Data Connectivity and Smart Group Formation in Wi-Fi Direct Multi-group Networks

Claudio Casetti, *Member, IEEE,* Carla Fabiana Chiasserini, *Senior Member, IEEE,* Yufeng Duan, Paolo Giaccone, *Member, IEEE,* Andres Perez

***Abstract*—Users of Device-to-Device (D2D) communication need efficient content discovery mechanisms to steer their requests toward the node in their neighborhood that is most likely to satisfy them. The problem is further compounded by the lack of a central coordination entity as well as by the inherent mobility of devices, which leads to volatile topologies. In this paper, we first discuss group-based communication among non-rooted Android devices using Wi-Fi Direct, a protocol recently standardized by the Wi-Fi Alliance. We propose intra- and inter-group communication methodologies, which we validate through a simple testbed where content-centric routing is used. Next, we address the autonomous formation of groups with the goal of achieving efficient device resource utilization as well as full connectivity. Finally, we evaluate the performance of our group formation procedure both in simulation and in a real testbed involving Android devices in different topologies.**

***Index Terms*—Device-to-device data transfer, network topology formation, experimental implementation and evaluation, Wi-Fi Direct.**

I. Introduction

Infrastructure-based communication is, by and large, the paradigm of choice for today's smartphones, tablets and laptops. Right out of the box, these devices invariably look for a cellular network or a Wi-Fi hotspot to associate with before they can operate. This paradigm is convenient for operators because it simplifies traffic monitoring, hence billing and service provisioning. Users too can obviously benefit from readily-available Internet connections provided by their association to cell towers and access points. However, infrastructure-based networking negates the advantage of direct communication between nearby devices should the need arise to exchange data or interact for social or gaming purposes. This is indeed an area where Device-to-Device (D2D) connectivity can play a major role, but it is not the only one. The trump cards of D2D communication, namely enhanced spectral efficiency and traffic offloading, make it an ideal choice for machine-to-machine communication, Internet of Things architectures and infrastructure replacement (in case of failure).

Facilitating the establishment of Device-to-Device (D2D) connectivity, whether in an unrestrained or in a network-controlled fashion, has been a recent goal of standardization and research efforts. The commercial appeal and widespread availability of D2D technologies such as Bluetooth Low Energy [1] and Wi-Fi Direct [2] is a testament to such a trend. Additionally, D2D (and Wi-Fi Direct in particular) is viewed by operators as a key enabler of LTE offloading strategies. D2D has also been standardized by 3GPP since LTE Release 12, initially for providing proximity based services for *public safety* applications. Subsequently, LTE/LTE-A standardization has introduced the concept of network-assisted D2D communication with an eye to the interoperability with other D2D technologies. The cellular interface would thus jump-start the D2D link between suitable devices by handling the discovery and authentication phases, thus serving as a broker party [3]–[5].

The biggest hurdle on the path to widespread adoption of D2D communication is the lack of impromptu service coordination. Even if the content needed by a device is cached by a nearby node, reachable through a multi-hop D2D path, its availability requires a robust content discovery and retrieval mechanism. Such mechanism should be aware of, and, if possible, should leverage the peculiarities of the D2D environment: high node churn, volatile topologies and resource-constrained devices.

In this paper, we focus on the potentiality of Wi-Fi Direct as D2D communication technology in medium and large-scale scenarios, using *non-rooted* Android devices. Our contribution is manyfold.

- We investigate the Wi-Fi Direct Android implementation and the roles that devices can play in a D2D multi-group topology. We then enhance it by designing and implementing a mechanism for multi-group data communication. Specifically, we design a multi-group, interconnected logical topology that overcomes the limitations of the physical one by exploiting application-layer tunneling. Such logical topology allows us to enable bidirectional, inter-group data transfers, which would otherwise be impossible in today's Wi-Fi Direct-based networks.
- We test our approach and show its validity, by using a testbed where we implemented a content-centric routing for data transfers. Our results also point out that building a smart network topology is of paramount importance to obtain good performance in inter-group data transfer.
- We therefore propose a fully distributed smart group formation mechanism in which devices can spontaneously form a physical multi-group communication network. This mechanism also enables devices to select their role in order to create an efficient logical topology for inter-group communications. Notably, we devise distributed algorithms that allow network formation accounting for group sustainability, device energy consumption, average throughput, and network coverage.

All authors are with the Department of Electronics and Telecommunications of Politecnico di Torino, Italy.

- We implement this smart group formation mechanism on non-rooted Android devices exploiting the procedures defined in Wi-Fi Direct and the Android Wi-Fi Direct framework. We experimentally verify the correct behavior of our implementation and assess its experimental performance in a real testbed.

To our knowledge, our work is the first that tackles bidirectional, inter-group communication in Wi-Fi Direct networks, and proposes and implements a solution to support this data transfer paradigm. Furthermore, we deploy a small-scale testbed using off-the-shelf, non-rooted Android devices and test both the feasibility of our multi-group topologies, as well as the efficiency of content-centric routing along with the registration/advertisement protocol. We also test the performance of our smart group formation mechanism in terms of topology creation time and coverage leveraging on this testbed.

The rest of the paper is organized as follows. Sec. II provides an overview of Wi-Fi Direct. Sec. III highlights some of the limitations that topology formation suffers from in Wi-Fi Direct devices and details the proposed multi-group communication mechanism. Thanks to this mechanism, we implement a content-centric multi-group routing scheme for data transfer, whose experimental performance is assessed in Sec. IV. We then introduce the design principles of our smart group formation mechanism and the related algorithms in Sec. V. In Sec. VI we describe the implementation of the smart group formation mechanism on non-rooted Android devices. We test this mechanism through both simulation and our testbed, and show the results in Sec. VII. Related work is discussed in Sec. VIII, while Sec. IX draws some conclusions and points out directions for future research.

A preliminary version of our work was presented in [6].

II. The Wi-Fi Direct Technology

Wi-Fi Direct is a recent protocol standardized by the Wi-Fi Alliance [2], with the aim to enable D2D communications between nodes, referred to as *peers*. Communication among peers in Wi-Fi Direct occurs within a single *group*. One peer in the group acts as Group Owner (GO) and the other devices, called *clients*, associate to it.

In order to establish wireless connections between devices, Wi-Fi Direct specifies the so-called P2P Discovery, which consists of three procedures: Device Discovery, Group Formation and Service Discovery. In Device Discovery, devices can exchange their own information including their device name, which is usually a human readable string that helps users to identify each other. When a device intends to connect to another, it performs the Group Formation procedure. According to this procedure, the two devices negotiate their roles, i.e., GO or client, by exchanging an integer value ($[0, 15]$), called *GO Intent*: the higher the value, the higher their willingness to become a GO. The Service Discovery procedure instead enables devices to advertise and detect higher-layer services that are available in their vicinity, without establishing any connection.

Once the GO is elected, the role of each peer remains unchanged during the whole group session. Only when the GO

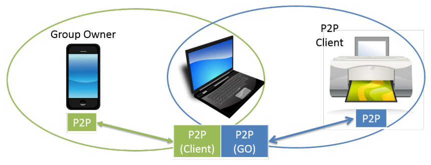


Fig. 1: Communication between two Wi-Fi Direct groups.

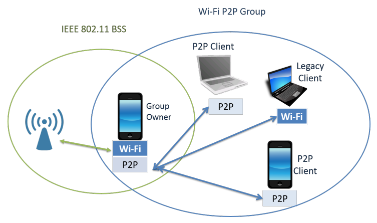


Fig. 2: Communication between a Wi-Fi Direct group and a Wi-Fi BSS.

leaves the group, the peers become disconnected and a new group should be created. The group works as an infrastructure Wi-Fi BSS, operating on a single channel, through which the peers communicate. The GO periodically transmits a beacon to advertise the group so as to enable disconnected devices to discover and, possibly, join the group. The new device exploits the standard Wi-Fi authentication and association procedure to join the group and becomes a client. Each client is either a *P2P client* or a *legacy client*. A P2P client supports the Wi-Fi Direct protocol, whereas a legacy client is a conventional Wi-Fi node that does not support Wi-Fi Direct and "sees" the GO as a traditional Wi-Fi AP. P2P clients and legacy clients coexist seamlessly in the same group. It is important to note that Wi-Fi Direct has been designed to support D2D communications within a group, however its protocol does not prevent the communication between different groups. Indeed, a peer can act as a bridge between two groups, as shown in Fig. 1 or between the group and other networks, as shown in Fig. 2. Note that the bridge peer must support two different MAC entities at layer 2, with two different MAC addresses.

III. Multi-group Communication

We investigate how to provide bidirectional multi-group communication in networks composed of Android devices. Android OS offers a limited, controlled set of networking capabilities for security reasons. It is possible to "root" a device in order to access advanced capabilities, but we do not take this possibility into account since the rooting process requires skills that are beyond the average user, and it renders the warranty null and void. Thus, we only act upon application-layer functionalities, i.e., no changes can be performed at the transport or network layer (like changing IP addresses for P2P interfaces, configuring routing tables, etc).

