RIC-Apps Conflict Management

White Paper

Hammad Zafar, Ehsan Tohidi, Martin Kasparick (Fraunhofer HHI),

Boris Lorbeer (Technische Universität Berlin),

Heiko Lehmann, Matthias Weh (Deutsche Telekom),

Gunja Rastogi, Jonas Charaf, Monika Tarwala (Capgemini),

Adrian Kliks (Rimedo Labs),

Daniyal Amir Awan (NOKIA),

Kaim Munshi (Vodafone)

This content has been produced by

Contents

1 Introduction and Motivation

The design and development of the Open Radio Access Network (O-RAN) disaggregated architecture has introduced flexibility and innovation in conventional cellular networks by separating network components and establishing standardized interfaces. This openness enhances interoperability, promotes vendor diversity, and expands the possibilities for network evolution by allowing the interchangeability of O-RAN components manufactured by various vendors without limiting network functionality. This is a significant change compared to traditional proprietary RANs, where hardware and software are typically provided by a single vendor. A unique aspect of O-RAN is the introduction of new components that can influence RAN operation, notably the Non-Real Time (Non-RT) and Near-Real Time (Near-RT) RAN Intelligent Controllers (RICs). These RICs react to current and forecast future network states and adapt parameters to optimize performance. They host intelligent applications known as rApps and xApps, which can be developed by independent parties and deployed on any O-RAN-compliant platform. These applications aim to influence network operations from the RICs, enhancing network functionality and performance, and facilitating comprehensive RAN automation. However, the introduction of multiple Apps (x/rApps), each with its own objectives and decision-making processes, presents new challenges in network management. As these applications independently interact with the network environment, the actions of one or more applications could potentially adversely impact the objectives of others. These conflicting actions and decisions necessitate robust conflict detection and mitigation mechanisms. This complexity is further heightened by the fact that these Apps can be developed by multiple vendors, each with different design objectives and implementation. The multi-vendor environment introduces additional layers of variability and unpredictability, making conflict mitigation even more challenging.

This work highlights the importance of having a robust framework for conflict detection and resolution within the RICs. This framework is essential to manage the diverse and dynamic nature of network applications and to maintain the stability and performance of the network. Several key factors emphasize the critical need for research in this area are:

- **Standard and Gap Analysis:** While the O-RAN Alliance has made significant strides in defining specifications, gaps remain in addressing the complex interactions between multiple intelligent Apps. A thorough analysis of current standards and their limitations is essential to identify areas requiring further development.
- **Conflicting Objectives:** Apps, designed to optimize different aspects of network performance, may have conflicting goals. For instance, an application focused on energy efficiency may conflict with another that prioritizes capacity maximization. Understanding and managing these conflicting objectives is crucial for maintaining overall network stability and performance.
- **Interoperability Challenges:** The multi-vendor environment promoted by O-RAN introduces interoperability challenges. Ensuring seamless operation between components from different vendors, particularly in the context of

intelligent applications, requires advanced coordination and conflict resolution mechanisms.

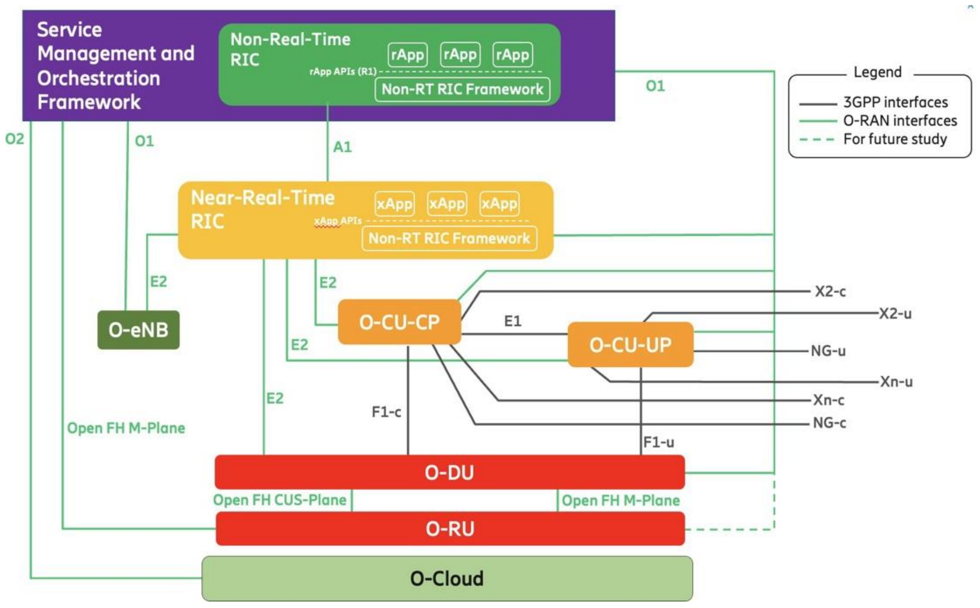

Figure 1: O-RAN architecture [O-R]

- **Detection and Mitigation Techniques:** As the complexity of interactions between intelligent Apps grows, so does the need for sophisticated detection and mitigation techniques. Exploring various approaches, from rule-based systems to advanced machine learning algorithms, is vital for developing effective conflict management strategies.

This paper aims to address these challenges by proposing a comprehensive framework for conflict detection and resolution within the Near-RT RIC ecosystem. Our goal is to contribute to the development of flexible, intelligent, and adaptive control functionalities that optimize RAN performance while managing the complexities introduced by multiple intelligent Apps.

2 Conflicts and Challenges

In this section, we delve into the different types of conflicts that can arise within the O-RAN architecture, exploring them from various angles to provide a comprehensive understanding of the challenges they present. Each conflict type has a unique set of challenges that must be addressed to ensure smooth and efficient network operation. To make these conflict types more relatable and actionable, we provide specific use cases that demonstrate how they manifest in real-world O-RAN scenarios. These examples highlight the practical implications of each conflict type and underscore the importance of developing effective conflict management strategies to enhance O-RAN's functionality and reliability.

2.1 Vertical vs. Horizontal conflicts

One key challenge in disaggregated and open architecture, where various stakeholders are allowed to deliver their own applications, is the need for effective conflict management. Residing within the RICs, xApps and rApps are, by assumption, tailored to realize specific goals within the wireless network. This is typically achieved by observing the network status and then, based on these observations, modifying selected parameters to steer the desired network behavior. When multiple applications are delivered by the same vendor, the risk of potentially conflicting decisions is minimized; properly designed solutions should not generate counteracting decisions or even suggestions to the decision module within the RIC. However, in typical situations, when different vendors provide applications, the probability of conflicting situations could be very high, and standardized procedures are necessary to minimize their negative impact.

Let us stress that these observations are very generic and refer to any situation where two or more applications cause conflicting changes in network behavior; they do not reflect either the type of the conflict or the location (in the network architecture) where the conflict appears. One of the most popular ways of conflict classification is to split them into three groups depending on their directness of interaction [BdPDB$^+$24, AK23, AK]: We distinguish direct, indirect, and implicit conflicts, and their detailed definition in terms of the O-RAN Alliance is provided later in this section. However, one may also classify the conflicts into vertical and horizontal ones [AK23] depending on the relation between the RICs, which contain and manage the conflicting applications (see Fig. 3). In detail, when conflicts appear at the same time-scale level within the O-RAN architecture (e.g., between rApp 1 and rApp 2, or between xApp 1 and xApp 2), they are treated as horizontal conflicts. On the other side, when a conflict exists between an rApp and an xApp (so between Non-RT RIC and Near-RT RIC, and both RICs are dedicated to controlling the same part of the wireless network), it is an example of a vertical conflict.

