On Lazy Soundness of Robustness Diagram with Loop and Time Controls

Mar Elden C. Afable*[1] and Jasmine Malinao[1]

Division of Natural Sciences and Mathematics, University of the Philippines Tacloban College, Tacloban City, Leyte, Philippines,
mcafable1@up.edu.ph*, jamalinao1@up.edu.ph

Abstract. This study introduces and formalizes lazy soundness in Robustness Diagrams with Loop and Time controls (RDLTs), focusing on systems with parallel activities. Lazy soundness is a relaxation of classical soundness that tolerates the non-completion of certain concurrent activities, provided that exactly one valid execution path leads to system completion. To support this, the study defines Complete Activity Structures (CAS) from Maximal Activity Structures (MAS) and proposes structural conditions and algorithms to verify lazy soundness. The Lazy RDLT Soundness Verification Algorithm (LRSVA) is presented along with its time and space complexity analysis. A modified complaint processing system is used to illustrate the framework, where the algorithm is applied under conditions of limited shared resources. Overall, the work advances theoretical insights into concurrent system verification and paves the way for future research in automated and soundness checking for parallel workflows.

Keywords: Workflows, Robustness Diagram with Loop and Time Controls, RDLT Lazy Soundness, Structural Profiles, Complaint Processing System

1 Introduction

Workflows are automated processes guided by rules, where tasks are passed between participants [1]. Workflow models formally characterize the structural composition of processes, the associated data flows, the participating agents, and the governing execution semantics, in accordance with specific user and application requirements [2]. As systems have grown more complex, various workflow models have been developed to represent real-world processes, each capturing at least one of the three workflow dimensions: resource, process, and case.

Among these workflows is the Robustness Diagram with Loop and Time Controls (RDLT) which can represent information in these three dimensions. Along with this, components of RDLT have reset functionalities that provide flexibility in representing various systems and models.

J. Caro et al. (eds.), *Proceedings of the Workshop on Computation: Theory and Practice (WCTP 2025)*, Atlantis Highlights in Computer Sciences 24,
https://doi.org/10.2991/978-94-6239-638-8_16

Robustness Diagram with Loop and Time Controls

The RDLT, introduced in [3], is a graph-based and multidimensional workflow formalism designed to model complex systems through a structured graphical representation. Formally, an RDLT is defined as a quadruple

$$R = (V, E, T, M),$$

where V denotes a finite set of typed vertices, E a constrained set of directed arcs with associated attributes, T a traversal-time mapping over arcs, and M a marker function that identifies specialized subsystems within the model.

A distinguishing feature of the RDLT is the incorporation of reset-bound subsystems (RBS), which are induced through the marking function M. When activated, a marked vertex serves as the center of a bounded, resettable subgraph. Figure 1 illustrates an RDLT example, including an RBS centered at vertex x_2 with its associated controllers y_4, y_5, and y_6. Arcs (y_1, x_2) and (y_2, x_2) are in-bridges, while arcs (y_5, y_7) and (y_6, y_7) are out-bridges of the RBS. These arcs are type-alike, as they serve as bridges of the same subsystem.

JOIN structures in RDLTs involve JOINS and SPLITS, which are also observed in the sample RDLT. Vertex y_3 is classified as an AND-JOIN, while the RBS center x_2 represents a MIX-JOIN [3]. Another type of JOIN, the OR-JOIN [3], is characterized by a vertex whose incoming arcs share the same C-values, though it is not present in this example.

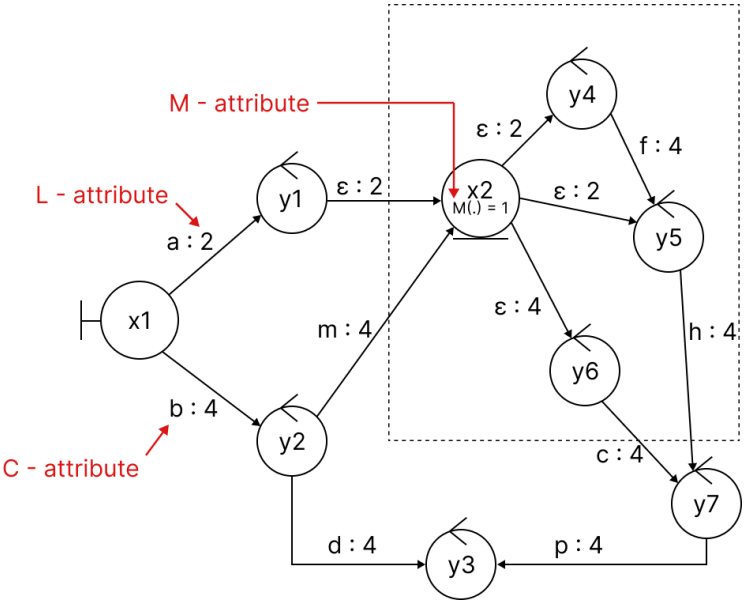


Fig. 1: Labeled RDLT [4]