A multi-group topology could be implemented by letting a device have two virtual P2P network interfaces: in this way, it could act as a bridge using a different MAC entity in each group. In non-rooted Android devices, however, the

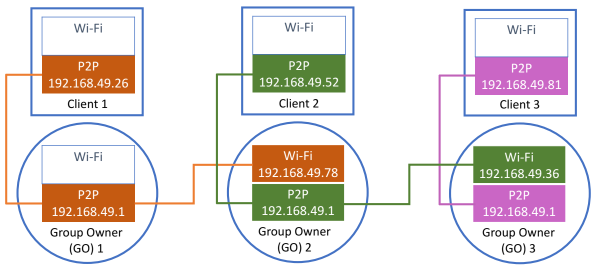


Fig. 3: Example of IP addresses (/24) for multi-group configuration with 3 groups: Group 1, Group 2 and Group 3.

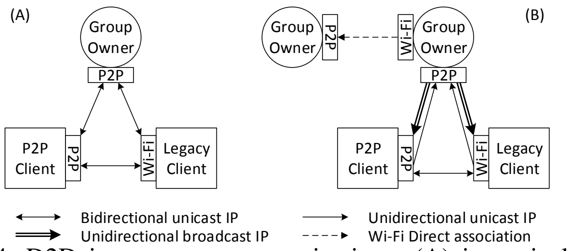


Fig. 4: D2D intra-group communications: (A) in an isolated group, (B) in a group whose GO is a legacy client in another group.

programmer cannot create a custom virtual network interface. Our experiments revealed that none of the following scenarios are feasible in Android, much though they are not expressly forbidden by the standard:

1) a device plays the role of P2P client in one group and GO in another,
2) a device behaves as the GO of two or more groups,
3) a device behaves as client in two or more groups.

Thus, in order to create a multi-group physical topology (i.e., bridge nodes), we let a GO be a legacy client in another group. Specifically, we proceed as depicted in Fig. 3, where three inter-connected groups are formed with six devices. GOs are represented by circles and clients by squares. In each peer, we enable two network interfaces, one of which is the conventional Wi-Fi interface and the other (P2P) is used for Wi-Fi Direct connection. The interfaces forming the same group are highlighted using the same color, while connections are represented by lines. It is important to remark that each group represents a different Wi-Fi Basic Service Set (BSS). Furthermore, note that GO2 and GO3 also act as legacy clients of GO1 and GO2, respectively. GO1 is not acting as a legacy client since it is not associated to any other group. As discussed later in Sec. III-A, the fact that one GO is a legacy client of another GO affects its forwarding capabilities.

In Android devices, once a Wi-Fi Direct connection is established, the GO automatically runs the DHCP to assign IP addresses to itself (192.168.49.1/24) as well as to the P2P clients or legacy clients in its own group (192.168.49.x/24 where x is a random number $\in [2, 254]$ to minimize the chance of address conflicts). Therefore, the P2P interfaces of all GOs have the same IP address, as shown in Fig. 3.

Given the above assignment of IP addresses, we design a *logical topology* that implements multi-group communications. Our methodology overcomes the limitations of the physical topology and of its addressing plan, which prevent data transfers on some D2D links.

A. Intra-group communications

Intra-group communications are the basis to enable bidirectional inter-group communications. Two cases have to be distinguished. In the first case, depicted in Fig. 4(A), the GO is not connected to any other group as legacy client. Since Wi-Fi Direct has been designed to provide full connectivity between the nodes of an isolated group, all possible D2D communications are enabled. Thus, any pair of devices (GO, P2P clients and legacy clients) can exchange data at the IP layer. Note that, in the specific example in Fig. 3, Group 1 falls in this case (hence all D2D communications are allowed) since GO2 is a standard legacy client as far as GO1 is concerned. In the second case, illustrated in Fig. 4(B), the GO is also connected to another group as a legacy client. Referring again to the example network in Fig. 3, Group 2 and Group 3 fall in this case.

We observe that all D2D unicast data transfers among clients (P2P or legacy clients) are allowed, thus TCP connections and/or UDP flows between clients are supported. Instead, between a GO and its clients, or between two GOs, *only a subset of D2D data transfers are allowed.* The reason underlying this limitation is twofold.

- First, when the GO wants to send a unicast IP packet to any client of its group, the packet is invariably sent through the GO Wi-Fi interface, since the latter entry is listed with higher priority than the P2P interface in the routing table of the device[1]. In the client→GO direction, instead, the communication is allowed since client routing tables list only one interface and no conflict occurs. Since only unidirectional unicast communication between the client and the GO can take place, no TCP connection can be established between the GO and its clients, whereas UDP flows are allowed only from the clients to their own GO.
- Secondly, two neighbor GOs cannot communicate directly, because of the IP address conflict. Note that in this case one of the GOs acts as legacy client of the other GO, as in the example of Fig. 3 where GO2 is legacy client of GO1. When GO2 wishes to transmit an IP packet to GO1, the destination is set to 192.168.49.1 and the packet is thus sent to its local loop and not to the Wi-Fi interface. Also, when GO1 sends an IP packet to GO2 (192.168.49.134), GO2 discards it since its IP layer detects that the packet source address matches its own (192.168.49.1).

In summary, *neither bidirectional unicast data transfer between GO and its clients, nor direct data transfer between GOs, is allowed.*

In order to overcome the first issue, we note that broadcast IP packets sent by the GO are always[1] sent through its P2P

[1]We consistently observed this behavior for different devices, of different brand, running Android 4.3 and 4.4.

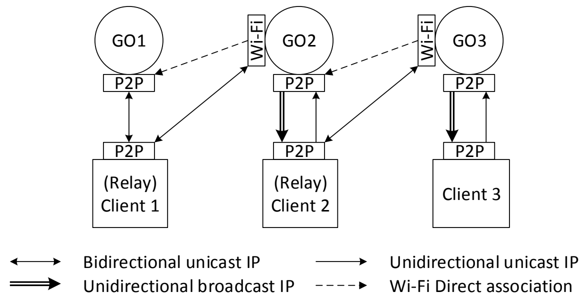


Fig. 5: Example of inter-group communication for 3 groups in a linear topology with 6 devices.

interface. This is an important observation as it allows the support of bidirectional data transfer between each client and its GO: broadcast IP packets can be used from the GO to the clients, while unicast IP packets can be adopted to transfer data from the clients to the GO. We remark, however, that broadcast packets generated by the GO will also reach the GOs associated to it as legacy clients, but then such packets will be discarded because of the conflict of source IP address, as discussed above. So, it is not possible for a GO to directly reach neighbor groups. Lines connecting the nodes in Fig. 4(B) summarize the possible intra-group data transfers at IP level. Below we address this issue and design a solution that enables inter-group communication.

B. *Inter-group communications*

We recall that D2D communications are allowed between any two *clients* within the same group (i.e., not involving the GO at IP layer). Thus, also the communication between a P2P client and a legacy client that is also the GO of a different group is allowed in both directions. This observation is crucial, since it provides support for our novel design that exploits *a client within the group as relay to reach a neighbor group*. Specifically, we provide bidirectional, inter-group communication between neighbor groups by adopting the communication scheme shown in Fig. 5. To send data from the central group (Group 2) to its right side group (Group 3), we leverage a P2P client (client 2) to relay the traffic toward GO3. Instead, to send data from Group 2 to its left side group (Group 1), GO2 itself is responsible to relay traffic toward a client in the left side group (client 1). In other words, we build a logical topology based on transport-level tunnels enabled by IP and MAC-layer connectivity, as follows.

- Unidirectional UDP tunnels between a GO and its P2P clients (e.g., GO1 and client 1). They are based on broadcast IP packets from the GO to clients and on unicast IP packets from clients to the GO. When reliable communication is required towards a single client, the GO can adopt a classical stop-and-wait protocol.
- Bidirectional UDP or TCP tunnels between P2P clients and legacy clients within the same group (e.g., between client 1 and GO2, or client 2 and GO3).

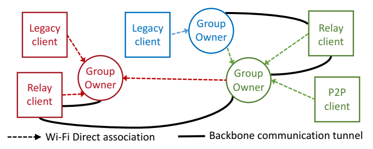


Fig. 6: Communication backbone over an arbitrary network topology.

Full connectivity among nodes in a multi-group network can thus be provided by leveraging a proper sequence of transport-layer tunnels established in the logical topology, and switching packets at the application layer (thus, *without rooting the devices*).

The role of the Relay Client. To define a routing process that properly leverages the above transport-layer tunnels, we select one client within each group, to act as a relay node with respect to neighbor groups. We name such node *Relay Client*. In the example in Fig. 3, client 1 (client 2) is the Relay Client connecting Group 1 (Group 2) to Group 2 (Group 3).

We devise a Relay Client selection scheme and implement it, as described in Sec. VI-E.

C. *The communication backbone*

To disseminate data across a large set of devices, we then propose a logical tree topology, connecting all groups by extending the approach shown in Fig. 5 to an arbitrary number of groups. By doing so, we build a communication backbone, as depicted in Fig. 6. The figure highlights the GOs (circles) and the Relay Clients that compose the backbone and provide connectivity to all other clients (P2P and Wi-Fi clients that do not act as GOs, i.e., that are not involved in the traffic relay process). In principle, our approach might scale indefinitely, even if we were able to validate it experimentally only for few groups, as shown in Sec. IV.

It is important to remark that a path over the backbone involving transfers from GO to Relay Client within the same group requires a broadcast IP transmission for each of such transfers. Instead, transfers from Relay Client to GO do not require any broadcast IP transmission.