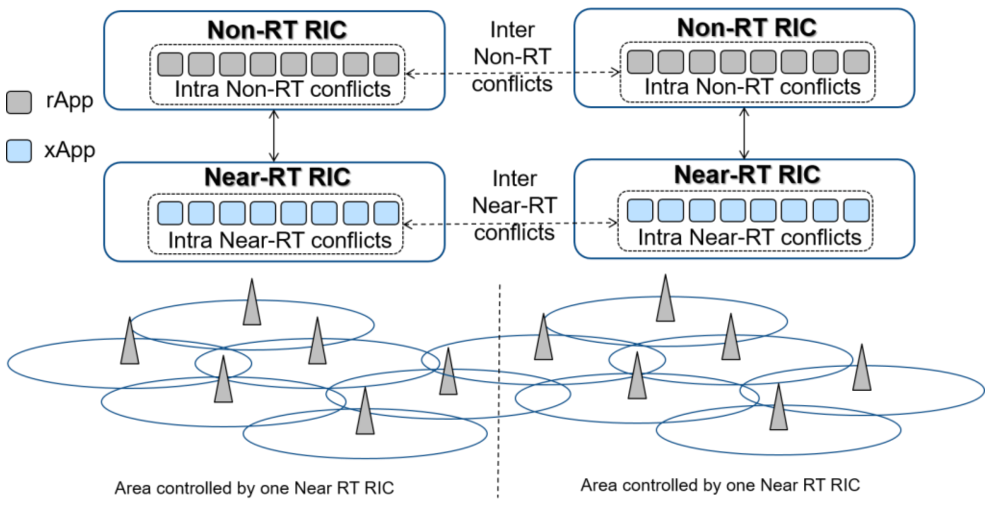

Figure 2: Illustration of intra- and inter-RIC conflicts

At the same time, conflicts may appear within one RIC (regardless of its type) or between RICs that control different parts of the network (see Fig. 2). In that context, one may distinguish between inter- and intra-RIC conflicts. For example, when there is a horizontal conflict between two xApps residing in two different RICs (each responsible for controlling another part of the network), this can be described as an inter Near-RT RIC conflict. The same conflict appearing between these same applications, but this time residing within the same RIC, can be classified as an intra Near-RT RIC conflict. An analogous classification can be applied to Non-RT RICs and rApps.

2.2 Conflicts in Near-RT RIC by O-RAN Alliance

The O-RAN alliance WG3 [ORA23] categorizes the conflicts appearing in the Near-RT RIC into:

Direct Conflicts: These conflicts appear when multiple xApps try to update the same control parameters. These conflicts may be observed immediately, e.g., two or more xApps propose setting different configurations to the same control parameter.

Example: Load Balancing vs. Energy Efficiency

The load balancing (LB) xApp minimizes the maximum load at each RU, thus activating the maximum number of RUs, whereas the energy efficiency (EE) xApp aims to activate a minimum number of RUs to serve the users in the network. The LB xApp minimizes the resource utilization at each RU by optimizing the assignment of UEs to RUs. This involves considering factors such as the users' quality of service (QoS) requirements, traffic demands, and network conditions. The EE xApp aims to activate the fewest possible RUs while minimizing the overall energy consumption of the network. This involves dynamically adjusting UE-RU associations based on real-time traffic demands, network conditions, and evolving user requirements. Both xApps optimize the UE-RU assignments based on their respective goals, which may lead to a direct conflict.

- **Indirect Conflicts:** These conflicts are not apparent, but some dependence between the xApp target parameters can be seen. Such conflicts appear when multiple xApps update different control parameters that have some dependencies and affect the same metrics of the network. For instance, various xApps target different configuration parameters to maximize the same performance metric. Although this won't lead to incompatible parameter settings, the xApps' action may still have an undesired or unexpected effect on the network.

  Example: Interference Management vs. Load Balancing

  An interference management (IM) xApp aims to reduce co-channel interference between neighboring cells to improve signal quality and overall user experience. At the same time, the LB xApp will try to evenly distribute network traffic across multiple cells to avoid congestion and ensure that no single cell is overloaded. The conflict will happen once the LB xApp decides to offload traffic from a congested cell to a neighboring cell that is already experiencing high levels of interference. The IM xApp might respond to the increased traffic by further adjusting power levels or frequencies in the target cell, which could inadvertently lead to higher interference levels in other parts of the network, affecting overall network stability and overall QoS. This conflict involves direct and indirect elements due to the cascading effects on network stability.

- **Implicit Conflicts:** These conflicts cannot be observed directly, and even the dependence between xApps is not obvious, e.g., different xApps may optimize different metrics and parameters. However, optimizing one metric may have implicit, unwanted, and maybe adversary side effects on one of the metrics optimized by another xApp.

  Example: Energy Efficiency vs. Slice Management

  The ultimate goal of the EE xApp is to minimize the overall energy consumption in the network; depending on the assumed optimization goals and operator's requirements, EE xApp may achieve it in different ways, for example by initiating actions leading to switching off the RUs or part of the MIMO board. Simultaneously, the slice management (SM) xApp will aim to fulfill high resource utilization requests to support bandwidth-intensive applications. These two applications, i.e., EE and SM, will influence the throughput experienced by the users. The EE xApp focuses on reducing energy consumption by minimizing the active RUs, while the SM xApp seeks to maximize the resources to improve the user throughput.

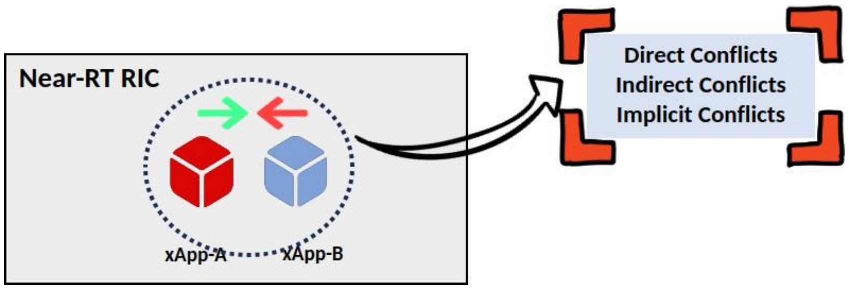

Figure 3: Types of conflicts by O-RAN Alliance

2.3 Impact of Conflicts

The categorization of conflicts in the preceding discussion is of an analytical and structural nature and serves as a starting point for the engineering of detection and mitigation approaches. From an operator's point of view, though, this needs to be complemented with operational and techno-economic considerations. In this sub-section, we capture this perspective by discussing principal metrics. Although these are, as yet, lacking a thorough mathematization, they are intuitive and comprehensible. These metrics can be applied to assess conflicts in general.

- **Severity:** Conflicts need to be given a quantitative measure describing their relevance for the actual operation of a network. This measure shall reflect the dimensions *stability/performance* and *cost effectiveness*. These, in turn, relate to different time scales: while in the real-time or near-real-time scale, stability will be the governing criterion, in the non-real-time scale, efficiency considerations will rule. Conflicts with negligible severity measures may be regarded as artifacts. Operators need to develop a time-resolved conflict management approach reflecting their techno-economical priorities.

- **Reach:** Conflicts need to be quantitatively described with respect to their propagation dynamics in the network considered. While the above distinction *direct – indirect implicit* addresses this need to some extent, a more concrete mapping of network distance measures is called for. Here, a clear conceptual separation of different space definitions must be maintained: while real space reach addresses the effects of considered conflicts in the served territories of the network, the network space itself is a graph representing the network components. In the latter, conflict effects cascade over-involved network components. Both space concepts are needed for the operator to derive a detailed risk management approach.

- **Vulnerability Creation:** This criterion relates the hitherto merely logical quality of conflicts to the emergence of a qualitatively new attack surface, i.e., the exploitation of conflicts, or even a mere conflict potential by a malevolent attacker. Recent research has shown that AI-based network management routines may be manipulated in novel ways to craft a new class of attacks. Obviously, the vulnerability creation dimension is more encompassing than the logical control conflicts themselves, containing also dormant conflict potential of “well-behaving” Apps. The operator needs to monitor these conflict potentials in an x/rApp landscape in a preemptive way to secure the network operation. The first step in this direction is a “Network AI/ML Sensitivity Analysis” employing network digital twin tools.