1.1 Parallel Activities and their Extraction in RDLTs

Definition 1. ***Reachability Configuration*** *[3]*
A ***reachability configuration*** $S(t)$ *in* R *contains the arcs traversed at time step* $t \in \mathbb{N}$

Definition 2. ***Activity Profile*** *[3]*
We call a set $S = S(1), S(2), ..., S(d)$ *of reachability configurations,* $d \in \mathbb{N}$*, as an activity profile in* R *where* $\exists(u, v) \in S(1)$ *and* $(x, y) \in S(d)$ *such that* $\nexists w, z \in V$ *where* $(w, u), (y, z) \in E$*.*

Definition 3. ***Unconstrained Arc*** *[3]*
An arc $(x, y) \in E$ *is unconstrained if* $\forall(v, y) \in E$*, where* (x, y) *and* (v, y) *are type-alike, any of the following traversal conditions hold,*

1. $C((v, y)) \in \{\epsilon, C((x, y))\}$*, i.e. the condition of* (v, y) *is the same as* (x, y)*, or no condition imposed by* (v, y) *at all,*
2. $|\{t_i \in T((x, y))|t_i \geq 1\}| \leq |\{t_j \in T((v, y))|t_j \geq 1\}| \leq L((v, y))$*, i.e. the number of checks and traversals done on* (x, y) *has not exceeded that of* (v, y)*'s,*
3. $C((v, y)) \in \Sigma \wedge C((x, y)) = \epsilon \wedge T(v, y) \neq 0$*, i.e. there is no unsatisfied constraint* $C(v, y)$*. That is,* (v, y) *has already been checked and/or traversed by the algorithm in a prior step.*

In RDLTs, multiple paths can exist between a source and a target vertex, including repeatable paths caused by loops and arc L-values. These paths give rise to different activity profiles. In line with this, *activity groups* were introduced in [5] as an aid for structural analysis. An **activity group** is a set of activities that share the same arcs and/or include additional looping arcs. Within an RDLT R, a **minimal activity** S_{min} [5] in an RDLT R is the smallest set of arcs required to perform an activity. Conversely, a **maximal activity** S_{max} [6] is the activity within an activity group that is the superset of arcs among all other activities within the group.

Definition 4. ***Parallel Activities*** *[4]*
A set of maximal activities A of an RDLT R are parallel if for every pair of activities $S, S' \in A$, the following holds:

1. *S and S' have the same set of input and output vertices,*
2. *S and S' do not interrupt each other,*
3. *S and S' are not competing activities,*
4. *they complete at the same time.*

2 Related Literature

2.1 Soundness of RDLTs

The property of soundness of workflow models defines how the model performs regarding deadlocks and anomalies.

Classical soundness establishes a formal correctness criterion for RDLTs under their execution semantics. It ensures that every activity profile representing a feasible execution from a set of source vertices to a designated final vertex satisfies both proper termination and liveness.

In RDLTs, proper termination requires that all activities should reach the target vertex during the termination step, while liveness requires that every vertex in R is included in at least one activity [7]. The relaxed, weak, and easy soundness are defined by weakening the requirements imposed by classical soundness. **Relaxed soundness** is a live RDLT that allows activities to not complete as long as one activity completes and terminates. **Weak soundness** requires proper termination and does not need to be live. An RDLT is considered **easy sound** if and only if the final vertex is reachable from the source vertex, while **lazy soundness** enforces mutual exclusion among concurrent activities, ensuring only one completes. According to [9], in a sequential context, there is no difference between the definitions of an easy and a lazy sound RDLT. In parallel systems, however, lazy soundness imposes stricter conditions by permitting the completion of only a single activity among those executed concurrently, thereby distinguishing itself from easy soundness.