IV. Experimental Evaluation Through Content-centric Routing

In order to validate the proposed multi-group communication mechanism, we implemented a content-centric routing and setup a testbed where nodes advertise the content they own and request the content they need.

The content-centric routing protocol is described in detail in our preliminary work [6]. In a nutshell, it allows devices to advertise the content they own, inside as well as outside their group, and to obtain the routing information to reach content items stored by others. In the following, instead, we focus on our experimental setup (Sec. IV-A) and on the results showing the validity of our approach to enable inter-group communication (Sec. IV-B).

A. Testbed

We developed a testbed including several Android devices of different type, namely, Google Nexus 7 and ASUS Transformer Pad TF300 tablets and 2 different smartphones (LG P700, Sony Xperia Miro ST23i). The Nexus tablets were equipped with Android 4.4.2, but our application was also tested with Android 4.3 on the same devices. LG smartphones used Android 4.0, which is the oldest version supporting Wi-Fi Direct. In our tests, LG smartphones acted as P2P clients and never as GOs, since the transport-layer tunnels from/to the GO discussed in Sec. III-A are fully enabled only for Android 4.3 and later versions. The ASUS tablets and the Sony Xperia were equipped with Android 4.2.1 and Android 4.0.4, respectively. Neither of them support Wi-Fi Direct; we used such devices only as legacy clients and not as group owners. This variety in the choice of devices allowed us to validate our multi-group communication mechanism in presence of heterogeneous devices and different conditions. *No device was rooted*, to be sure that we could validate the approach for off-the-shelf devices.

We developed an Android application to implement our solution for bidirectional, multi-group communication and content-centric routing, as well as to validate the whole approach and assess its performance. For brevity and ease of presentation, in the following we show the results that we obtained using an experimental setting with two Wi-Fi Direct groups. Group 1 includes 4 devices (GO1, client 1A, client 1B and GO2, the latter acting as legacy client in Group 1), while Group 2 comprises 2 devices (GO2 and client 2A). Client 1B and client 2A operate as Relay Clients in their own groups. All the tablets were located in proximity of each other, to reduce the effects of propagation delays and signal attenuation due to distance.

B. Experimental Results

We evaluate the achievable performance for the content data transfer from one device to another, based on the content-centric scheme we implemented [6]. Each content is divided into chunks of fixed size equal to 1400 bytes, to avoid IP fragmentation. To vary the offered traffic load, the content provider periodically sends a new chunk, encapsulated into a Content Data message, with the chunk rate being a varying application parameter.

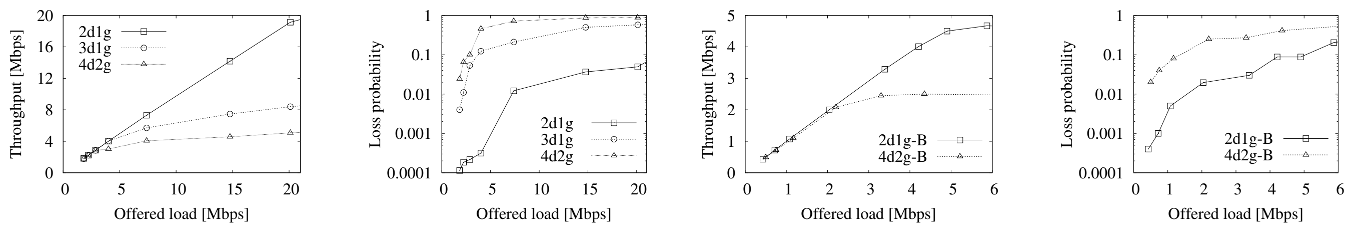
We validated the data delivery mechanism by picking different pairs of devices among the possible ones, and letting them act as source-destination nodes. We therefore verified the full bidirectional connectivity over the whole multi-group network, and recorded the application-layer throughput and the packet losses experienced at the IP layer, as functions of the application-layer traffic offered load. For each configuration, we run 100 different experiments, to obtain throughput results with a 1% relative width of the 95% confidence interval.

In the following, we mainly focus on the scenarios detailed below, run on our testbed.

1) "2 devices - 1 group" (2d1g), in which the source is client 1A and the destination is GO1. The communication between a client and its GO involves just one hop at IP and MAC layer, since each message is sent through a single unicast IP packet, carried by a single MAC frame.
2) "3 devices - 1 group" (3d1g), with client 1A as source and client 1B as destination. The communication between two clients in the same group involves one hop at the IP layer, but two hops at the MAC layer (client 1A $\rightarrow$ GO1 $\rightarrow$ client 1B).
3) "4 devices - 2 groups" (4d2g), in which the source is client 2A and the destination is client 1B. The communication between the two clients in 2 groups requires two hops at IP layer (client 2A $\rightarrow$ GO2 $\rightarrow$ client1B) and three hops at MAC layer (client 2A $\rightarrow$ GO2 $\rightarrow$ GO1 $\rightarrow$ client 1B).
4) "2 devices - 1 group - broadcast" (2d1g-B), in which the source is GO2 and the destination is client 2A. The communication within the same group now occurs in the opposite direction with respect to the 2d1g case, but notably the single-hop communication is based on a broadcast transmission, since GO2 is also legacy client of GO1.
5) "4 devices - 2 groups - broadcast" (4d2g-B), with client 1B as source and client 2A as destination. The communication between the two clients in two different groups involves 2 hops at IP layer (client 1B $\rightarrow$ GO2 $\rightarrow$ client 2A) and 3 hops at MAC layer (client 1B $\rightarrow$ GO1 $\rightarrow$ GO2 $\rightarrow$ client 2A), in which the last hop is based on a broadcast transmission.

Note that we do not show the case of content transfer from GO1 to client 1A since it is equivalent to the 2d1g case; indeed, GO1 is not a legacy client of any other group, thus it can send unicast IP packets directly to client 1A. Also, for fair comparison, we start by evaluating the first three cases, which imply only unicast transmissions; then, we will move on to the last two cases, involving broadcast transmissions.

Fig. 7(a) shows the application-layer throughput vs. the offered load. As expected, the throughput increases with the load, and reaches a maximum value of about 19 Mbit/s (2d1g scenario), 8.4 Mbit/s (3d1g scenario) and 5.0 Mbit/s (4d2g). These results are coherent with the fact that the throughput decreases proportionally to the number of hops, due to the channel contention among the transmitters operating on different hops. Note that current available Wi-Fi Direct virtual interfaces have to work on a single frequency channel and, thus, the whole multi-group network is part of the same collision domain. In general, the number of hops traversed by a packet depends only on the distance, in terms of number of groups, over the backbone between source and destination (typically, two hops at the MAC layer per each traversed group), whereas it is independent of the total number of devices composing the network. While the single collision domain increasingly affects the performance as the number of active transmitters grows, having the hop number independent of the group size improves scalability. Additionally, the impact of the single collision domain lessens as the network gets larger: when transmitters are far away from each other, some degree of spatial diversity is possible and interference among parallel transmissions greatly reduces. It follows that the network throughput decreases less than proportionally to the number



(a) Application-layer throughput (b) IP-layer packet loss (c) Application-layer throughput (d) IP-layer packet loss

Fig. 7: Performance vs. offered traffic load, for packet transfers involving only unicast transmissions (a), (b), and for packet transfers involving also broadcast transmissions (c), (d).

of hops.

Fig. 7(b) shows the overall packet loss probability. Note that packet losses are almost negligible in the 2d1g scenario. They become noticeable for the 3d1g case (0.4% for the lowest load) but still have little impact on the throughput. Under the 4d2g scenario, instead, packet losses are more significant (2.4% for the lowest load), since the transmissions over the three hops that every message has to undergo at the MAC layer interfere with each other.

Figs. 7(c) and 7(d) depict throughput and packet losses, respectively, for the last two scenarios, 2d1g-B and 4d2g-B, both implying one broadcast transmission by GO2. The maximum throughput is 4.6 Mbit/s for 2d1d-B and 2.5 Mbit/s for 4d2g-B. Such numbers are much smaller than in the first three scenarios, since, at MAC layer, 802.11 broadcast packets are transmitted at the minimum data rate (6 Mbit/s), whereas much higher rates are used for unicast transmissions (up to 54 Mbit/s). Note also that, even if three hops are involved in 4d2g-B, the first two hops occur through unicast transmissions (hence at much higher data rate than broadcast transmissions) and, thus, they mildly affect the throughput. Looking at Fig. 7(d), it can be seen that the loss probability is higher in the two scenarios with broadcast transmissions than in previous cases. This behavior is also expected: in case of failure, broadcast packets are never retransmitted at the MAC layer, thus the reliability of the communication from the GO to its Relay Client is severely reduced.

In summary, the performance of the communication backbone is strongly affected by the traffic flow direction. The two different relay schemes, adopted within a group to work around the constraints imposed by Wi-Fi Direct, show significantly different performance. The main bottleneck is represented by broadcast communications from the GOs to their Relay Clients.

V. Smart Group Formation

As confirmed by the above results, the Wi-Fi Direct network topology plays a crucial role in the performance achieved by inter-group data transfers. Thus, in this section we propose a fully-distributed group formation mechanism that (i) meets all the technology requirements, (ii) accounts for the devices physical resources and power consumption, and (iii) generates topologies that facilitate an effective inter-group communication.