3 Conflict Management

Conflict management is a critical aspect of xApp deployment and operation in O-RAN environments, encompassing both conflict detection and mitigation strategies. Effective conflict management ensures the harmonious operation of multiple xApps, optimizing network performance and maintaining service quality. The following subsections will delve into how conflict management for O-RAN differs from traditional Self-Organizing Networks (SON). This distinction is crucial, as O-RAN introduces new challenges and opportunities in managing conflicts due to its open, disaggregated architecture and the potential for third-party xApp development. Moreover, this discussion encompasses both the operator’s and xApp developer’s viewpoints. From the operator’s standpoint, we examine the need for robust conflict management mechanisms to ensure network stability and maintain service level agreements (SLAs). From the xApp developer’s approach, we highlight the importance of implementing conflict-aware designs that enable their applications to coexist harmoniously and collaborate effectively within the dynamic O-RAN ecosystem. The subsequent sub-sections will explore various strategies for conflict detection and mitigation, taking into account the unique characteristics of O-RAN and the diverse ecosystem of xApps. These strategies aim to provide a comprehensive framework for identifying potential conflicts and resolving them efficiently, contributing to a more resilient and optimized O-RAN deployment.

3.1 Requirements and KPIs

To ensure the effectiveness of a conflict mitigation framework, it is crucial that the key performance indicators (KPIs) and requirements are defined and driven by the operators and xApp developers. These stakeholders possess the in-depth knowledge and practical experience needed to identify potential conflicts and tailor the framework to address real-world challenges. By involving both operators and xApp developers in the process, the framework can be designed to meet specific operational needs and ensure that it effectively mitigates conflicts, leading to more reliable and optimized system performance.

From the operator’s perspective, understanding the impact of conflicts is essential for deriving the corresponding operational requirements. To ensure that these requirements are effectively addressed, it is important to prioritize them systematically. The following table presents these requirements in a priority-ordered manner, where each lower-level requirement should only be considered if it does not conflict with those ranked higher. This approach helps to maintain a clear hierarchy and ensures that the most critical operational needs are met without compromise.

Requirement	Time Scale	Description
Operational stability	(near) real-time	Close to physics, any running xApps ensemble must be conflict-free concerning UE handling. Conflict avoidance costs are considered secondary. The logical coupling of any xApp to Non-RT RIC, demanding iterative action, is disregarded.
Network security	all scales	The conflict exploitation potential of any x/rApp ensemble needs to be assessed upfront and monitored at runtime. Novel AI/ML-based attack vectors need to be constantly studied in appropriate network digital twin tools.
Third party security	all scales	Any attack vector to corrupt customer-facing service or attack third party data needs to be studied up-front and monitored during runtime.
Operational efficiency	(near) real-time	Any running xApp ensemble in the Near-RT RIC is required to operate in a lean way. Possible action space overlaps need to be studied up-front so that at runtime, any redundant or circular action is suppressed.
Network performance	near-real-time, non-real-time	Customer-facing KPIs such as throughput, latency, and quality of service need to be maintained at a level specified by the operator. Underlying x/rApp conflicts need to be managed in a way to serve this goal. This also entails that rigorous solutions are explicitly not called for any management strategy leaving the customer-facing KPIs intact is good enough.
Minimal Total Cost of Ownership (TCO)	life-cycle scale	Relating to a budget year, or even to the life-cycle span of field elements, aggregate effort/benefit considerations reflect on x/rApp conflict management: any mitigation strategy accruing undue cost on a TCO-scale need to be reconsidered.

The perspective of an xApp developer complements the operator's view by focusing on the policies, goals, and SLA requirements that form the foundation for software providers. Depending on the operator's requirements and expectations, the xApp and rApp providers should put more emphasis on these specific aspects and adjust the solutions accordingly. However, some generic observations can be discussed as well.

First, various approaches to identify, mitigate, and/or avoid conflicts are possible. The conflict management module within the RIC may make all the decisions and be solely responsible for all processes related to conflict management. On the other hand, xApps (or applications in the broader sense) may be involved in the whole procedure and may adjust their behavior based on the guidelines received from the conflict mitigator (CM) module. The latter is particularly interesting when pre-action (proactive) conflict resolution and management are considered. Following the E2 guidance procedure (introduced into the O-RAN specifications in 2023 [ORA23]), the xApp may apply the request/response application programming interface (API) procedure to obtain guidance from the Near-RT RIC prior to any action. By doing this, the xApp will receive guidance with recommendations when the CM module identifies any potential conflicts with other installed xApps; clearly, the recommendations may change as the situation in the network evolves.

Second, the implementation of any algorithm within the xApps has to be flexible enough to reflect the recommendations provided through the E2 Guidance procedure. So, the algorithm has to not only react to the situation observed in the network, the operator's goals, and requirements, but it should also be adjustable to the recommendations that originate by the instantaneous configuration of the xApps installed in the RIC by the operator. Such observation poses some challenges to the algorithm design principles especially if the operator's goal and SLA have to be fulfilled as well. These may call for the need for adaptive solutions, where some AI tools are applied.

Finally, a particular challenge from the xApp provider perspective is the need for unified procedures, protocols, interfaces, and messages that may be used for conflict management. To address this, there should be APIs defined over standardized interfaces dedicated to conflict management such as E2 guidance. Additionally, xApps should be compatible with the standardized conflict management procedures developed in O-RAN. If different RIC instantiations do not adhere to consistent procedures for delivering conflict mitigation functionality, it will be very challenging for the xApps to follow the CM module guidance. Therefore, the RIC platform should support conflict management functionality and standardized API procedures/messages.

A separate perspective is that applications could follow a template-based approach, where the application is defined in detail using the meta-data file with specific descriptors. The template may contain the list of requested (inputs) and modified (output) parameters by the application. Such an approach may simplify the conflict detection and resolution procedure, as discussed in [KDR$^+$23]. However, such an approach requires the standardization of the template that will be used to describe the xApp characteristics and behavior.

3.2 Background

When evaluating conflict mitigation strategies in wireless networks, it is essential to consider both perspectives from network operators and xApp developers. However, this topic has been also extensively explored in the literature, underscoring its importance and complexity. In the literature, conflict mitigation in wireless networks has been extensively discussed as a critical challenge. A systematic analysis of the conflict mitigation challenge is typically addressed through a combination of game theory, access control, and coordination mechanisms [CCBG07], [TCM00]. Conflict mitigation in RAN control has been extensively explored within the context of SON. SON functions can be implemented in both centralized and distributed architectures. Centralized SON relies on a central controller to manage network optimization tasks, while distributed SON allows individual network elements to perform these tasks autonomously. SON functions are typically designed to optimize specific aspects of the network, such as Mobility Robustness Optimization (MRO) and Mobility Load Balancing (MLB), within a relatively constrained scope of control [moy, JPG$^+$14]. SON primarily deals with direct conflicts, where multiple SON functions attempt to modify the same network parameters simultaneously. Conflict detection in SON often involves monitoring these parameters and implementing coordination mechanisms to prevent conflicts. The 3GPP standards provide guidelines for SON conflict detection and resolution, focusing on predefined scenarios and optimization tasks. Recent research has introduced machine learning frameworks for SON conflict mitigation, allowing the network to learn from past experiences and optimize future decisions. For instance, Moysen et al. [moy] and Jorguseski et al. [JPG$^+$14] propose a generalized machine learning framework for SON functions such as MRO and MLB, aiming to optimize handover performance and load balancing. Additionally, Mwanje et al. [MSMT16] introduce a SON coordination scheme based on a reinforcement learning approach, allowing the network to learn from previous experiences to enhance future network decisions. Traditionally, SON implementations are often provided by a single vendor, which simplifies interoperability but limits flexibility and vendor diversity. While there are efforts to standardize SON functions, the lack of a multi-vendor ecosystem means that interoperability challenges are less pronounced compared to O-RAN.