2.2 Techniques to Verify Soundness if RDLTs

The study in [8] presents soundness verification methods of RDLTs, focusing on the workflow's structural profile. These methods are outlined as follows:

1. *Compositional Analysis of RDLTs*
   This method simplifies an RDLT by decomposing it into two vertex-simplified models: R_1, which excludes the reset-bound subsystem (RBS), and R_2, which focuses on the RBS and its components. A graph contraction strategy is then applied to iteratively merge vertices via arc contraction. If the entire RDLT can be reduced to a single vertex, the workflow is guaranteed to be deadlock-free, ensuring liveness and proper termination.
2. *Verification by Model Decomposition*
   Verification by model decomposition converts the RDLT into a different workflow model, such as Petri Nets, which are widely used for verification tasks, then checks the preservation of soundness.
3. *L-Safeness of RDLTs without Resets*
   This technique deals with soundness verification of RDLTs, focusing on loops and JOIN structures of RDLTs that do not include reset-bound subsystems.

3 Methodology

Lazy soundness in workflow nets requires that there should ultimately be only one token that reaches the sink place, a condition that can be verified only when processes run in parallel. Consequently, to formalize the notion of lazy soundness in RDLTs, this study will be based on the notion of lazy soundness in workflow nets.

3.1 Lazy Soundness Notion of RDLTs

Definition 5. ***Lazy Soundness of RDLTs***
An RDLT R is ***Lazy Sound*** *if and only if its set of activities* $S = \{G_1 = \{{A_1}^1, {A_2}^1, \ldots, {A_{m_1}}^1\}, G_2 = \{{A_1}^2, {A_2}^2, \ldots, {A_{m_2}}^2\}, \ldots, G_n = \{{A_1}^n, {A_2}^n, \ldots, {A_{m_n}}^n\}\}$ *from R, where* G_i *is a set of activities that has the same completion time, and every pair* G_k *and* $G_{k'}$ *have activities with different completion time, the following hold:*

1. ***Option to Complete.*** *There is an activity that completes at the sink, i.e. for every* G_i*, there is an activity* ${A_k}^i = \{S(1), \ldots, S(p)\}, 1 \leq p \leq diam(R)$*, where* $diam(R)$ *is the diameter of R, such that there is a path* $P = x_1 x_2 \ldots x_p$ *where* $(x_j, x_{j+1}) \in S(j), (x_{j+1}, x_{j+2}) \in S(j'), j < j', x_1$ *is the source, and* x_p *is the sink in R,* $i = 1, 2, \ldots, p-2$*.*
2. ***Proper Termination (Weakened).*** *Exactly one activity completes at the sink, while other activities, if any, do not, i.e. when* ${A_i}^j \in G_j$ *reaches the sink, every* ${A_r}^s \in G_s, \forall r \in \{1, 2, \ldots, m_n\} \backslash \{i\}, \forall s \in \{1, 2, \ldots, n\} \backslash \{j\}$ *cannot complete.*

The option to complete requirement ensures that the sink is reachable from the source. In addition, to guarantee that only one activity reaches the sink, the model enforces a weakened proper termination condition. Together, these two requirements define a lazy sound RDLT, as shown in (Figure 2).

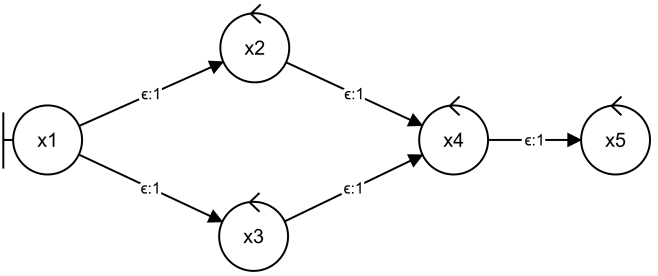


Fig. 2: A Lazy Sound RDLT

Theorem 1 (Parallel Activities of a Lazy Sound RDLT). *Given an RDLT R with one input and output vertices, and its set of maximal activities $A = \{A_1, A_2 \ldots, A_n\} of R, R$ is **Lazy Sound** if and only if:*

1. *A does not contain parallel activities.*
2. *$\forall A_i$, $A \in A_i$ impedes $B \in A_j$, $\forall B, \forall A_j$*

Proof. Assume, for contradiction, that R is Lazy Sound, but at least one of the two conditions does not hold.

For item 1, suppose that A contains at least one pair of parallel activities A_i and A_j, with $i \neq j$.

By definition of lazy soundness, there must only be one activity that will eventually be able to complete execution. However, since there exists activities A_i and A_j that are parallel with each other, then by definition, A_i and A_j are not competing activities and complete at the same time-step. The existence of parallel activities means that at least two activities complete simultaneously, which violates the lazy soundness property. This directly contradicts our assumption that R is lazy sound.