Before introducing the mechanism, we highlight the main principles (P1-P4) that guide our design.

(P1) When creating the network topology, the number of groups should be minimized while ensuring full connectivity. Indeed, as shown in Sec. IV, the throughput between two peers degrades with the number of hops the packets traverse. Properly choosing the GOs allows the control of the virtual topology on which data is routed to reduce the number of traversed hops. Note that involving more peers in a single communication also creates unnecessary power consumption of the relay peers along the path.

(P2) The choice of the GO should be optimized not only by taking into account its position within the topology, but also selecting the node that best meets critical requirements, such as high residual energy or unused computing/storage resources. As highlighted in the previous sections, GOs play an important role in each group: forwarding packets at the MAC layer, keeping track of the content with associated peers, managing inner- and inter-group content exchange, etc. As a consequence, GO devices consume more power than clients, due to higher computation and communication requirements. Moreover, once a GO leaves, all the nodes in the Group become disconnected. Therefore, properly assigning the GOs can make groups last longer and provide better performance. Besides, devices that can provide other resources (e.g., localization, Bluetooth availability, Internet connection) could be also preferred candidates to act as GOs.

(P3) Another crucial role in the network is played by Relay Clients, since they handle all the inter-group traffic, hence, their selection must be optimized as well, in the same spirit of choosing GOs.

(P4) Finally, the multi-group network should involve as many devices as possible. As introduced in Sec. II, devices discover each other during the Device Discovery procedure. Once started, Device Discovery remains active until a device initiates a P2P connection and establishes a P2P Group with others. After a device finishes the Device Discovery, the procedure will never start again unless it is manually triggered by the user. As a consequence, some devices may not have the chance to find each other, leaving some devices disconnected. While setting up the network, clearly we aim at maximizing the number of connected devices in order to maximize the connectivity opportunities and the amount of shared content in the whole network.

We therefore devise a procedure that accounts for the above requirements and lets each Wi-Fi Direct device coordinate with its neighbors so as to effectively decide its own role in the network topology (i.e., GO, client or Relay Client), and, connect to other devices according to the selected role. For ease of presentation, we describe the procedure by referring to a set of devices that have not established any wireless link yet, and, following the Wi-Fi Direct specifications, they perform the Device Discovery procedure to acquire information on their neighborhood. We therefore start by introducing the information that each device needs to collect about its neighborhood (Sec. V-A) and then we detail the steps of the network formation procedure (Sec. V-B).

A. Neighborhood information

Unconnected network devices execute the Wi-Fi Direct Device Discovery procedure till they become part of a group. Through such procedure, devices can advertise their own presence and information about themselves to neighbor nodes, as well as receive other devices' advertisements. In the scheme we devise, the advertised information consists of the following main pieces:

- *Suitability to be a GO.* The GO Ability Index (GOAI) quantitatively expresses the overall level of available resources at the advertising device. It corresponds to the weighted sum of the level of available resources (e.g., battery level, availability of Internet connection, CPU maximum frequency, amount of RAM and of non-volatile storage, etc.). Note that the GOAI plays an important role also in the selection of the Relay Client.
- *List of neighbors* of the advertising device. The *neighbor* of a device is defined as a node whose presence can be detected through the Device Discovery procedure.
- *Current state in the topology formation.* As the topology formation progresses, this indicates the state of the procedure reached by the advertising device.

Note that, according to the above scheme, each device can acquire only a local view of the surrounding nodes, based on the list of its 1-hop neighbors with their GOAI, and on the list of its 2-hop neighbors. Indeed, enabling each device to learn the entire network topology would require a large exchange of information, which may lead to network congestion and increased latency to form the group.

B. Topology formation

We devise a distributed algorithm running at each device to decide its role (GO/client/Relay Client) whose aims are manyfold. First, a device with higher GOAI should have higher probability to become a GO. Second, the GOs should form a connected backbone to allow multi-group communication. Third, the number of groups should be minimized.

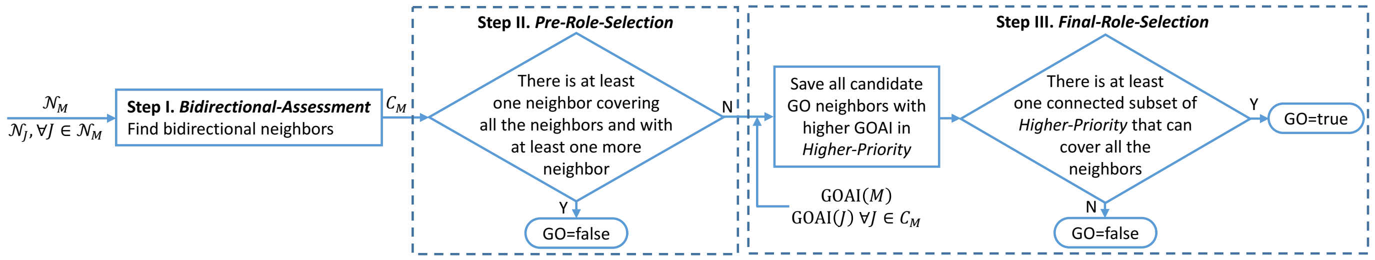
We consider the undirected connectivity graph among the nodes, according to which each vertex corresponds to a device and an edge exists between two vertices if the corresponding devices can communicate directly. The topology formation can be modeled as a problem of Minimum Connected Dominating Set (MCDS) on the connectivity graph. A *dominating set* (DS) is a set of nodes such that all the nodes outside this set can reach the nodes in the DS within one hop. Each node in the DS is called the *dominant node* and corresponds to a GO in our multi-group network. If all nodes in the DS can be connected to each other through only nodes in the DS (i.e., the DS induces a connected subgraph in the connectivity graph), then the DS is a Connected Dominant Set (CDS). The CDS will thus represent the communication backbone among GOs, which enables multi-group communications. Clearly, in order to minimize the number of groups, we need to minimize the number of nodes in the CDS.

It is well known that the MCDS problem is NP hard. Several distributed, approximated algorithms, e.g., [7]–[9], have been proposed in the literature. In [7] a node selects its role based on the choices of other nodes, which requires costly synchronization and coordination among the nodes, unlike our proposed approach. In [8] nodes make decisions dynamically, while in our case the nodes only decide their roles at the beginning. Indeed, recall that a Wi-Fi Direct group ends whenever the device acting as GO disconnects from the other nodes and, at that point, a new group has be be formed from scratch. Besides, the algorithm in [8] requires the geographical positions of the neighbors, which may not be available in our scenario.

Our approach is instead based on Dai and Wu's (DW) algorithm [9], which has been designed with a different goal in mind than ours, namely data routing rather than topology formation. In [9], each node is assigned a unique ID, which is a value representing the node's priority to become dominant. The DW algorithm runs in two synchronous steps. Initially, during the *marking step*, each node marks itself as dominant if it has two neighbors that are not directly connected. Clearly, this marking process generates a large number of dominant nodes. In the subsequent *self-pruning step*, the set of nodes in the DS is reduced. Each marked node applies a self-pruning rule, called *Rule-k*, to *unmark* itself, where k is a positive integer. According to this rule, a marked node, say node u, unmarks itself if all of its neighbors can be covered[2] by k nodes that (i) have a higher ID than its own, (ii) are marked, and (iii) induce a connected subgraph. The last condition ensures that such a set of k dominant nodes is eligible to build the GOs backbone but requires full knowledge of the connectivity among nodes that are arbitrarily far from u – an information that is hard to acquire in Wi-Fi Direct networks. At the end of the self-pruning step, the marked nodes elect themselves as dominant.

The DW algorithm is tailored to wireless ad-hoc networks where nodes are already connected and solves the routing problem, indeed only dominant nodes are allowed to route messages. As a result, it does not consider the full mesh topology where all the nodes are neighbors of each other. In this case, applying the DW algorithm, no node can mark itself

[2]A node set A is covered by node set B if and only if each node in A is the neighbor of at least one node in B.



$\mathcal{N}_M$
$\mathcal{N}_J, \forall J \in \mathcal{N}_M$
Step I. *Bidirectional-Assessment*
Find bidirectional neighbors
C_M
Step II. *Pre-Role-Selection*
There is at least one neighbor covering all the neighbors and with at least one more neighbor
Y
N
GO=false
Step III. *Final-Role-Selection*
Save all candidate GO neighbors with higher GOAI in *Higher-Priority*
GOAI(M)
GOAI(J) $\forall J \in C_M$
There is at least one connected subset of *Higher-Priority* that can cover all the neighbors
Y
N
GO=true
GO=false

Fig. 8: The three steps of the role selection algorithm run by node M.

Algorithm 1 BIDIRECTIONAL-ASSESSMENT at node M
Input: $\mathcal{N}_M$, $\mathcal{N}_J$ for all $J \in \mathcal{N}_M$
Output: $\mathbf{C}_M$ ▷ List of M's bidirectional neighbors

```
1: initialize $\mathbf{C}_M = \{\}$
2: for every neighbor device $J \in \mathcal{N}_M$ do
3:     if $M \in \mathcal{N}_J$ then                ▷ if $M$'s neighbor sees $M$
4:         add $J$ to $\mathbf{C}_M$      ▷ then, bidirectional connectivity
5:     end if
6: end for
```

in the marking step since no node can find two neighbors that are not connected. In our scenario, instead, we aim at selecting proper dominant devices and forming groups that lead to efficient data transfers across the network.