O-RAN features a disaggregated architecture where the RAN components are decoupled and managed by RICs. This architecture allows for greater flexibility and interoperability among multi-vendor components. O-RAN must manage not only direct conflicts but also indirect and implicit conflicts. Indirect conflicts occur when actions by one xApp affect parameters controlled by another xApp, while implicit conflicts arise from complex interactions that are not immediately apparent. O-RAN employs advanced conflict detection mechanisms, such as monitoring control parameters stored in a catalog or database and using AI/ML techniques to predict potential conflicts. These mechanisms are more sophisticated due to the dynamic and multi-vendor nature of O-RAN.

As a solution tailored for O-RAN, [AK23] introduces a framework for conflict resolution in O-RAN, presenting a general approach that demonstrates improvements in certain network performance indicators at the expense of slight decreases in others. Another approach, detailed in [ZZEK22], proposes a team learning-based strategy to reduce and eliminate conflicts among xApps in the Near-RT RIC. Here, xApps learn to cooperate and prevent conflicts using a Deep Q-Network (DQN) architecture. Furthermore, [IRZZ$^+$22] builds upon this strategy by presenting a case study of multi-agent team learning for xApps controlling various RAN parameters. This approach mitigates conflicts post-deployment. In [YNSSH24], the authors offer an overview of how conflicts can lead to misconfiguration in O-RAN and provide a solution to detect conflicts. In [dPDB$^+$24], the authors introduced a framework aimed at detecting, characterizing, and addressing conflicts within the O-RAN ecosystem through statistical analysis and hierarchical graphs. Meanwhile, [YNSSH24] offers an overview of how conflicts can lead to misconfigurations in O-RAN, highlighting their effects on all layers of the protocol stack, which may result in higher energy consumption, reduced performance, and instability.

O-RAN Alliance Standards: The O-RAN Alliance has introduced the E2 guidance related procedures [ORA23] to address conflicts within the Near-RT RIC. The E2 guidance procedure consists of two sub-services: E2 Guidance Request/Response (xApp initiated) and E2 Guidance Modification (Near-RT RIC initiated). The E2 Guidance procedure enables xApps to obtain preemptive conflict resolution advice from the conflict mitigation function within the Near-RT RIC before issuing an E2 Subscription or E2 Control request. This guidance is crucial for ensuring that xApps can operate harmoniously without causing detrimental interference with each other. The guidance provided can include indications of potential conflicts, recommendations for modifying proposed E2 API messages to avoid conflicts, and updates to previous guidance issued to other xApps or platform functions.

The E2 Guidance procedure promotes more effective and coordinated network management by allowing xApps to query the conflict mitigation component within the Near-RT RIC before initiating actions. It helps in identifying and resolving conflicts early, thereby reducing the likelihood of performance degradation or resource contention. The guidance request from the xApps (containing, e.g., its planned control actions, resource consumption, and KPIs in the RAN), can be evaluated using internal databases/records, data analysis, and side information, amongst other sources/functionalities, to assess whether conflicts with other xApps may arise. The guidance procedure is initiated by an xApp sending an E2 Guidance request via the API, and the Near-RT RIC responds with appropriate E2 Guidance Modifion. This process can also result in modified guidance being issued to other xApps or platform functions to resolve conflicts and ensure overall network harmony.

In summary, the E2 Guidance procedure and the associated standards play a pivotal role in managing conflicts within the Near-RT RIC, ensuring that xApps can function without causing adverse effects on each other. This framework is vital for maintaining the stability and performance of the network, especially in a multi-vendor, O-RAN environment where interoperability and coordinated management are key challenges. Given the above state-of-art, one can envisage the standardization requirements that can be anticipated in relation to the preceding discussion.

3.3 Conflict Detection Approaches

In the realm of conflict management, detecting conflicts is the crucial first step. Accurate detection lays the groundwork for effective resolution and mitigation strategies. In the following discussion, we will explore various practical approaches for detecting conflicts, emphasizing methods that have proven to be both reliable and efficient. Understanding these approaches is key to anticipating issues before they escalate, ensuring smoother operations and better overall outcomes.

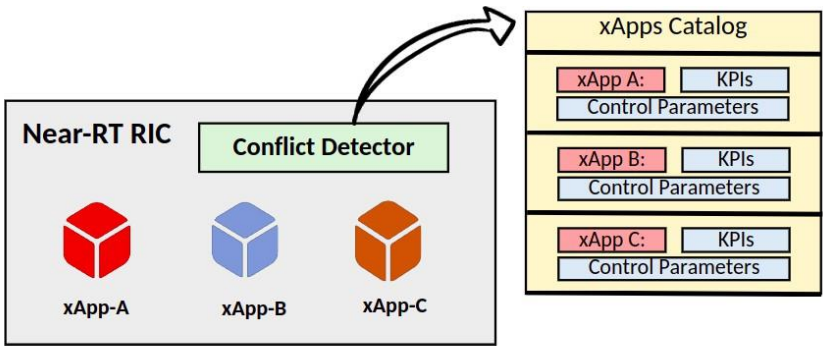

Figure 4: Conflict detection using catalog of xApps

One approach for conflict detection is to implement a dedicated conflict detector (CD) component within the Near-RT RIC. This module would utilize a catalog or database containing detailed information about the control parameters and KPIs targeted by each xApp installed on the Near-RT RIC. By referencing this catalog, the CD can identify potential conflicts more effectively. As illustrated in Fig. 4, such a catalog would also serve as a valuable resource for developers, helping them design xApps in a way that minimizes potential conflicts. Direct conflicts, which occur when multiple xApps attempt to modify the same control parameters, can be detected relatively easily. In such cases, by leveraging the catalog, the CD can identify if any control action shares the same control parameters with the running xApps, thereby detecting a direct conflict efficiently. Unlike direct conflicts, indirect and implicit conflicts cannot be directly identified. Indirect conflicts can also be detected using the catalog by observing how one xApp's control actions influence the control parameters of other xApps. These control parameters can be monitored by the KPM xApps and verified against the catalog by the CD to detect indirect conflicts. Similar to indirect conflicts, implicit conflicts can also be detected by analyzing the KPIs influenced by each xApp installed on the Near-RT RIC, with the CD declaring an implicit conflict when degradation exceeds predefined thresholds. This approach for conflict detection, i.e., utilizing the comprehensive catalog, enables efficient identification of all types of conflicts, thereby enhancing overall network performance and stability in the O-RAN environment. A framework for advanced conflict detection in Near-RT RIC is presented in [Eng22].

This framework employs machine learning algorithms to anticipate potential conflicts and alert xApps and rApps, facilitating the mitigation of adverse interactions. It provides a robust conflict detection mechanism by ensuring effective conflict resolution and coordination between different xApps. The architecture detailed in [Eng22] is designed with layered abstraction to handle these complexities, offering seamless integration and prioritization rules for identifying and addressing conflicts efficiently. Additionally, the conflict manager module [AD22, AY23] plays a crucial role in detecting potentially overlapping or conflicting RAN control decisions from multiple xApps and in notifying them accordingly. For instance, suppose xApp-A aims to transfer users from one cell to another because of high traffic load, while concurrently, xApp-B intends to move users back as the handover boundary shifts due to a high rate of handover failures. Without a conflict mitigation function overseeing the actions of the xApps and monitoring network performance, users might experience frequent "ping-ponging" between cells. In this scenario, the role of the conflict mitigation function is to coordinate and synchronize the actions of the xApps to prevent such undesirable outcomes.