For item 2, suppose that there exist two different activities A_i and A_j such that there is no impediment between any activity $A \in A_i$ and any $B \in A_j$.

If neither activity impedes the other, then both are able to execute and complete without competition. This means that both maximal activities could reach completion. This violates lazy soundness which requires that only one maximal activity completes while all others are blocked or remained constrained. The existence of maximal activities that do not impede each other mean that multiple activities could complete, which contradicts the lazy soundness property.

Since both violations lead to a contradiction, our initial assumption must be false. Therefore, for an RDLT to be lazy sound:

1. It must not contain parallel activities.
2. Every maximal activity must impede every other maximal activity.

The first condition ensures there are no parallel activities among the maximal activities of the RDLT, so only one can reach the sink at a time, satisfying weakened proper termination. However, non-parallel activities completing at different times may still violate lazy soundness. To prevent this, the second condition requires that all maximal activities impede each other, ensuring only one can complete. This behavior is essential for verifying lazy soundness and is defined in (Definition 6).

Definition 6. ***Generalized Impedance***
An input RDLT R is said to exhibit ***generalized impedance*** *if the set of maximal activities originating from the source and terminating at the sink impede each other.*

The key distinction of **generalized impedance** is, instead of examining maximal activities in pairs, as is done in identifying if an RDLT is impedance-free, it evaluates each maximal activity against the entire set of maximal activities. This holistic approach ensures that lazy soundness is preserved in the RDLT. Consequently, an RDLT that exhibits generalized impedance is inherently non-separable and not impedance-free.

3.2 Structural Profiles of a Lazy Sound RDLT

This section introduces the structural profiles of a lazy sound RDLT.

Theorem 2 (Separability of a Lazy Sound RDLT). *A lazy sound RDLT with more than one derivable maximal activity is not separable.*

Proof. Theorem 3 of [6] states that R is separable if and only if R is impedance-free. An RDLT can only be separable AND impedance-free, OR both false. Consequently, we have proved in (Theorem 1) that all the derivable maximal activities of a lazy sound RDLT impedes each other and thus is not impedance-free.

Therefore, a lazy sound RDLT with more than one derivable maximal activity is not separable.

As mentioned previously, current literature focuses on maximal activities of an RDLT R. Accordingly, Maximal Activity Structures (MAS), the structural representations of maximal activities, are central to various verification methods, particularly those involving parallel execution of activities. However, using MAS to verify lazy soundness becomes limiting for two key reasons:

1. For RDLTs with an RBS, EVSA must first be applied before the extraction of the set of MAS is applied. This results in two distinct sets of MAS, one for the level-1 vertex-simplified RDLT R_1 and another for the level-2 vertex-simplified RDLT R_2; and
2. The L-values associated within an MAS does not directly reflect that of the original RDLT, since an MAS represents a single maximal activity.

In order to broaden the scope of RDLTs and its verification methods, this study proposes a novel concept, a structure that is formed by the combination of an MAS from R_1 and an MAS from R_2, all while preserving the L-values of arcs as they appear in R. This combined structure is called a **Complete Activity Structure (CAS)** and is formally defined as follows:

Definition 7. ***Complete Activity Structure (CAS)***
A ***Complete Activity Structure (CAS)*** *of R is a connected RDLT induced from a set of subgraphs composing an MAS of its R_1 and an MAS of its R_2 where for some vertex x in R_1 and R_2, x has an in- or out-bridge in R.*

Since obtaining the set of MAS of R requires applying EVSA, the process of obtaining the CAS of R involves merging two MAS while restoring the original L-values and maintaining the original path from the source to the sink going through the RBS. Given this, several important considerations regarding the CAS must be noted:

1. The vertices of every CAS are controllers, as they are inherited from their respective MAS.
2. The L-values of arcs found outside the RBS of every CAS are identical to those in the original RDLT R.
3. The L-values of arcs found inside the RBS of every CAS are their respective eRU values, except in cases where the sink is inside the RBS.

4. A maximal activity of R can be found by identifying a subgraph in R that matches the vertices and arcs of one of its CAS.

From this point on, this paper will use the concept of CAS to verify lazy soundness.