Our algorithm, named *Smart-Group-Formation* (SGF), is fully distributed and runs at each node independently, following a procedure based on three subsequent steps. Thus, some time coordination is necessary in order to ensure that all the nodes are simultaneously running the same step; an implementation that meets such requirement is presented in Sec. VI. In the following we explain each step focusing on a tagged node M, and we will refer to its GOAI as GOAI(M).

Step I: Bidirectional-Assessment. Due to the unreliability of the radio communication and the Device Discovery protocol, the visibility between any pair of nodes may not be bidirectional, and thus a node may not appear in the neighbor list of its neighbors. In this step, each nodes finds the neighbors for which bidirectional visibility is assured, namely, its *bidirectional neighbors*. Only such nodes are considered "valid" neighbors for the following steps. In particular, let $\mathcal{N}_M$ be the set of devices discovered by M, and let $\mathbf{C}_M \subseteq \mathcal{N}_M$ be the set of devices with which M has bidirectional visibility. Algorithm 1 reports the pseudocode for the BIDIRECTIONAL-ASSESSMENT of SGF, aimed at obtaining the set of bidirectional neighbors $\mathbf{C}_M$: for each neighbor, node M checks whether M itself appears in the neighbor list of each of its neighbors.

Step II: Pre-Role-Selection. This step is described in Algorithm 2: each node makes a preliminary decision on whether to become a GO or not, by checking if there are two unconnected neighbors, similarly to the marking step in DW algorithm. To solve the issue that no device would be marked in a full mesh topology under the DW algorithm, we modify

Algorithm 2 PRE-ROLE-SELECTION at node M
Input: $\mathbf{C}_M$, $\mathcal{N}_M$, $\mathcal{N}_J$ for all $J \in \mathbf{C}_M$
Output: GO(M) ▷ Boolean flag

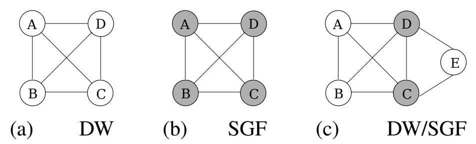
```
1: for every device $J$ in $\mathbf{C}_M$ do
2:     if $(\mathcal{N}_M \cup \{M\}) \subset (\mathcal{N}_J \cup \{J\})$ then
3:         GO($M$)=false                          ▷ $M$ is not GO
4:         return
5:     end if
6: end for
7: GO($M$)=true                          ▷ $M$ is a candidate GO
```

Algorithm 3 FINAL-ROLE-SELECTION at a candidate GO node M
Input: $\mathcal{N}_M$, GOAI(M), GOAI(J) for all $J \in \mathbf{C}_M$
Output: GO(M) ▷ Boolean flag

```
1: // Find all candidate neighbors with higher GOAI than $M$
2: Initialize Higher-Priority={}  ▷ Neighbor candidate GOs
   with higher GOAI
3: for every device $J$ in $\mathbf{C}_M$ do
4:     if (GOAI($J$) > GOAI($M$)) and (GO($J$)=true) then
5:         add $J$ to Higher-Priority
6:     end if
7: end for
8: // Decide the role of $M$
9: for each fully connected subset $\Omega \subseteq$ Higher-Priority do
10:     if $\mathcal{N}_M \subseteq \mathcal{N}_\Omega$ then
11:         GO($M$)=false                          ▷ $M$ is not GO
12:         return
13:     end if
14: end for
15: GO($M$)=true                          ▷ $M$ becomes GO
```

the DW marking step as shown in lines 2-4: a device M does not become a GO if there exists at least one neighbor that can cover all the neighbors of M and has at least one more neighbor than M; otherwise, the node becomes a candidate GO.

Fig. 9 illustrates the reasoning behind the different marking approach adopted by SGF compared to DW. Fig. 9(a) shows a full mesh topology in which no node can become a GO according to the DW algorithm, since no one in the graph has unconnected neighbors. On the contrary by applying SGF, as



(a) DW marking (b) SGF marking (c) DW/SGF marking

Fig. 9: Effect of different marking rules (DW and SGF) for two different topologies (a-b and c). Marked nodes are highlighted in gray and represent candidate GOs.

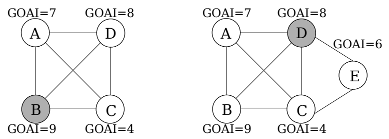


Fig. 10: Results of running Final-Role-Selection on the candidate GOs obtained in Pre-Role-Selection, for the cases shown in Fig. 9(b) and Fig. 9(c) respectively. Gray nodes denote the final GOs.

shown in Fig. 9(b), no node has any neighbor whose coverage is larger than its own, hence, all nodes become candidate GOs. Fig. 9(c) shows a different topology in which, according to the DW marking step, nodes C and D would become candidate GOs since they both have a neighbor E that is not connected to their other neighbors. According to our approach, we get the same result. Specifically, for nodes A and B, their neighbors, C and D cover all the neighbors of node A and node B, and have a distinct neighbor, node E. Thus nodes A and B cannot become GOs and the role selection step ends at this stage for them. Node E does not become GO either. In the meanwhile, none of nodes A, B or E can fully cover the neighbors of nodes C and D. As a consequence, only C and D are marked as candidate GOs.

Step III: Final-Role-Selection. It is executed only by those nodes that marked themselves as candidate GOs at the end of the previous step (i.e. GO(M)=true). As discussed in Sec. V-A, thanks to the advertised GOAI, each candidate node can build a set, named Higher-Priority, of bidirectional neighbors that are candidate GOs and have higher GOAI than itself (see Algorithm 3, ln. 3-7). Then, in order to select its role, M applies a restricted version of Rule-k: node M unmarks itself if there exists a subset of 1-hop neighbors that (i) appear in the Higher-Priority set and (ii) are 1-hop away from each other. Thus, in lines 9-14 of Algorithm 3, if at least one subset in the Higher-Priority set meets requirement (ii), then M unmarks itself and does not become GO. Note that in the pseudocode, given a generic subset Ω of Higher-Priority, $\mathcal{N}_\Omega$ denotes the union of the set of neighbors discovered by each node in Ω: $\mathcal{N}_\Omega = \cup_{X\in\Omega}\mathcal{N}_X$.

Fig. 10 shows the results of running Final-Role-Selection after Pre-Role-Selection in the topologies shown in Fig. 9. In the full mesh topology of Fig. 9(b), all the nodes become candidate GO after finishing Algorithm 2. When running Algorithm 3, nodes A, C and D only add node B to their Higher-Priority set, since B is the neighbor candidate GO with the highest GOAI. When checking all the possible subsets of Higher-Priority, which is actually $\{B\}$, the three nodes find that this subset can cover their neighbors. As a result, all these three nodes decline to become GO. Instead, node B finds that none has a higher GOAI than itself, and thus its Higher-Priority set is empty; as a result, B becomes GO. In the topology of Fig. 9(c), node C and D are candidate GOs at the end of Algorithm 2 and they are the only running Algorithm 3. Node D has an empty Higher-Priority set, as the only neighbor candidate GO C has a lower GOAI; thus D becomes GO. On the contrary, node C adds node D to its Higher-Priority set; since D can cover all of its neighbors, node C declines to become a GO.

VI. Implementation of Smart Group Formation

We implemented the SGF mechanism through five sequential phases: (i) *Neighborhood Information Collection (NIC)*, when devices gather information about other devices, (ii) *Neighborhood Advertisement (NA)*, when each device advertises its 1-hop neighbors, (iii) *Role Selection (RS)*, when devices decide their roles according to the distributed approach described in Sec. V-B, (iv) *Connection (CO)*, when devices set up the fully connected, multi-group network, and, after a group becomes consolidated, (v) *Relay Client Selection*, when the GO selects one client as Relay Client. These implementation phases are summarized in Fig. 11. Notably, during the first 4 phases, the devices are not yet connected at layer 2 and interact only exploiting the advertised device name. Only after phase (iv), the devices are connected at layer 2.

Recall that the Device Discovery procedure in Wi-Fi Direct allows a device to discover other nodes by acquiring their MAC address and device name, which is a human-readable string of ASCII characters. Also, a device can be discovered by others while performing the discovery procedure. In our SGF implementation, we use the device name field of the messages transmitted during the Device Discovery procedure to encode the information that devices need to advertise at various stages of the topology formation process.

A. Neighborhood Information Collection (NIC)

Upon starting, each device computes its own GOAI and encodes it into human-readable ASCII characters in the interval $[32, 127]$, which appears explicitly in the device name advertised by the node, as string `SGF_ID-GOAI`. In more detail, the `SGF` string identifies all the nodes running our SGF scheme. The following 4 ASCII characters represent the device ID, corresponding to the last 2 bytes of the MAC address of the interface, in order to minimize ID collisions. During the initial Device Discovery, each device discovers its neighbor nodes in terms of MAC address and device name. Notably, we avoided to use the GO Intent field – an integer in the interval $[0, 15]$ included in the GO Negotiation Request [10] to set up a connection – for two reasons: (i) without rooting the device, it is not possible to modify the GO Intent used to compare with the requester at the receiver, and (ii) there is a limited number of bits (just 4) to encode the GOAI.

