On k-Soundness and Maximal Profiles in Robustness Diagrams with Loop and Time Controls

Ronnie Ramirez II [1]*, Richelle Ann Juayong [1], Francis George Cabarle [1], and Jasmine Malinao [2]

[1] Department of Computer Science, University of the Philippines Diliman, Quezon City 1101, Metro Manila, Philippines,
[2] Division of Natural Sciences and Mathematics, University of the Philippines Tacloban College, Tacloban City 6500, Leyte, Philippines
{rmramirez4*, rbjuayong, fccabarle, jamalinao1}@up.edu.ph

Abstract. The Robustness Diagram with Loop and Time Controls is a kind of diagram that encapsulates every workflow dimension. Classical soundness, a workflow property of RDLTs, implies proper termination and liveness in the model if its requirements are met. Other soundness notions can be realized through loosening such requirements. Current literature has solely focused on notions with an inherent sequential context, creating a lack of studies that formalize inherently parallel notions. This study formalizes the k -soundness notion based on the maximal activities of the RDLT. Behavioral and structural profiles of the soundness notion, as well as verification algorithms that ensure the k -soundness of RDLTs, are also discussed.

Keywords: Workflows, Soundness, Model verification, Parallel activities, Robustness Diagram with Loop and Time Controls

1 Introduction

According to the Workflow Management Coalition [7], workflows are used to analyze systems granularly and are created by automating the passing of information between participants. Different models exist that capture the workflow of such systems, as many possible systems exist in real life. One example is the Robustness Diagram with Loop and Time Controls, which has multi-dimensionally modeled different systems, such as the disease surveillance and response system established by the Philippine Department of Health [8]. Also known as RDLTs, one property that they have is classical soundness, which implies proper termination and liveness in all cases [10]. If an RDLT properly terminates, then it has no unfinished processes. On the other hand, a live RDLT guarantees no dead ends, ensuring eventual termination of the system.

Current research has made significant progress in formalizing new RDLT soundness notions, such as classical, relaxed, and weak soundness [10,11,14,16].

© The Author(s) 2026
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_17

However, these notions only describe systems handling one case due to the sequential nature of the activity extraction algorithm of RDLTs. To fill this gap, an algorithm for extracting parallel activities was formulated [5]. This allows property formalization in a parallelized context, including notions for systems handling multiple concurrent cases, such as lazy [4] and k-soundness.

This study formalizes the notion of k-soundness in the context of RDLTs. This entails defining the requirements that the workflow must satisfy. In addition, both behavioral and structural profiles will also be established, including verification methods that ensure the k-soundness of a given input RDLT. Formalizing such notions will create a structured analysis of real-world parallelized systems by ensuring soundness with their RDLT representations, allowing systematic improvements in efficiency and accuracy.

2 Basic Definitions and Notations

2.1 Workflow Nets

A *Workflow Net*, shortened as WF-Net, is a kind of net that highlights the process and case dimensions [9] and is used to check for system correctness [16]. Figure 1 is an example of such a net. It is composed of nodes called *places* and *transitions*, which are connected by *arcs* [1,3]. A place or transition can either be an *input* or an *output*, depending on whether the directed arc is facing away or toward it, respectively. If every input place has a token, the transition connected is *enabled* and *fires*, giving every connected output place a token. WF-Nets differ from classical PNs by having two unique places, each of which has no input or output transitions [2]. These are known as the source i and sink o. Additionally, these nets must be strongly connected, i.e., every node has a path to every other node, if i and o are connected to a transition t, as an output and input place, respectively.

2.2 Soundness Notions of WF-Nets

WF-Nets can be described through properties such as soundness. Classically, this property guarantees that the net has no anomalies that prevent the system from properly terminating, i.e., livelocks and deadlocks [1,3]. With classical soundness, other notions with looser requirements exist, defined in the context of WF-Nets. For relevance, only classical, weak, and k-soundness are defined.

Definition 1 (Classical Soundness of Workflow Nets [1,3]). *A WF-Net is classical sound iff these conditions are satisfied:*

- *Option to complete: the state where other places are empty except for o with one token is reachable from all possible states starting from the state where other places are empty except for i with one token.*
- *Proper termination: when o receives a token, all other places must be empty.*
- *Liveness: every transition t must fire in some traversal.*

Definition 2 (Weak Soundness of Workflow Nets [3]). *A WF-Net is weak sound iff these conditions are satisfied:*

- *Option to complete: the state where other places are empty except for o with one token is reachable from all possible states starting from the state where other places are empty except for i with one token.*
- *Proper termination: when o receives a token, all other places must be empty.*

Weak soundness is looser than classical soundness as liveness is no longer required. Specifically, classical sound WF-nets require that every transition fire, unlike weak sound WF-Nets. Instead, it simply needs a traversable set of nodes and arcs, starting from the initial node to the final node, for every possible state, and no pending processes when a token reaches the sink. Figure 1 satisfies weak soundness, as $t3$ will never fire.

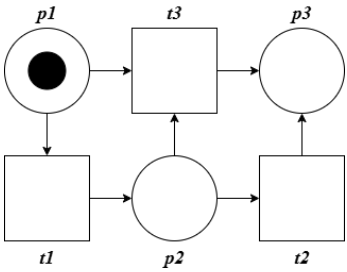

Fig. 1. A weak sound Workflow Net (based on [3]).