Theorem 3 (Structural Profile of an RDLT Exhibiting Generalized Impedance). *An RDLT R exhibits generalized impedance if and only if every induced complete activity structure (CAS) shares a common arc (x, y) with all other complete activity structures, such that $L(x, y) = 1$.*

Proof. Assume that R is an RDLT with no RBS, where every induced complete activity structure shares a common arc (x, y) such that $L(x, y) = 1$. We will show that R exhibits generalized impedance.

Since all CAS share the arc (x, y), every maximal activity will eventually encounter this shared arc. Due to the capacity constraint $L(x, y) = 1$, only one activity can pass through this arc at any given execution.

As each CAS converge on this arc, they are forced to compete for resource. Only one CAS will be able to successfully traverse the arc (x, y) and continue execution. All other activities will be blocked at (x, y), thereby ensuring that only one maximal activity completes.

This behavior satisfies the condition for generalized impedance, as each CAS is effectively impeded by the presence of others through a shared, constrained resource. Therefore, R exhibits generalized impedance.

Algorithm 1: Proposed Algorithm for Verifying Generalized Impedance in an RDLT

```
   Input: Complete Activity Structures (CAS) of R
   Output: True, false otherwise
1: Initialize P ← CAS of R
2: if |P| = 1 then
3: |  return false //should have at least 2 CAS to verify generalized impedance
4: else
5: |  for every CAS p ∈ P do
6: |  |  if p shares a common arc (x, y) and L(x, y) = 1 with all other CAS in
   |  |    P then
7: |  |  |  continue
8: |  |  else
9: |  |  |  return false
10: |  |  |  end if
11: |  end for
12: |  return true
```

For generalized impedance to hold, all maximal activities must traverse a common shared arc with an L-value of 1. Merely requiring each activity to include some shared arc with $L = 1$ is not enough. Shared resources with insufficient restrictions can lead to multiple activities reaching the sink and violating

generalized impedance. Thus, a stricter condition is needed: a shared arc with $L = 1$ must be present in every maximal activity. (Figure 3) illustrates both a valid case and a counterexample.

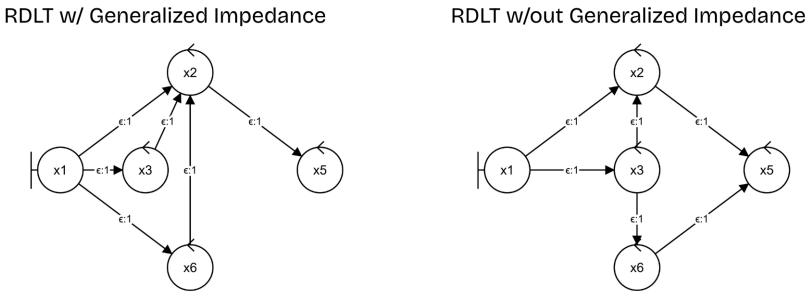


Fig. 3: Illustration of the necessity of the strict Generalized Impedance requirement

Theorem 4 (Structural Profile of a Lazy Sound RDLT with A Single Complete Activity Structure). *An RDLT R is lazy sound if and only if there is exactly one complete activity structure (CAS) derivable from R.*

Proof. Assume that R is classically sound that does not contain parallel activities. By definition, R attains proper termination. Since there are no parallel activities in R, then there is exactly one maximal activity derivable from it. This follows because if multiple activities were able to complete independently without being parallel, R would violate proper termination.

Since there is only one activity derivable from R, there is no other process that is able to reach the final vertex aside from this process, and thus satisfies the requirement of a weakened proper termination that a lazy sound RDLT must satisfy.

Theorem 5 (Structural Profile of a Lazy Sound RDLT with Multiple Complete Activity Structure). *An RDLT R with multiple complete activity structure (CAS) is lazy sound if and only if R exhibits generalized impedance.*

Proof. Follows Theorem 3.

Theorem 6 (Structural Verification of the Lazy Soundness of an RDLT). *An RDLT R is lazy sound if one of the following conditions holds:*

1. *There exists a single derivable Complete Activity Structure (CAS) of R; or*
2. *The set of CAS of R observes generalized impedance.*

Proof. Follows Theorem 4 and Theorem 5.