The conflict manager's guidance or response should indicate whether the proposed E2related API message or series of messages from an xApp might conflict with E2-related API messages from other xApps (This procedure is optional for the xApp). Figure 6 illustrates the workflow of the conflict manager, which involves the following steps:

- Conflict manager maintains a list of resources (UEs, Cells, Slices, etc.) received in RIC CONTROL REQ and RIC E2 GUIDANCE REQ.
- xApp may check with the conflict manager before sending a control request (optional).
- New RMR message types RIC E2 GUIDANCE REQ, RIC E2 GUIDANCE RESP
- shall be defined.
- All control messages from xApps shall be forwarded to the conflict manager from the E2 Terminator.
- Conflict manager will be adding state variables (for each parameter) to each UE, cell, and slice.
- State variables are updated based on RIC CONTROL REQ and RIC E2 GUIDANCE REQ.
- The current status of state variables shall determine the conflicting request.

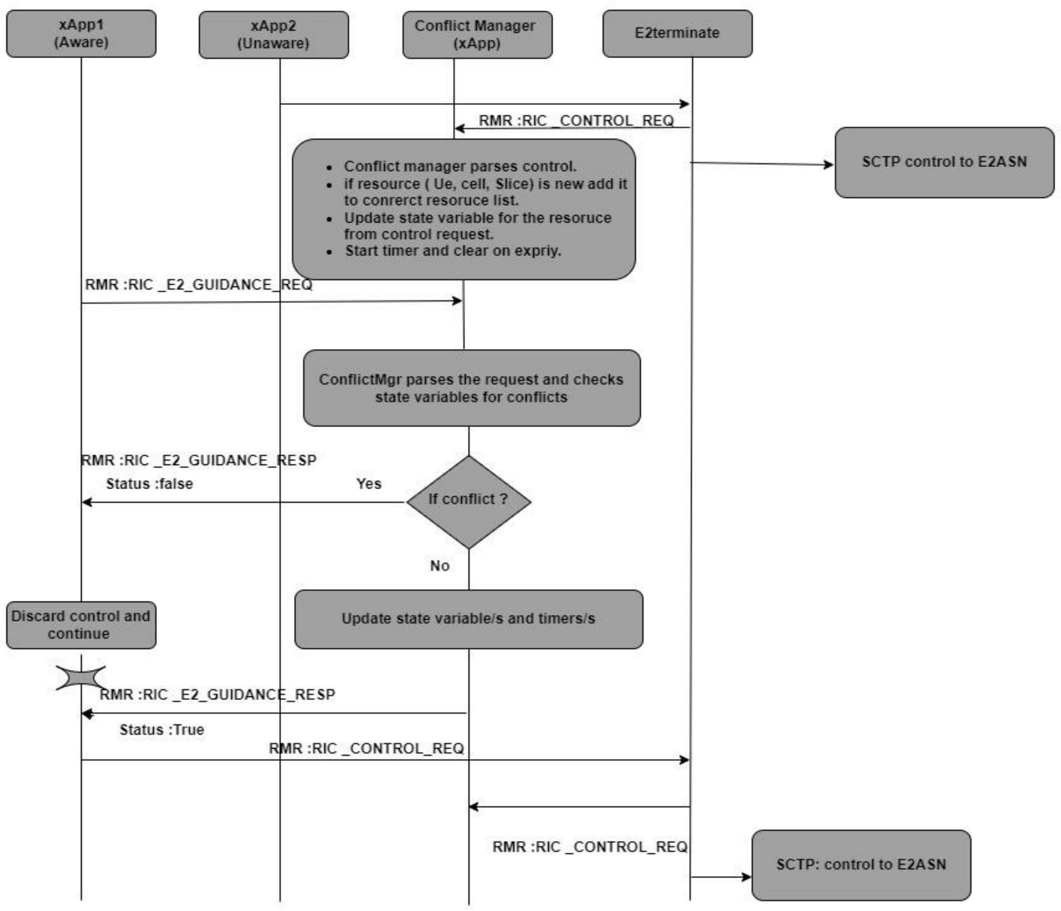
xApp1 (Aware)
xApp2 (Unaware)
Conflict Manager (xApp)
E2terminate
RMR :RIC _CONTROL_REQ
SCTP control to E2ASN
Conflict manager parses control.
if resource (Ue, cell, Slice) is new add it to conrerct resoruce list.
Update state variable for the resoruce from control request.
Start timer and clear on expriy.
RMR :RIC _E2_GUIDANCE_REQ
ConflictMgr parses the request and checks state variables for conflicts
RMR :RIC _E2_GUIDANCE_RESP
Status :false
Yes
If conflict ?
No
Discard control and continue
Update state variable/s and timers/s
RMR :RIC _E2_GUIDANCE_RESP
Status :True
RMR :RIC _CONTROL_REQ
RMR :RIC _CONTROL_REQ
SCTP: control to E2ASN

Figure 5: Conflict Manager Workflow

The principles of the theory of causality can also be used for conflict detection in O- RAN. While conflict detection with causal analysis can be useful for all three types of conflicts described in the taxonomy in Section 2.2, it is arguably most useful in dealing with the second type, indirect conflicts, and the third type, implicit conflicts.

Correlation is not Causation: Here, we can only skim the surface of what causal analysis means. For more details, see, e.g., [PJS17], [Pea09], [SGS01]. There are mainly two ways to create the catalogs mentioned earlier. We can (i) use *our understanding* of the involved mechanisms to reason about mutual influences or (ii) infer those influences *from data* that has been collected from the real world or from simulations. Here, the focus is on the second, the data-driven approach. If the xApps and rApps become more numerous, originate from different vendors, and become more complex, it will often be impossible to fully understand how they affect each other. In those cases, we have to resort to detecting inter-app effects from data, i.e., measurements of the relevant parameters. This task will be considered in a principled way in this section.

The first step in detecting those effects from data is the application of statistical / machine learning methods to obtain information about stochastic dependencies between parameters. This is also addressed earlier, which focuses on the *Conflict Manager*. A less precise but more common way of describing this stochastic approach is to say that we look for *correlations*. However, to properly understand how apps affect each other, we have to understand which app is *causing* which change in parameters. Nowadays, everyone is aware of the famous dictum "correlation is not causation". This statement is immediately evident if one realizes that correlation is a symmetric relation while causation is not, i.e., if entity *A* is correlated with entity *B*, then *B* is also correlated with *A*, which is not true for causality.

But the difference runs deeper. Consider, e.g., a parameter describing the workload *W* of an xApp (e.g., the CPU load caused by this xApp) and the energy consumption *E* in the network. Let's presume that the xApp is quite "small" in the sense that changing *W* will only have a negligible effect on *E*, i.e., there is no causal effect from *W* on *E*, which is described in the causal graph of Figure 6 (left) by a missing arrow from *W* to *E*. Also, let's presume that the xApp doesn't care about the energy consumption, so *W* is also not affected by *E*, i.e., there is also no arrow from *E* to *W*. Imagine further that the xApp will have a higher workload if the demand *D* of attached UEs creates more network traffic, meaning there is an arrow going from *D* to *W*. Finally, more traffic will usually also increase the energy consumption *E* in the network, depicted by an arrow from *D* to *E*. This finalizes the construction of the true causal graph in Figure 6 (left), which we want to infer from the data. The decisive point here is that we will measure a positive correlation between the two parameters *W* and *E*. That is even though neither *W* has a causal effect on *E* nor *E* has a causal effect on *W*. It is the third entity, the demand *D* of the UEs, that is driving both *E* and *W*. In this situation, we call *D* a *confounder*. Thus, confounders "fake" connections; they create correlations between components that do not causally affect each other.

Interventions: To summarize so far, machine learning methods only provide us with correlations, and those are not enough to infer the causal relationships, which are essential for the detection of conflicts. But how, then, can we measure causality?