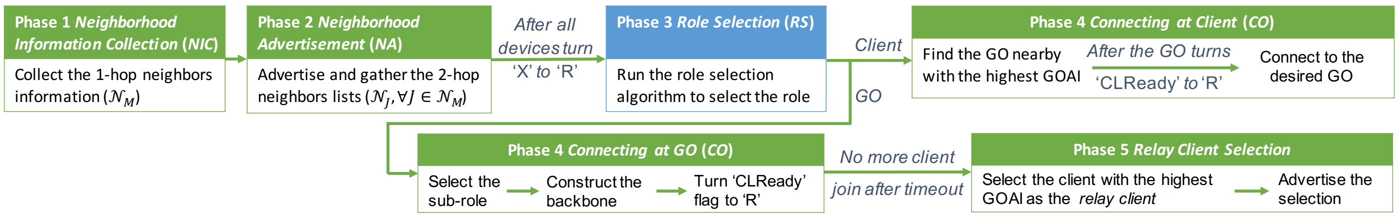

Phase 1 Neighborhood Information Collection (NIC)
Collect the 1-hop neighbors information ($\mathcal{N}_M$)
Phase 2 Neighborhood Advertisement (NA)
Advertise and gather the 2-hop neighbors lists ($\mathcal{N}_J, \forall J \in \mathcal{N}_M$)
After all devices turn 'X' to 'R'
Phase 3 Role Selection (RS)
Run the role selection algorithm to select the role
Client
GO
Phase 4 Connecting at Client (CO)
Find the GO nearby with the highest GOAI
After the GO turns 'CLReady' to 'R'
Connect to the desired GO
Phase 4 Connecting at GO (CO)
Select the sub-role
Construct the backbone
Turn 'CLReady' flag to 'R'
No more client join after timeout
Phase 5 Relay Client Selection
Select the client with the highest GOAI as the relay client
Advertise the selection

Fig. 11: The implementation phases of SGF for a device M: from the initial device discovery to the formation of the multi-group communication network.

TABLE I: Minimum number of discovered devices required in each time interval to end the P2P Discovery phase, with $N_1 \geq N_2 \geq N_3 \geq N_4$

Time Interval $[s]$	Minimum number of devices	Example value
$[0, 2]$	N_1	4
$[0, 5]$	N_2	3
$[0, 10]$	N_3	2
$[0, 15]$	N_4	1

Due to the asynchronous nature of the Device Discovery process, the vision of the neighborhood of each node becomes coherent with that of all other nodes only after some transient time. Thus, it is important to set properly the maximum time allowed for this initial phase in order to find the best compromise between the consistency of the devices' view and the duration of the network topology formation. According to [11], during P2P Discovery, most of the devices tend to be found either within the initial 2 seconds or after around 15 seconds; these results provide some guidelines to tune the discovery timing. In particular, we propose the threshold-based timing shown in Table I, according to which it is necessary to see at least a minimum number of neighbor devices in a given interval to conclude the NIC phase. For growing time intervals, we decrease the minimum required number of discovered neighbor devices. If a device has found no neighbors within 15 seconds, it restarts P2P Discovery and resets the timer. The list of discovered neighbors is stored in the local device's *neighbor list*.

B. Neighborhood Advertisement (NA)

At the end of the NIC phase, each device starts advertising the list of its neighbors in the device name field, which will be acquired by the nodes in its proximity. The string used for this purpose is the following:

`SGF_ID-GOAI-ID1#ID2# ··· #IDk-X`

which is an extension of the name adopted during the NIC phase. After the GOAI string, coded as in the NIC phase, a device includes the list of all the neighbor IDs separated by '#'. The final 'X' character is the final string delimiter and it indicates whether the advertising device is ready ('R') or not ('N') for the following phase. This field is initially set to 'N'. Each device monitors its neighbors by checking the neighbor list and the `X` value in their device names. Once a device receives the advertisements from all discovered neighbors (i.e., those that appear in the list it advertises), it will set `X` equal to 'R'. As soon as all the neighbors of a device have turned their `X` flag to 'R', the device will enter the RS phase. Notably, this procedure does not require bidirectional visibility since the final 'R' decision is taken at a node independently of the fact that the neighbor nodes are able to see it.

C. Role Selection (RS)

During this phase, a device follows the topology formation approach described in Sec. V-B in order to set its role. At the outset, each device runs BIDIRECTIONAL-ASSESSMENT to compute the list of neighbor nodes with bidirectional visibility, using the neighbor information advertised by others. Then PRE-ROLE-SELECTION makes a preliminary decision on becoming candidate GO. Whenever the decision is taken, the result is reported in the advertised device name, according to the following format: `SGF_ID-GOAI-Y`, where `Y` can be either the string `Client` when the device declines to become a GO or `Marked` whenever the device becomes a candidate GO.

When all neighbors have advertised their decision, the node enters the FINAL-ROLE-SELECTION. At the end, each device advertises its final decision in the device name with the following format: `SGF_ID-GOAI-Role`, in which the `Role` can be either `Client` or `GO`. After setting its role, each client waits for all of its neighbors to finish the SGF procedure by checking their device names, and then it enters the next phase directly. The GOs, after all its neighbors have finished their role selection, add a numerical `nGOs` value in the device name field, as follows: `SGF_ID-GOAI-Role-nGOs`. This field is used to advertise the number of neighbor GOs of each GO; a GO enters the next phase only after it has advertised its `nGOs` value.

D. Connection (CO)

In this phase, devices setup connections between each other so as to establish the final multi-group network. The main idea is to exploit the number of neighbor GOs as a priority to determine how to build the network backbone that will be used to route the traffic between the groups. The behavior of a device in this phase depends on the role selected in the previous one. We therefore describe the behavior of the devices depending on whether they are GO or clients.

Connection at the GO. Consider a generic node x selected as GO. If x is associated at layer-2 to another GO, x is tagged

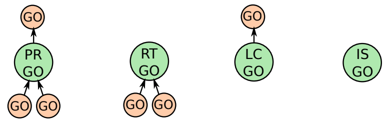


Fig. 12: All the possible cases for the GO sub-roles when/whether connected to other GOs.

with a sub-role denoted as *legacy client* (LC). If other GOs have a layer-2 association to x, then x is tagged with a sub-role denoted as *Root*. Depending on how/whether a GO is connected to other GOs, x will take on one of the four sub-roles depicted in Fig. 12: an *LC-GO* acts just as legacy client; an *RT-GO* acts only as a Root; a *PR-GO* acts as a parent GO, i.e., x is a legacy client of another GO and there is a GO that is connected to x as a legacy client; an *IS-GO* is an isolated GO, i.e., not connected to any other GO.

A generic GO x decides its sub-role as follows. As a first step, x looks for neighbor GOs. If there is none, x becomes an IS-GO. Otherwise, it checks the `nGOs` values advertised by all the neighbor GOs and compares them against its own. GO x becomes an RT-GO if it has the highest `nGOs`. Else, x tags itself as an LC-GO and connects to the neighbor GO with the maximum `nGOs` (ties are solved based on the GOAI and, if needed, on the MAC address). Such node is referred to as *target* of x; next, x adds a subfield, called `RT_Target`, to its device name and sets it to the MAC address of its target node so that it can be advertised to its neighbors. The device name advertised through the Device Discovery procedure now has the following format: `SGF_ID-GOAI-Role-RT_Target/Accepting-nGOs -CLReady`. Then x searches among all its neighbor GOs for a device whose name contains x's MAC address in the `RT_Target` field. If such a GO exists, x becomes a PR-GO.

After having selected their sub-roles, RT-GOs, LC-GOs and PR-GOs cooperate to construct the routing backbone across different groups. Specifically, each RT-GO firstly sets up a group, defines a *service* encoding the credentials of its own group, and disseminates this information through the standard P2P Service Discovery. Recall that, as described in Sec. II, a device can acquire the information of a service even if it is not connected to the service provider. After broadcasting its credentials, each RT-GO sets the `Accepting` field in the device name to 'R' (i.e., Ready) to notify the availability to accept incoming connection requests. Only after such notification, the LC-GOs and PR-GOs start connecting to their GO targets. Once a legacy connection with its target GO is established, a PR-GO follows the same steps taken by an RT-GO: it creates a group, broadcasts the group credentials and sets its `Accepting` field to 'R'.

In order to let the client devices join the group, a GO turns the `CLReady` field to 'R'. IS-GOs, not being involved in the backbone establishment, set their `CLReady` field to 'R' as soon as the group has been established. Instead, an RT-GO or a PR-GO does so only when all LC-GOs and PR-GOs, for which it is a target, have connected.

Connection at clients. We now describe the behavior of a client node after the RS phase. Each client ranks its neighbor GOs based on their GOAI and tries to connect to the neighbor GO with the highest GOAI. Note that, before connecting to the GO, a client has to wait for the target to change its `CLReady` field to 'R'. To find a reasonable tradeoff between choosing the best GO and the time required to join a group, after a timeout the client restarts the association procedure selecting the GO ranked just after the previous target GO.

