the set PS_B to the RSN of its direct super area. Each super area therefore holds the PS_B of each of its sub areas. By only communicating the abstraction, which size depends on the number of border nodes, we achieve scalability throughout the entire hierarchy. The abstraction also limits the amount of details shared between MNOs.

B. The Discovery Process

The discovery is performed by sending discovery request messages (DISCREQ) independently from each group member through the hierarchical overlay. Each DISCREQ carries a group identifier (*groupID*), the number of members (*groupSize*) and the maximum allowed end-to-end propagation delay. It further includes a list of *candidate paths* P_c which is empty at start and a *direction flag* set to *up*. We assume the group definition to be done by the application and distributed to all group members, allowing them to construct their DISCREQs. For brevity, we only describe (and show in fig. 4) one group member's request from here onward.

First, the member sends its DISCREQ to the RSN of the physical area it is located in (e.g., area a). RSN_a now calculates shortest paths from the user to the area's border nodes (step ❶ in fig. 4). A path $p(S, b_t, d)$ is described by the path segments S it consists of, the border node b_t it currently terminates at and its total delay d. The resulting first set P_c is included in the DISCREQ which is forwarded one level upwards in the hierarchy to the RSN of a's super area $S_{1,1}$.

Based on the direction flag (either *up* or *down*) a request is handled differently by a receiving RSN. For the up direction, the RSN of $S_{1,1}$ performs the following steps:

- Each path p_i from P_c is extended to each border node b_i in $S_{1,1}$'s border node set B by concatenation with path segments s_i in PS_B of $S_{1,1}$ where s_i starts at p_i's b_t and ends at b_i (❷). As shortest path segments are concatenated the resulting path always is a shortest path itself within the scope of that super area.
- A new P'_c is built by adding the path with the lowest propagation delay d to P'_c for each border node b_i as described above. P'_c therefore only contains the shortest path to each of the $S_{1,1}$'s border nodes.
- P_c in the DISCREQ is replaced by P'_c and the request is sent up to the super area (e.g., $S_{2,1}$) where this process repeats (❸).

In addition to sending a DISCREQ up, $S_{1,1}$ replicates the original request and sends copies down to all sibling areas, i.e., all sub areas of $S_{1,1}$ except the area the request came from. Computing P'_c for these requests is identical except paths are extended to the respective sibling area's border instead of the border of $S_{1,1}$ (❹). The request is sent down (setting the direction flag) to the sibling area's RSN.

At each stage in this multi-stage discovery process, paths in P_c that exceed the maximum delay can be discarded immediately. Additionally, as we do not care for an optimal, complete solution the process presents ample opportunities to limit the areas requests are sent to, e.g., based on heuristics.

C. Result Collection

Physical area RSNs temporarily store the DiscReqs until all group members' requests arrived. Only then the RSN computes a set of resources within that area, that 1. fulfill the application's resource requirements and 2. feature feasible end-to-end paths between all group member pairs.

For that, the paths from all P_c sets are extended to all resources within that area (❺). Here, the physical area uses its full topology, analogue to how RSN_a calculated shortest paths from the user to its border nodes. The result are complete paths from the each member to all matching resources. It is sent upwards in the hierarchy, aggregated with sibling area's results and again forwarded to the next super area's RSN. At the highest area visited by the DiscReq a last aggregation produces the final result which contains set of feasible resources and their respective paths to/from all members.

III. Preliminary Results

The performance of Sherpa is evaluated through simulation experiments. Early results demonstrate the concept's feasibility in a test topology consisting of 1200 nodes and 1726 edges forming a 4 level deep hierarchy. Two resources are randomly placed in each of its 62 physical areas. This topology models three independent mobile networks stretching across a 1000 km square with corresponding propagation delay added to all links. To model ultra-low latency links between the networks, we connect them (assume peering) where nodes are geographically close to each other with some probability. The very brief results we present here show a scenario involving a group of 5 members placed randomly across the topology. $n = 30$ runs of the simulation were performed. The Sherpa implementation used, already includes heuristics to steer and limit discovery, i.e., it does not traverse the complete hierarchy.

We first compare Sherpas ability to find matching resources against an optimal solution obtained by calculating all-pairs shortest paths between each user and all resources in the topology. Figure 5 shows that Sherpa is able to find good sized sets of resources in all runs we performed, even though Sherpa visits only 41.5 % of all physical areas on average.

To show Sherpa's request messages stay within reasonable bounds we analyze the requests processed and approximate their respective message sizes. In fig. 6 we show message sizes on each level in the hierarchy and what total volume a single discovery process induces. DiscReq sizes stay below 1200 B on all hierarchy levels. The discovery on average results in 11.8 kB traffic for a single user, distributed over 40.9 requests on average.

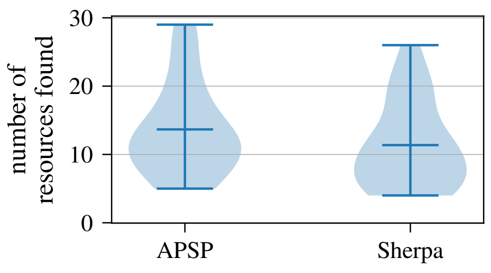

Fig. 5. Sherpa discovers only slightly smaller sets of resources compared to all-pairs shortest paths (APSP).

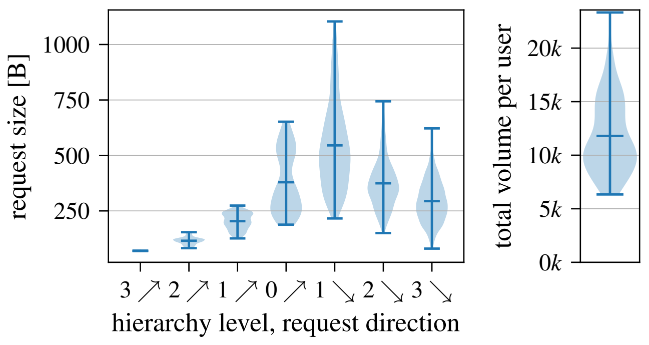

Fig. 6. Approximate request message sizes grouped by level in the hierarchy and its direction. Right: total volume for a single user's discovery process.

IV. Challenges and Future Work

We outlined Sherpa and presented promising first results. An open issue is how we can limit the search while guaranteeing a minimum result quality.

In the future we plan to extend Sherpa to allow for multi-criteria path selection to be able to account for additional path segment properties like available QoS, packet loss rates (for wireless links) or others.

V. Acknowledgement

This work is funded by Germany Federal Ministry of Education and Research (BMBF) under the project 6G-ANNA (grant number 16KISK089).

References

[1] P. Jain, "Imt-2030 use case workshop: A catalyst for 6g planning," *3GPP Highlights*, vol. 8, p. 25, Jun. 2024.
[2] C. Elvezio, F. Ling, J.-S. Liu, and S. Feiner, "Collaborative virtual reality for low-latency interaction," in *Adjunct Proceedings of the 31st Annual ACM Symposium on User Interface Software and Technology*, ser. UIST '18 Adjunct. New York, NY, USA: Association for Computing Machinery, Oct. 2018, p. 179–181.
[3] L. Brown, E. Marx, D. Bali, E. Amaro, D. Sur, E. Kissel, I. Monga, E. Katz-Bassett, A. Krishnamurthy, J. McCauley, T. Narechania, A. Panda, and S. Shenker, "An architecture for edge networking services," in *Proceedings of the ACM SIGCOMM 2024 Conference*. Sydney NSW Australia: ACM, Aug. 2024, p. 645–660.