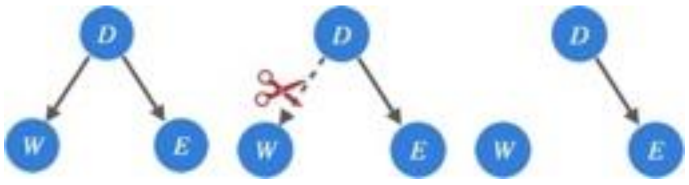

Figure 6: Left: The causal graph for a confounder D of two random variables W and E. Middle: The act of intervention on W to obtain the parameters it causally affects. Right: The new system after intervention.

The standard approach is to apply *interventions*. In causality, performing an intervention means to *change the system into a similar one whose causality we understand better*. The most common type of interventions are *do-interventions*, see [Pea09], which is depicted in Figure 6 (middle and right), for an intervention on *W* . If there were a causal connection from *W* to *E*, it would still be present in the intervened graph. Interventions on *E* work similarly with the same result.

While applying interventions as described above is the most powerful method of detecting causal relationships, it is not the only one. Another approach is to use *soft* interventions, i.e.

different *environments*, rather than the *hard* ones explained above. For more details, see, e.g., [Ebe14] or [RHPM15]. An example of different environments would be the same O-RAN network with the same Apps, but at different times, e.g., day versus night, or different locations, e.g., city versus rural areas.

There are still further approaches, e.g., based on additive noise models [ZH12], information geometric approaches [JMZ+12], or ICA-based methods [SHH+06], but they all share the problem that they don't have the same discerning power as interventions.

Three challenges with causal analysis in mobile networks: Below, we will enumerate three major challenges that are faced when applying causal analysis to O-RAN networks: (i) Interventions in mobile networks tend to be expensive or even impossible, (ii) existence of hidden confounders, and (iii) mobile networks are inherently cyclic.

Interventions are expensive: Using interventions to obtain causal information encounters severe obstacles when applied to mobile networks since changing the system locally in such a way is often too expensive or even outright impossible. One way of dealing with this problem is to develop realistic simulators, see Section 4, which describes the i14y Lab simulator.

Hidden confounders: Above, we have already explained confounders, and in our example, we identified the demand D of UEs as a confounder for the energy consumption E and the workload W of some xApp. If there were no measurements of D, only of W and E, D would be called a *hidden* confounder, see Figure 7 (left). Most causality methods presume that the true underlying causal graph has no hidden confounders. In mobile networks, this is frequently not justified. For example, the rate demand of UEs is rarely known, as is the dynamics of channel characteristics. The purely observational theory of systems with hidden confounders is based on MAG graphs, see [RS02].

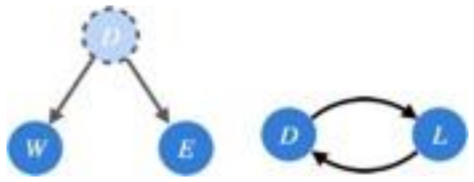

Figure 7: left: a hidden confounder, right: a cycle

Cycles: Another restriction presumed by almost all causal analysis techniques is that the system's causal graph must not have cycles. This, too, is quite a severe restriction. Consider, e.g., in a mobile network the demand D of UEs and the load L of cells. Clearly, the demand drives the load, so the demand causally affects the load. But there are applications running on UEs that can adapt their rate demand to the current network conditions, and reduce the demand if load is high. Thus, the load can also causally affect demand. This is depicted in Figure 8 (right).

Methods allowing for hidden confounders and cycles: A fundamental treatise of those systems can be found in [FM17] and [BFPM21]. Among the few algorithms that remain applicable, we want to mention LLC [HEH12], sigmasep [FM18], [LM23], BackShift [RHPM15], and CCI [Str19].

3.4 Conflict Mitigation Approaches

In developing conflict mitigation strategies, it is crucial to recognize that the conflicting xApps may come from different vendors, and each xApp could have its own potentially conflicting objectives. This situation often results in a reluctance to share extensive information between components from different vendors. Achieving full coordination among xApps can be challenging, particularly when maintaining operational autonomy is a priority. In such environments, traditional conflict mitigation techniques, such as rolling back the actions of one xApp or prioritizing the actions of another, are often employed. While these strategies can be effective to some extent, they can sometimes lead to suboptimal outcomes, as they may not address the underlying causes of conflicts or leverage the full potential of collaborative decision-making. To move beyond these limitations, it is essential to explore more sophisticated techniques that can anticipate and resolve conflicts in a manner that balances autonomy with coordinated action, ultimately enhancing system performance and efficiency.

In O-RAN environments, centralized or coordinated efforts involve utilizing the capabilities of the CM within the Near-RT RIC. The CM serves as a crucial intermediary, resolving conflicts among xApps and ensuring that only conflict-free policies are delivered to the E2 nodes. It is granted extensive access to all xApps and is designed to consult with them prior to executing actions. The core functions of the CM include evaluating proposed actions from each xApp and assessing the associated costs. With an integrated feedback mechanism, the CM could direct xApps to revert actions in the event of a conflict, thereby preventing network disruptions. This continuous exchange of information and feedback ensures effective conflict resolution. By enhancing coordination among xApps, the CM would optimize RAN performance, showcasing the potential of intelligent conflict management in advancing O-RAN capabilities. This proactive strategy aims to ensure more efficient and stable network operations in the complex O-RAN environment.

The level of coordination, driven by information sharing, directly impacts the effectiveness of conflict mitigation. Inadequate coordination often leads to performance degradation, whereas enhanced coordination consistently yields superior outcomes. Optimal results are attained when there is full coordination among all involved entities, underscoring the pivotal role of comprehensive information sharing. However, within the multi-vendor O-RAN ecosystem, achieving extensive information sharing and complete coordination may pose challenges, especially when prioritizing operational autonomy. Hence, alternative strategies are imperative to strike a balance between information sharing, network performance, and operational independence. It is important to note that the CM does not have access to the full action space, preventing it from optimally resolving conflicts. Instead, it relies on shared information from each xApp to provide feedback and calculate costs.

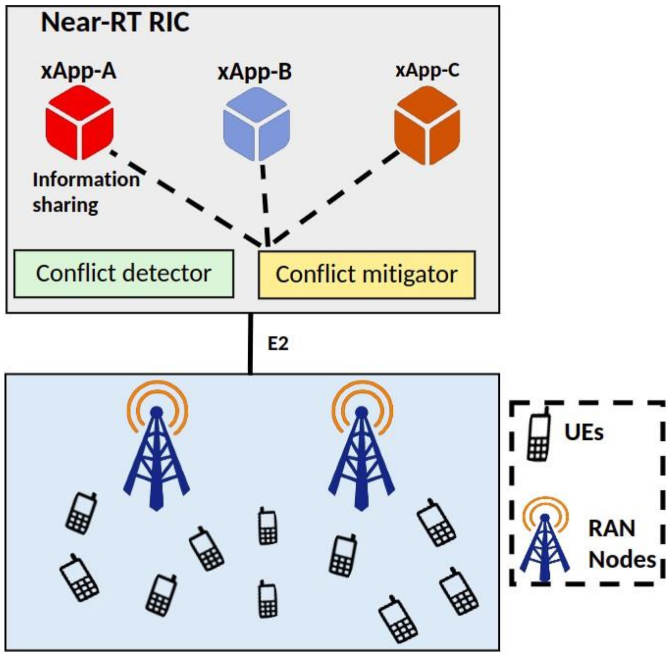

Figure 8: Conflict mitigation via information exchange

The O-RAN standardization aspects for conflict management also follow similar arguments as mentioned earlier. In general, conflict mitigation functionality in Near-RT RIC, since it resides inside the platform, is a suitable centralized agent for performing at least simple conflict detection and management/avoidance. For example, the platform can always block xApp subscriptions or merge them in various ways to align xApp objectives, assuming that it knows the objectives. However, tasks that require data collection or advanced AI-type mechanisms are better suited to be deployed as xApps. Multiple vendors can then compete for such functionalities in the form of xApps and test them in a standardized environment. For advanced coordination solutions, the O-RAN standard may also standardize the information that xApps must share/expose to the CM. Of course, this information may not include vendor business intelligence (i.e., why an xApp chose a certain action or how it derives its objective from the KPIs) but it may include standardized control parameters, use-cases, control actions, and KPIs that an xApp considers relevant. Alternatively, xApps can communicate directly via inter-xApp APIs to coordinate their actions. Such APIs, however, do not yet exist so for now any communication would be indirect through the platform. As mentioned above, standardized access to a digital twin or simulator may significantly enhance conflict detection and management/avoidance of conflict mitigation functionality. The interaction between the platform and such tools may be subject to standardization.