Since clients act asynchronously, concurrent connection requests may collide/overlap in time. Indeed, establishing a connection between two devices takes some time and, according to the Wi-Fi Direct application-layer protocol, connection requests arriving during a connection establishment fail. To reduce the collision/overlap probability, we have implemented a back-off mechanism to stagger connection requests.

E. Relay Client Selection

In the Android implementation of the Wi-Fi Direct standard, a GO is notified about any new client joining the group. If no more clients join the group for a given time interval, a GO selects the Relay Client among the associated clients. Note that waiting for the group to become stable allows the GO to have a comprehensive knowledge of the clients. Very simply, the GO chooses as Relay Client the client with the highest GOAI and advertises its decision by transmitting a *Relay Client appointment* message at the application level including the MAC address of the selected node. The GO will retransmit the message till it receives an acknowledgment from the client matching the MAC address, or a maximum number of retransmissions have been reached. In the latter case, the GO will then select another Relay Client based on the GOAI ranking.

VII. Performance of Smart Group Formation

We study the performance of the smart group formation through both numerical simulations and experiments on a real testbed.

A. Numerical evaluation

We developed a Matlab simulator to generate random geometric graphs and to implement the Pre-Role-Selection and Final-Role-Selection phases described in Sec. V-B. Each vertex in the graph represents a portable device and an edge exists between two vertices if and only if they are in each other's radio range. To simulate the GOAI, each vertex is also assigned a random integer in $[32, 127]$. The vertices are distributed in a two-dimensional space where the x and y coordinates are chosen uniformly at random in the range $[-40, 40]$ m, thus the maximum distance is around 113 m. We vary the total number of nodes and the transmission range, which is assumed to be the same for all nodes.

We focus on the *GO-ratio*, defined in the interval $[0, 1]$ as the ratio between the final number of GOs generated through the role selection and the total number of nodes in the network. We discarded all the sample graphs exhibiting node partitions

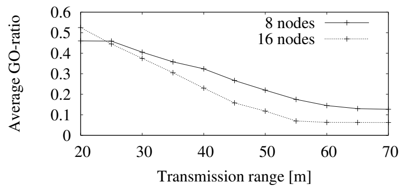


Fig. 13: GO-ratio on random geometric graphs and random values of GOAI.

and run 100 simulations for each scenario, each run with a different graph and assigned GOAIs.

Fig. 13 shows the average GO-ratio with respect to the transmission range for 8 and 16 nodes. When increasing the transmission range, the graph becomes more connected and thus fewer GOs are required in order to build a connected backbone; this explains the decreasing behavior of both curves. When the transmission range is large enough, the graph becomes fully connected and thus just one GO is required to cover all nodes; in this case, the GO-ratio becomes 0.125 and 0.0625, for 8 and 16 nodes respectively. On the contrary, when the transmission range is too small, the graph is sparse and the node degree is low, thus the choice of GOs is very limited. This explains why we observe a constant GO-ratio for 8 nodes and transmission range ≤ 25 m.

B. Experimental evaluation

We adopted the same testbed described in Sec. IV-A to test the actual implementation of SGF. Specifically, we set up different topologies, each composed of four devices and, for every topology, we tested the time required by each of the four phases (i.e., NIC, NA, PE, CO) in the SGF procedure.

We addressed many experimental hurdles. Firstly, in order to test a specific desired topology, we had to force the devices to neglect certain neighbors. Secondly, the devices start the formation procedure at different time instants. Recall that devices are synchronized by the `X` flag when transiting into the RS phase, after the NA phase; thus, the formation starting time has an impact on the RS and NA phases. Thirdly, during the CO phase, the GOs and clients have distinct behaviors. For a GO, we considered that the phase ends when the flag `CLReady` is turned into 'R', i.e., when the local backbone has been established. For a client, we considered the finishing time when it successfully connects to the target GO.

We mainly focused on three different topologies and, for each topology, we carried out 30 independent experiments. The time required for each phase was measured at the application level. All the average results were obtained with an accuracy $\leq 1\%$, evaluated as the relative width of the 95% confidence interval.

We first focus on the full-connected topology shown in Fig. 14(left) where all the four devices can hear each other. During the RS phase, device 4 has the largest GOAI and becomes GO; all other devices choose to be clients and connect to it. Table II(left) shows the time elapsed at each phase and the total time. Recall that a device ends the NIC phase and enters the NA phase as soon as it discovers enough neighbors within given intervals. During our tests, all the devices found the others in 2 seconds. Since all the devices start the formation almost at the same time, the durations of the NA phase of the four devices are similar. Notably, NA is the longest phase, due to the temporal thresholds adopted in the process. Recall that a device finishes the RS phase only when all of its neighbors have selected their roles, thus devices act synchronously at the end of this phase; it follows that the devices experience the same duration of the RS phase. Since device 4 is the only GO in this phase, it becomes an IS-GO with an immediate decision and it is the first to end the SGF procedure, after 26 s. The duration of the final CO phase is not negligible due to the time needed by the adopted protocol and that required for the device name update and advertisement in the Android implementation of Wi-Fi Direct. Thus, all the other clients end the procedure after 39 s.

The second topology, shown in Fig. 14(center), is asymmetric: devices 1, 2 and 3 can hear each other, whereas device 4 is in radio proximity of device 2 only. As expected, according to our SGF, during the RS phase device 2 becomes an IS-GO because of its centrality, despite its smaller GOAI, and all the other devices become clients connected to it. Table II(center) presents similar results to the full-connected scenario in Table II(left), since all the clients take around 42 s and the IS-GO around 27 s. As in the previous case, the clients take longer due to the CO phase.

The last topology, shown in Fig. 14(right), is linear and each device sees just a subset of neighbor nodes. The experimental results are reported in Table II(right). Now both devices 2 and 3 become GOs since they are directly connected and have different neighborhoods. Also, the two GOs must coordinate to create the communication backbone, hence the duration of the CO phase is no longer negligible. Since node 2 has a higher GOAI than node 3, it includes the credentials of its group in a message and broadcasts the message to advertise its service to the nodes in its proximity. When device 3 receives this information, it uses the credentials to connect to device 2 (which has a higher GOAI than itself) as a legacy client (LC). As a final result, device 2 tags itself as RT-GO and device 3 as LC-GO. Devices 1 and 4 become clients and wait until the backbone is established. We remark that this scenario is representative of a large wireless network, in which nodes are very far from each other and our proposed multi-group scheme is the only viable approach to let the devices form a fully-connected network.

VIII. Related Work

Several recent studies have investigated the features and the performance of the Wi-Fi Direct technology.

One of the first studies has appeared in [12], where Camps Mur et al. consider a single-group Wi-Fi Direct network with the group owner sharing access to a 3G network with a set of connected devices. The work analyzes the power saving protocols defined in Wi-Fi Direct and design two algorithms that use such protocols to save energy while providing good throughput

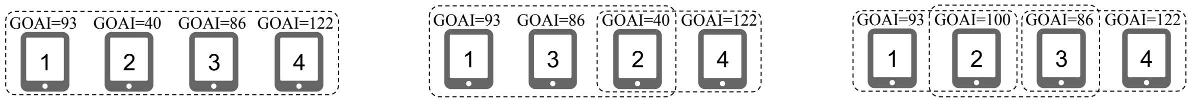


Fig. 14: Tested topologies: a full-connected topology with one area of proximity (left); asymmetric topology with two areas of proximity (center); a linear topology with three areas of proximity (right).

TABLE II: Experimental results for the full-connected (left), asymmetric (center) and linear (right) topology of Fig. 14.

Phase	Duration $[s]$			
	Device 1	Device 2	Device 3	Device 4
NIC	2.0	2.0	2.0	2.0
NA	18.1	17.9	17.9	18.0
RS	6.8	5.8	5.8	6.1
CO	12.1	11.3	13.5	0.0
Total	39.0	37.0	39.2	26.1

Phase	Duration $[s]$			
	Device 1	Device 2	Device 3	Device 4
NIC	2.0	2.0	2.0	2.0
NA	18.8	18.3	18.4	18.0
RS	4.6	6.8	4.5	4.1
CO	16.5	0.0	16.5	17.5
Total	41.9	27.1	41.4	41.6

Phase	Duration $[s]$			
	Device 1	Device 2	Device 3	Device 4
NIC	2.0	2.0	2.0	2.0
NA	16.8	20.4	17.7	17.5
RS	3.3	14.0	11.3	4.9
CO	50.4	19.9	27.7	74.8
Total	72.5	56.3	58.7	99.2

performance. An improved power management scheme for Wi-Fi Direct is proposed in [13], which dynamically adapts the duty cycle of P2P devices to the properties of the application to be supported.

An overview and experimental evaluation of Wi-Fi Direct using two laptops running Linux is presented in [14], where the emphasis is on the standard group formation procedures and the performance that they exhibit in terms of delay and power consumption. Group formation is also the focus of the work in [15], which investigates the ability to create opportunistic networks of devices using Wi-Fi Direct to establish communication links. The performance of group formation is studied experimentally, by varying the protocol parameters and considering scenarios that are typical of opportunistic networks. A preliminary study of multi-group physical topologies of Wi-Fi Direct networks can be found in our previous work [16], where however only some of the limitations of the Android OS are investigated and only unidirectional D2D communication is tackled.