Algorithm 2: Lazy RDLT Soundness Verification Algorithm (LRSVA)

```
Given an RDLT R that has one source and one sink,
Preprocessing Step: Extract all Complete Activity Structures (CAS) of R.
Input: Set of CAS extracted from R
Output: True, false otherwise
Let 𝒫 ← CAS of R
if |𝒫| = 1 then
    return true //there is only one CAS, lazy sound
else
    for each CAS p ∈ 𝒫 do
        if p shares a common arc (x, y) and L(x, y) = 1 with all other CAS in
          𝒫 then
            continue //continue to next CAS
        else
            return false
        end if
    end for
end if
return true //returns true if manages to iterate through the whole for-loop
```

4 Results and Discussion

4.1 Mapping of Real-world System

This section illustrates a real-world system. (Figure 4) presents an RDLT that models the processing of complaints. This model is a modified version of the complaint process system from the WF-net model of [10], enhanced by introducing cases, additional constraints, and the concept of reusability through the use of an RBS.

The normal flow begins with a complaint being registered and evaluated, while a questionnaire is simultaneously sent to the complainant. If the complainant returns the questionnaire on time and the complaint is deemed valid, the complaint proceeds to processing. After processing, the system checks for success, transitions to the OK state, and finally archives the case. Other possible activity paths can be identified by extracting the set of CAS from the RDLT model, as shown in (Figure 5).

The set of CAS reveals that the system can exhibit four distinct activities:

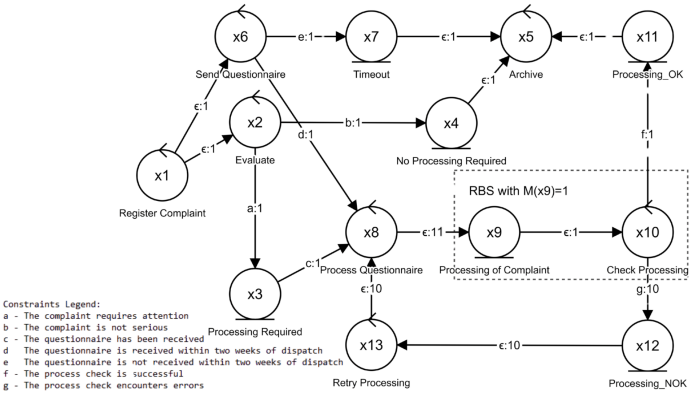


Fig. 4: An RDLT for the Processing of Complaints. Model adopted from [10]

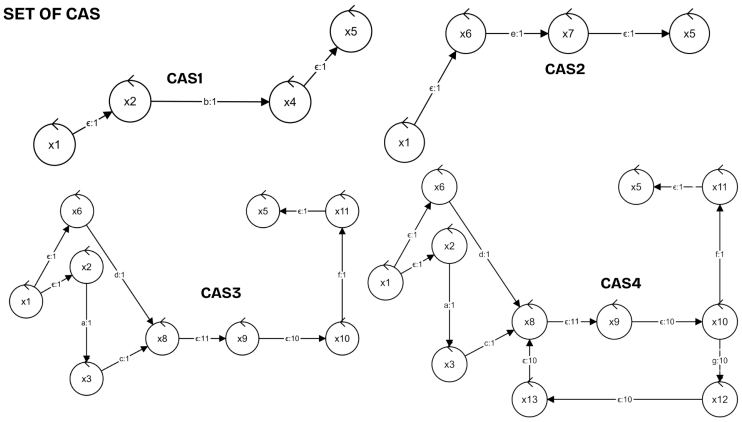


Fig. 5: The Set of CAS Derived from R

1. **Non-serious complaint**: If the complaint is evaluated as not serious, no further processing occurs, and the system directly archives it for record-keeping.
2. **Late questionnaire**: If the questionnaire is not returned on time, the complaint receives a timeout status and is archived.
3. **Successful processing on first attempt**: If the complaint requires attention and the questionnaire is returned on time, the system processes the complaint successfully on the first attempt and then archives it.
4. **Reprocessing required**: Similar to the third, but the initial processing fails. The system enters a reprocessing loop and, after n attempts, eventually passes the process check and archives the complaint.

4.2 Lazy Soundness Verification of the Real-World System

The original RDLT model produced four CAS that failed to meet lazy soundness due to lacking shared arcs. To address this, a modified RDLT (Figure 6a) was created, redefining the sink as x_14. The resulting CAS set (Figure 6b) all share a common arc (x_5, x_14) with an L-value of 1, ensuring that only one activity can complete, satisfying lazy soundness.