4 i14y Lab Simulator as a Tool for Conflict Analysis

Using a simulator as a digital twin offers a transformative approach to conflict analysis and detection within dynamic systems. By replicating real-world scenarios in a virtual environment, the simulator provides a platform for observing and analyzing the behavior of system components and entities. The i14y Lab simulator is a digital twin for O-RAN systems and it can simulate complex network scenarios and evaluate the performance of various networking protocols and algorithms. The i14y Lab simulator is an ns-3-based simulator, providing a modular architecture, and offering sophisticated models to replicate complex network entities. Through sophisticated modeling techniques, complex interactions can be simulated, allowing for the identification of potential conflicts and their underlying causes before they manifest in reality. The i14y Lab simulator is integrated with the OSC Near-RT RIC [O-R24] and offers an end-to-end system to test the functionality of the xApps. Using the Near-RT RIC in conjunction with the simulator, multiple conflicting xApps can interact with the RAN enabling monitoring and analysis capabilities for the detection of emerging conflicts and facilitating proactive interventions to prevent or resolve them. Moreover, the simulator offers several models (energy, load, etc.) to monitor KPIs, in order to identify potential conflicts. The simulator also allows the analysis of the effects of conflicting xApps in terms of network scalability and solutions.

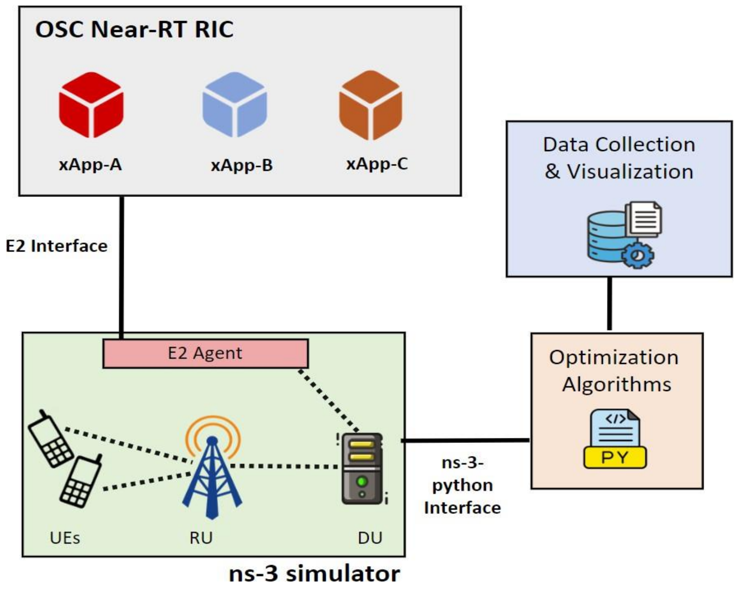

Figure 9: i14y Lab simulator architecture

The i14y Lab simulator provides several remarkable advantages when it comes to examining multiple scenarios. First of all, it can, in principle, generate any amount of data. In particular, it can generate data for corner cases that are so rare that it is unlikely that we can observe and measure them in practice. Second, in the analysis of causal effects like in A/B tests, we can obtain the ground truth of the counterfactuals by running the simulation twice with the same network parameters and simulation initialization. Third, the i14y Lab simulator allows us to do interventions. In complex settings, it is often impossible to understand or measure the causal relationships in the system without interventions, i.e., changing the system. While this is usually impossible in practice, a digital twin enables us to do exactly that, which provides us with the most powerful techniques in causal discovery and analysis. Along with this the simulator also supports the ns3-ai library designed to facilitate the interaction between the ns-3 network simulator and Python. This integration enables the application of data analytics and machine learning algorithms to enhance conflict detection capabilities, uncovering hidden patterns and trends within the data. By leveraging ns3-ai, users can perform sophisticated data analysis for conflict detection, identifying potential issues more accurately and efficiently. Additionally, this tool allows for the implementation of controlled interventions in the network, providing a robust platform to test and evaluate various conflict mitigation approaches. The flexibility offered by ns3-ai not only supports conflict analysis but also allows them to integrate advanced analytical techniques into their network simulations, ultimately leading to more intelligent and adaptive conflict management solutions.

5 Conclusion

The introduction of intelligence through the O-RAN architecture, specifically via the RICs hosting Apps, significantly enhances network optimization and automation. However, this increased intelligence leads to potential conflicts as multiple applications with diverse objectives and decision-making processes operate simultaneously. Addressing these conflicts reveals a critical gap in current standardization efforts, underscoring the need for comprehensive recommendations. Standardizing interfaces, identifying gaps in existing standards, and implementing necessary architectural changes, including new entities, are essential to ensuring harmonious and efficient network operation.

In this paper, we summarize the perspectives of network operators as well as x/rApp developers, while highlighting the standardized procedures from the O-RAN Alliance related to this topic. From the network operator's perspective, managing these conflicts requires prioritizing operational stability, network security, operational efficiency, and network performance. x/rApp developers face complementary challenges, needing to align with operator policies and SLA requirements while ensuring flexibility and adaptability in their solutions. These developers must not only meet stringent technical requirements but also anticipate evolving network conditions and operator needs, which adds complexity to the development process. Proactive conflict resolution and adopting a template-based approach to define application characteristics can further simplify conflict detection and resolution.

Overall, collaborative standardization and adaptable solutions from both operators and x/rApp developers are crucial for effective and efficient network optimization in the O-RAN environment. In this paper, we also examine the standardized procedures established by the O-RAN Alliance to support these goals and facilitate smoother integration and cooperation between stakeholders. Clear guidelines and procedures are necessary to ensure that all parties work within a common framework, reducing the potential for conflicts and enhancing overall network performance. This work also discusses different frameworks for both conflict detection and mitigation in O-RAN's Near-RT RIC. We expect future work in this direction to have more sophisticated and smart components, potentially with the use of AI/ML-based tools for detection and mitigation of conflicts. These advanced tools can offer predictive capabilities, allowing for preemptive identification of potential conflicts before they escalate, which will be key in maintaining the agility and efficiency of future networks.

6 References

[AD22] Capgemini Arnab Das. OPEN RAN Needs To Automate – and Fast. https://www.capgemini.com/insights/expert-perspectives/open-ran-needs-to-automate-and-fast/, 2022. [Online; accessed 12-July-2024].

[AK] Cezary Adamczyk and Adrian Kliks. In *IEEE INFOCOM 2023 - IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*.

[AK23] Cezary Adamczyk and Adrian Kliks. Conflict Mitigation Framework and Conflict Detection in O-RAN near-RT RIC. *IEEE Communications Magazine*, 2023.

[AY23] Capgemini Ashish. Yadav. RAN Intelligent Controller – The Innocation Engine in O-RAN. https://www. capgemini.com/us-en/insights/expert-perspectives/ran-intelligent-controller-the-innovation-engine-in-o-ran/, 2023. [Online; accessed 12-July-2024].