The use of Wi-Fi Direct as a D2D technology to be integrated into LTE and LTE-A cellular networks is explored in [4], [5], [17]. In particular, while [4] mainly focuses on architectural issues, [5] and [17] also quantify the estimated network performance gains from offloading cellular traffic onto Wi-Fi Direct-based, D2D connections.

As far as content dissemination and sharing in mobile ad-hoc networks are concerned, a number of solutions have been proposed in the literature, e.g., [18]–[20]. However, very few works exist that specifically address Wi-Fi Direct-based networks. Among these, the study in [21] presents a Wi-Fi Direct-based overlay architecture for content sharing among peers belonging to the same group. In particular, they leverage the P2PSIP protocol, which enables real-time communication using the application-layer signaling protocol SIP in a peer-to-peer fashion. The work in [22], instead, implements the decentralized iTrust mechanism [23] for information publication and retrieval. In particular, it proposes a peer management technique to facilitate group creation and allow peers to set up and maintain connectivity over Wi-Fi Direct. Another approach for D2D communication on smartphones is proposed in [24]. It leverages the tethering functionality on smart phones to setup access points. Although this solution does not require rooted devices, as in our proposed scheme, it does not support concurrent inter-group communications. Indeed, in the tethering mode only one 802.11 network interface is locally available and a device cannot operate in two groups simultaneously. To act as relay node between two groups, a node must disconnect from one group and associate to the other, introducing very large latencies in the process (1-10 seconds).

As mentioned, to the best of our knowledge, none of the existing works has investigated, solved and experimentally evaluated bidirectional communication in Wi-Fi Direct multi-group networks. As complementary approach, LTE Direct [25] enables D2D communications among nodes in the proximity, but, differently from our scenario, it requires the cooperation of the telecom operators. Note also that this technology is still not supported by commercial smartphones.

IX. Conclusions and Future Work

We designed and implemented bidirectional, multi-group communication in Android devices supporting the Wi-Fi Direct protocol. This allowed us to extend the achievable communication range for a protocol whose current implementation in off-the-shelf, non-rooted Android devices has been tailored just to single group D2D communication.

In particular, we proposed a solution to overcome the limitations of the physical Wi-Fi Direct network topology and of its addressing plan, and we built a logical topology that enables bidirectional inter-group data transfers. The logical topology we devised is based on a cooperative traffic relaying scheme among adjacent groups and, through transport-layer tunnels, leads to the formation of a network backbone that provides full network connectivity. We implemented our solution in Android and validated it by implementing a content-centric routing scheme, which properly exploits the above backbone, and by developing a testbed comprising a heterogeneous set of devices. To our knowledge, this is the first work that enables a content-centric network with multi-group communication on legacy smartphones, without the facilitation of existing infrastructure.

We then proposed a smart group formation mechanism according to which devices can establish a physical multi-group network in a fully distributed manner, focusing on network performance such as group sustainability, data transfer throughput and network coverage. We designed a role selection algorithm, run by each node to autonomously determine the role it should take. Also, in order to establish the logical topology required for inter-group communication, our mechanism includes a procedure to select properly Group Owners and Relay Clients. Finally, we implemented our smart group formation mechanism by a multi-phases operation on non-rooted Android devices, exploiting only the standard procedures defined in Wi-Fi Direct and the existing Wi-Fi Direct functionalities on Android OS.

Our work opens up several future research directions. An in-depth study could be carried out to determine the system scalability with the number of network devices. Such study could also factor in the choice of nodes to be selected as Relay Clients (their number and typology) as well as the techniques to efficiently manage the consequences of node churning. Furthermore, bidirectional inter-group communication can be the basis for disruptive cooperative applications and service models. Finally, the device association in Wi-Fi Direct requires certain security credentials. Further studies are needed to understand how to distribute the security credentials seamlessly among the devices.

ACKNOWLEDGMENTS

This work was supported by *Fondazione CRT* under the initiative "La Ricerca dei Talenti", through the MIMOSE project.

REFERENCES

[1] "Bluetooth low energy," http://www.bluetooth.com/Pages/low-energy-tech-info.aspx, 2014.

[2] "Wi-Fi Direct Alliance," http://www.wi-fi.org/, 2014.

[3] G. RP-122009, "Study on LTE device to device proximity services," 3GPP TSG RAN Meeting #58, 2012.

[4] A. Asadi and V. Mancuso, "WiFi Direct and LTE D2D in action," in *IFIP Wireless Days (WD)*, Valencia, Spain, 2013, pp. 1–8.

[5] S. Andreev, A. Pyattaev, K. Johnsson, O. Galinina, and Y. Koucheryavy, "Cellular traffic offloading onto network-assisted device-to-device connections," *IEEE Communications Magazine*, vol. 52, no. 4, pp. 20–31, 2014.

[6] C. Casetti, C. F. Chiasserini, L. C. Pelle, C. D. Valle, Y. Duan, and P. Giaccone, "Content-centric routing in Wi-Fi Direct multi-group networks," in *2015 IEEE WoWMoM*, June 2015, pp. 1–9.

[7] B. Chen, K. Jamieson, H. Balakrishnan, and R. Morris, "Span: An energy-efficient coordination algorithm for topology maintenance in ad hoc wireless networks," *Wireless networks*, vol. 8, no. 5, pp. 481–494, 2002.

[8] I. Stojmenovic, M. Seddigh, and J. Zunic, "Dominating sets and neighbor elimination-based broadcasting algorithms in wireless networks," *IEEE Transactions on parallel and distributed systems*, vol. 13, no. 1, pp. 14–25, 2002.

[9] F. Dai and J. Wu, "Distributed dominant pruning in ad hoc networks," in *Communications, 2003. ICC'03. IEEE International Conference on*, vol. 1. IEEE, 2003, pp. 353–357.

[10] "Wi-Fi certified Passpoint," http://www.wi-fi.org/discover-and-learn/, 2013.

[11] "How Android Wifi state machine works," http://jhshi.me/2014/04/25/how-android-wifi-state-machine-works/, 2014.

[12] D. Camps-Mur, X. Perez-Costa, and S. Sallent-Ribes, "Designing energy efficient access points with Wi-Fi Direct," *Computer Networks*, vol. 55, no. 13, pp. 2838–2855, 2011.

[13] K.-W. Lim, W.-S. Jung, H. Kim, J. Han, and Y.-B. Ko, "Enhanced power management for Wi-Fi Direct," in *Wireless Communications and Networking Conference (WCNC)*, 2013, pp. 123–128.

[14] D. Camps-Mur, A. Garcia-Saavedra, and P. Serrano, "Device-to-device communications with Wi-Fi Direct: overview and experimentation," *IEEE Wireless Communications*, vol. 20, no. 3, pp. 96–104, 2013.

[15] M. Conti, F. Delmastro, G. Minutiello, and R. Paris, "Experimenting opportunistic networks with WiFi Direct," in *IFIP Wireless Days (WD)*, Valencia, Spain, 2013, pp. 1–6.

[16] Y. Duan, C. Borgiattino, C. Casetti, C. Chiasserini, P. Giaccone, M. Ricca, F. Malabocchia, and M. Turolla, "Wi-Fi Direct multi-group data dissemination for public safety," in *World Telecommunications Congress (WTC)*, Berlin, Germany, June 2014.

[17] A. Pyattaev, K. Johnsson, A. Surak, R. Florea, S. Andreev, and Y. Koucheryavy, "Network-assisted D2D communications: Implementing a technology prototype for cellular traffic offloading," in *IEEE Wireless Communications and Networking Conference (WCNC)*, Istanbul, Turkey, April 2014.

[18] M. Fiore, C. Casetti, and C. Chiasserini, "Information density estimation for content retrieval in MANETs," *IEEE Transactions on Mobile Computing*, vol. 8, no. 3, pp. 289–303, 2009.

[19] P. Meroni, E. Pagani, G. Rossi, and L. Valerio, "An opportunistic platform for Android-based mobile devices," in *Proceedings of the Second International Workshop on Mobile Opportunistic Networking*, ser. MobiOpp. ACM, 2010, pp. 191–193.

[20] P. Tiago, N. Kotilainen, and M. Vapa, "Mobile search - social network search using mobile devices demonstration," in *IEEE Consumer Communications and Networking Conference (CCNC)*, January 2008, pp. 1245–1245.

[21] T. Duong, N.-T. Dinh, and Y. Kim, "Content sharing using P2PSIP protocol in Wi-Fi Direct networks," in *International Conference on Communications and Electronics (ICCE)*, 2012, pp. 114–118.

[22] I. Lombera, L. Moser, P. Melliar-Smith, and Y.-T. Chuang, "Peer management for iTrust over Wi-Fi Direct," in *International Symposium on Wireless Personal Multimedia Communications (WPMC)*, 2013, pp. 1–5.

[23] I. Lombera, L. Moser, P. Melliar-Smith, and Y. Chuang, "Mobile decentralized search and retrieval using SMS and HTTP," *Mobile Networks and Applications*, vol. 18, no. 1, pp. 22–41, March 2013.

[24] S. Trifunovic, B. Distl, D. Schatzmann, and F. Legendre, "WiFi-Opp: ad-hoc-less opportunistic networking," in *Proceedings of the 6th ACM workshop on Challenged networks*. ACM, 2011, pp. 37–42.

[25] "LTE Direct," https://ltedirect.qualcomm.com/, 2015.