4.3 Algorithm Analysis

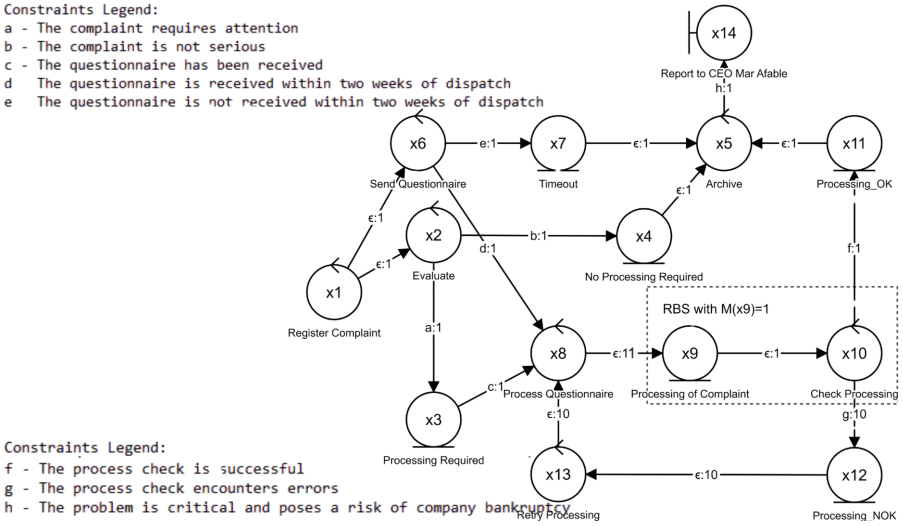
Theorem 7 (Time Complexity of LRSVA). *The time complexity of verifying that an RDLT is lazy sound is $O(|CAS| \cdot |E|)$ where CAS is the complete activity structure of R.*

Proof. The LRSVA requires a preprocessing step of extracting the set of CAS of R. This set will be taken as input and the algorithm consists of executing an additional verification step:

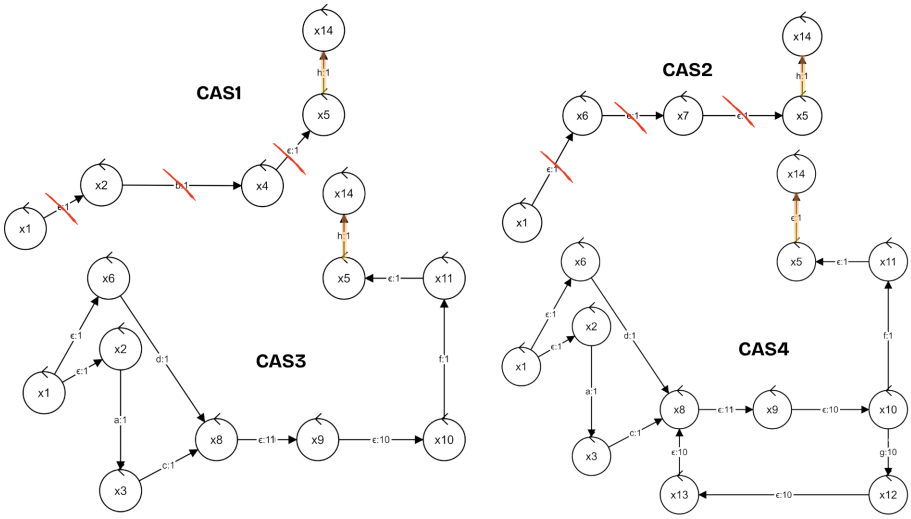
1. **Induced Complete Activity Structures (CAS) Checking**. Unlike ERSVA, where only the existence of a path suffices, LRSVA requires extracting and checking all CAS from the source to the sink. Specifically, this step verifies whether each CAS contains a common shared arc (x, y) with an L-value of 1.
   - Each CAS can have at most $O(|E|)$ arcs.
   - Checking each arc takes $O(1)$.
   - Thus, verifying a single CAS takes at most $O(|E|)$.
   - If there are at most $|CAS|$ substructures, the total complexity for this step is $O(|CAS| \cdot |E|)$.

   The overall time complexity of the LRSVA is $O(CAS| \cdot |E|)$.

Theorem 8 (Space Complexity of LRSVA). *The space complexity of verifying that an RDLT is lazy sound is $O(|CAS| \cdot |E|)$ where CAS is the complete activity structure of R.*



(a) A Modified RDLT for the Processing of Complaints



(b) Application of LRSVA on the set of CAS

Fig. 6: A lazy sound real-world RDLT model and the application of LRSVA

Proof. To analyze the space complexity of LRSVA:

1. **Induced Complete Activity Structure (CAS) Checking**. This process explicitly stores all maximal activity paths. If we assume that at most $|CAS|$ substructures exist, and each substructure contains up to $O(|E|)$ arcs, then the worst-case space complexity for this step is $O(|CAS| \cdot |E|)$.