[BdPDB+24] Pietro Brach del Prever, Salvatore D'Oro, Leonardo Bonati, Michele Polese, Maria Tsampazi, Heiko Lehmann, and Tommaso Melodia. PACIFISTA: Conflict Evaluation and Management in Open RAN. *arXiv:2405.04395 [cs.NI]*, pages 1–12, May 2024.

[BFPM21] Stephan Bongers, Patrick Forré, Jonas Peters, and Joris M Mooij. Foundations of Structural Causal Models with Cycles and Latent Variables. *The Annals of Statistics*, 49(5):2885–2915, 2021.

[CCBG07] Frédéric Cuppens, Nora Cuppens-Boulahia, and Meriam Ben Ghorbel. High-level Conflict Management Strategies in Advanced Access Control Models. *Electronic Notes in Theoretical Computer Science*, 186:3–26, 2007.

[dPDB+24] Pietro Brach del Prever, Salvatore D'Oro, Leonardo Bonati, Michele Polese, Maria Tsampazi, Heiko Lehmann, and Tommaso Melodia. Pacifista: Conflict evaluation and management in open ran. *arXiv preprint arXiv:2405.04395*, 2024.

[Ebe14] Frederick Eberhardt. Direct Causes and the Trouble with Soft Interventions. Erkenntnis, 79:755–777, 2014.

[Eng22] Capgemini Engineering. RATIO: CAPGEMINI RIC For Intelligent Open RAN Operations. Technical report, Capgemini Engineering, Feb 2022.

[FM17] Patrick Forré and Joris M Mooij. Markov Properties for Graphical Models with Cycles and Latent Variables. *arXiv preprint arXiv:1710.08775*, 2017.

[FM18] Patrick Forré and Joris M Mooij. Constraint-based Causal Discovery for non-linear Structural Causal Models with Cycles and Latent Confounders. *arXiv preprint arXiv:1807.03024*, 2018.

[HEH12] Antti Hyttinen, Frederick Eberhardt, and Patrik O Hoyer. Learning Linear Cyclic Causal Models with Latent Variables. *The Journal of Machine Learning Research*, 13(1):3387–3439, 2012.

[IRZZ+22] Pedro Enrique Iturria-Rivera, Han Zhang, Hao Zhou, Shahram Mollahasani, and Melike Erol-Kantarci. Multi-agent Team Learning in Virtualized Open Radio Access Network (O-RAN). *Sensors*, 22(14):5375, 2022.

[JMZ+12] Dominik Janzing, Joris Mooij, Kun Zhang, Jan Lemeire, Jakob Zscheischler, Povilas Daniušis, Bastian Steudel, and Bernhard Schölkopf. Informationgeometric Approach to Inferring Causal Directions. *Artificial Intelligence*, 182:1–31, 2012.

[JPG+14] Ljupco Jorguseski, Adrian Pais, Fredrik Gunnarsson, Angelo Centonza, and Colin Willcock. Self-organizing Networks in 3GPP: Standardization and Future Trends. *IEEE Communications Magazine*, 52(12):28–34, 2014.

[KDR+23] Adrian Kliks, Marcin Dryjanski, Vishnu Ram, Leon Wong, and Paul Harvey. Towards Autonomous Open Radio Access Networks. *ITU Journal on Future and Evolving Technologies*, 4(2):251–268, 2023.

[LM23] Boris Lorbeer and Mustafa Mohsen. Comparative Study of Causal Discovery Methods for Cyclic Models with Hidden Confounders. In *2023 IEEE 5th International Conference on Cognitive Machine Intelligence (CogMI)*, pages 103–111. IEEE, 2023.

[moy] Conflict resolution in mobile networks: A self-coordination framework based on non-dominated solutions and machine learning for data analytics [application notes].

[MSMT16] Stephen S Mwanje, Lars Christoph Schmelz, and Andreas Mitschele-Thiel. Cognitive Cellular Networks: A Q-learning Framework for Self-organizing Networks. *IEEE Transactions on Network and Service Management*, 13(1):85–98, 2016.

[O-R] O-RAN Alliance. O-ran alliance. https://www.o-ran.org/. Accessed: 2024-08-26.

[O-R24] O-RAN SC. OSC Near Realtime RIC. https://wiki.o-ran-sc.org/display/RICP/2022-05-24+Release+F, 2024. Accessed: 04-Aug-2024.

[ORA23] O-RAN.WG3.RICARCH-R003-v05.00 O-RAN ALLIANCE. O-RAN Work Group 3 (Near-Real-time RAN Intelligent Controller and E2 Interface) Near- RT RIC Architecture . https://oranalliance.atlassian.net/wiki/

spaces/NearRTRIC/pages/2219999444/Approved+Specifications, 2023.

[Pea09] Judea Pearl. *Causality*. Cambridge university press, 2009.

[PJS17] Jonas Peters, Dominik Janzing, and Bernhard Schölkopf. *Elements of Causal Inference: Foundations and Learning Algorithms*. The MIT Press, 2017.

[RHPM15] Dominik Rothenhäusler, Christina Heinze, Jonas Peters, and Nicolai Meinshausen. BACKSHIFT: Learning Causal Cyclic Graphs from Unknown Shift Interventions. *advances in neural information processing systems*, 28, 2015.

[RS02] Thomas Richardson and Peter Spirtes. Ancestral Graph Markov Models. *The Annals of Statistics*, 30(4):962–1030, 2002.

[SGS01] Peter Spirtes, Clark Glymour, and Richard Scheines. *Causation, Prediction, and Search*. MIT press, 2001.

[SHH$^+$06] Shohei Shimizu, Patrik O Hoyer, Aapo Hyvärinen, Antti Kerminen, and Michael Jordan. A Linear non-Gaussian Acyclic Model for Causal Discovery. *Journal of Machine Learning Research*, 7(10), 2006.

[Str19] Eric V Strobl. A Constraint-based Slgorithm for Causal Discovery with Cycles, Latent Variables and Delection Bias. *International Journal of Data Science and Analytics*, 8:33–56, 2019.

[TCM00] Catherine Tessier, Laurent Chaudron, and Heinz-Jürgen Müller. *Conflicting Agents: Conflict Management in Multi-agent Systems*, volume 1. Springer Science & Business Media, 2000.

[YNSSH24] Noe M Yungaicela-Naula, Vishal Sharma, and Sandra Scott-Hayward. Misconfiguration in O-RAN: Analysis of the Impact of AI/ML. *Computer Networks*, page 110455, 2024.

[ZH12] Kun Zhang and Aapo Hyvärinen. On the Identifiability of the Post-nonlinear Causal Model. *arXiv preprint arXiv:1205.2599*, 2012.

[ZZEK22] Han Zhang, Hao Zhou, and Melike Erol-Kantarci. Team Learning-based Resource Allocation for Open Radio Access Network (O-RAN). In *ICC 2022-IEEE International Conference on Communications*, pages 4938–4943. IEEE, 2022.

About i14y Lab

The i14y Lab is an open lab for interoperability testing of disaggregated telco systems, such as Open RAN, led by Deutsche Telekom together with the consortium partners BISDN, Capgemini engineering, EANTC, Fraunhofer HHI, highstreet technologies, NOKIA, Rohde & Schwarz, Telefonica, TU Berlin, Vodafone, and supported with public funding from the German Ministry of Digital and Transport (BMDV).

The i14y Lab provides infrastructure for integration tests with the aim to evaluate market readiness and accelerate production readiness of multi-vendor disaggregated telco solutions. By creating and providing a vendor-independent environment, we promote the development of an innovative, open, and interoperable telco ecosystem. For more information, go to www.i14y-lab.com.

Funded by:

Consortium Partners

Funded by:
Federal Ministry for Digital and Transport
on the basis of a decision by the German Bundestag

Associated Partners

Supporting Partners

Contact

info@i14y-lab.com | www.i14y-lab.com