The overall space complexity of the LRSVA is $O(|CAS| \cdot |E|)$.

5 Conclusion

This paper contributed to the formalization of the notion of lazy soundness in the context of Robustness Diagrams with Loop and Time controls (RDLTs), particularly focusing on systems that exhibit parallel activities. Through the definition of Complete Activity Structures (CAS), generalized impedance, structural theorems, and algorithmic verification strategies, the study introduced a systematic approach to determine lazy soundness. A real-world complaint processing system was modeled to illustrate the applicability of the proposed concepts, showing how shared resources with specific L-values can enforce mutual exclusion among parallel activities.

As the first algorithm specifically designed to verify lazy soundness in RDLTs, this study currently lacks benchmark datasets or prior algorithms for comparison. Once alternative approaches emerge, comparative benchmarking will be crucial for evaluating practical effectiveness. Future research should therefore focus on developing standardized benchmarks to assess the performance and scalability of lazy soundness verification.

Future researchers may also focus on pursuing two possible directions that may stem from this study. First, researchers may develop an automated verification procedure for the lazy soundness of RDLTs and integrate it into the module proposed by [11] and the tool developed by [12]. This integration would utilize the PAE algorithm introduced in [4] to enhance usability and extend the system's verification capabilities. Second, future work may explore extended notions of soundness from the literature, specifically k-soundness, $up - to - k$ soundness, and *generalized* soundness. With parallelism now being considered, formalizing these notions of soundness may represent the next significant step in advancing the field of soundness verification.

References

1. Hollingsworth, D.: Workflow Management Coalition: The Workflow Reference Model. Workflow Management Coalition, 2 Crown Walk, Winchester, Hampshire, UK (1995)
2. Belhajjame, K., Vargas-Solar, G., Collet, C.: A Flexible Workflow Model for Process-Oriented Applications. In: Proc. of the 2nd Int. Conf. on Web Information Systems Engineering, vol. 1, pp. 72–80 (2001)

3. Malinao, J.A.: On Building Multidimensional Workflow Models for Complex Systems Modelling. PhD thesis, Technische Universität Wien, Vienna, Austria (2017)
4. Doñoz, R.B., Malinao, J.A.: Parallel Activities in Robustness Diagram with Loop and Time Controls. Bachelor's Thesis, University of the Philippines Tacloban College, Tacloban City, Philippines (2024)
5. Eclipse, K.P., Malinao, J.A.: Model Decomposition of Robustness Diagram with Loop and Time Controls to Sequence Diagrams. Bachelor's Thesis, University of the Philippines Tacloban College, Tacloban City, Philippines (2024)
6. Malinao, J., Juayong, R.A.: Model Separability of Robustness Diagram with Loop and Time Controls. Science & Engineering Journal, Philippine-American Academy of Science & Engineering (PAASE) (2024)
7. Malinao, J.A., Juayong, R.A.B.: Classical Soundness in Robustness Diagram with Loop and Time Controls. Philippine Journal of Science 152(6B), pp. 2327–2342 (2023)
8. Malinao, J.A., Juayong, R.A.B.: Reset Profiles and Classical Soundness in Robustness Diagrams with Loop and Time Controls. In: Pre-proceedings of the 12th Workshop on Computation: Theory and Practice (WCTP 2023), Chitose City, Japan, pp. 371–394 (2023)
9. Ramirez, R., Malinao, J.A.: On Weak, Lazy, and Easy Soundness in Robustness Diagrams with Loop and Time Controls. Bachelor's Thesis, University of the Philippines Tacloban College, Tacloban City, Philippines (2024)
10. Van der Aalst, W.M.P.: Workflow Verification: Finding Control-flow Errors Using Petri-net-based Techniques. In: Business Process Management: Models, Techniques, and Empirical Studies. Springer, pp. 161–183 (2002)
11. Ejercito, J.A., Malinao, J.A.: An Automated Verification of the Notions of Soundness of Robustness Diagram with Loop and Time Controls. Bachelor's Thesis, University of the Philippines Tacloban College, Tacloban City, Philippines (2025)
12. Labanan, H., Malinao, J.A.: A Modelling and Simulation GUI Tool for Robustness Diagram with Loop and Time Controls. Bachelor's Thesis, University of the Philippines Tacloban College, Tacloban City, Philippines (2025)