These notions are easily describable sequentially, as they require that the input and output places have just one token. Tokens in WF-Nets can be understood as cases in a general sense; thus, it can be said that the entire system only handles one case. Other soundness notions, such as k-soundness as defined below, do not necessarily follow this version of proper termination, as multiple cases, represented by multiple initial tokens in i, are considered.

Definition 3 (k-Soundness of Workflow Nets [3]). *A WF-Net is k-sound iff these conditions are satisfied:*

- *Option to complete: the state where other places are empty except for o with k tokens is reachable from all possible states starting from the state where other places are empty except for i with k tokens.*
- *Proper termination: when o receives k tokens, all other places must be empty.*

Like weak soundness, it does not require liveness. The difference is the number of tokens that reach o from i, which is k. Following this, weak soundness is equal to k-soundness given that $k = 1$ [3]. Thus, Figure 1 satisfies k-sound when $k = 1$, i.e., 1-sound. Suppose that the same net has two tokens in i instead of one. This results in o gaining two tokens as well, making the net 2-sound. This behavior is consistent for every $k \in \mathbb{N}$, which is equal to generalized soundness [3].

2.3 Robustness Diagrams with Loop and Time Controls

The *RDLT*, based on the Robustness Diagram, encompasses the three workflow dimensions with its additional loop and time controls. This graph has a set of vertices V and a set of arcs E. Each vertex can be either a controller or an object, representing functions and components, respectively. Objects can also be either an entity or a boundary. Unlike objects, controllers can connect to other controllers. These nodes are connected by directed arcs, which have two attributes: C- and L-attributes. Specifically, the $C-$value of an arc depicts the constraints needed for traversal, if any. If there is no constraint, then the C-value of an arc is ε. The L-value tells us how many times traversal is allowed via the arc. An RDLT also has two more elements: the T-attribute, mapped onto E, shows the time step when the arc is traversed, and the M-attribute, which object vertices have. If such a value of an object is equal to 1, then it is the center of the Reset-Bound Subsystem (RBS), and every vertex that descends from the center is considered a part of such a subsystem. The RBS represents a volatile subsystem, where its L-values reset to their original value whenever traversal exits the RBS. Figure 2 shows a labeled example based on the given definitions.

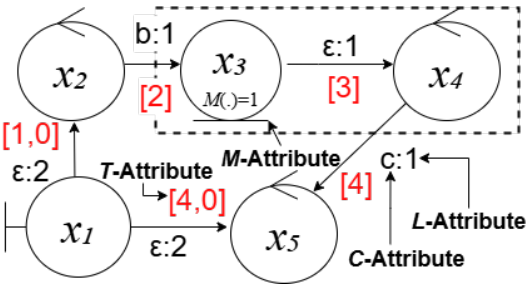

Fig. 2. A labeled Reset-Bound Subsystem-containing RDLT with a boundary object x_1, an entity object x_5, and controllers x_2, x_3, and x_4.

The *Activity Extraction Algorithm* is used to determine the behavioral profiles of an RDLT, starting from a source and ending at a sink [9]. The algorithm creates an activity S, which is a set of cases that involve the vertices. If R has no extractable activity, a null set is returned. Identifying structures within the model is important as it can help us determine the structural profiles that affect traversal. Listed below are such structures, as described in [10], which all merge at vertex y.

1. *AND-join*, which is comprised of arcs (v, y) and (u, y) with distinct and non-empty constraints, i.e., $C((v, y)) \neq C((u, y)) \neq \varepsilon$.
2. *MIX-join*, which is comprised of one or more pairs of arcs (v, y) and (u, y) with one non-empty and one empty constraint, i.e., $C((v, y)) \in \Sigma$ and $C((u, y)) = \varepsilon$.

3. *OR-join* which is comprised of type-alike arcs (v, y) and $(u, y) \in E$ where their C-values are equal, i.e., $C((v, y)) = C((u, y))$.

2.4 Parallel Activities of RDLTs

Multiple activities can be extracted from an RDLT depending on the paths and processes chosen by the algorithm for activity extraction, as structures can influence the algorithm's walk. To facilitate structural analysis, the concepts of activity groups and minimal activities act as the foundation for parallel activities [6]. An activity group, $ActGr(S)$, where S is an activity profile, consists of activities using the same arcs as S. Other activities may have arcs that loop back to previously reached vertices, called looping arcs [9]. A minimal activity, S_{min}, is the activity with the smallest set of arcs that reaches the sink from the source in $ActGr(S)$.

On the other hand, a maximal activity, S_{max}, is the activity with the largest set of arcs that reaches the sink from the source in $ActGr(S)$ [12]. Essentially, this activity consists of a minimal activity, including all looping arcs connecting vertices within the activity. Since maximal activities can contain all the structural and behavioral profiles of $ActGr(S)$, they have become the basis of recent literature on RDLT analysis [12,4].

As discussed earlier, the Activity Extraction Algorithm can be used with any RDLT to establish their behavioral profiles, making the algorithm essential for describing other workflow properties. However, this algorithm was formulated sequentially, thus making the extracted activity profile represent a single case. This results in the inability to depict processes that can handle multiple cases simultaneously. In response, the Parallel Activity Extraction (PAE) Algorithm was proposed, focusing in determining which activities in the RDLT R are parallel to each other by extracting all possible maximal activities in R [5]. When a split occurs, the algorithm induces a separation of activities. However, for joins, the type of the reached join decides whether the two activities merge or separate. AND-joins merge the activities into one, while MIX- and OR-joins induce a separate activity per joining process.

Remark 1. If R has only one maximal activity, i.e., $k = 1$, then verifying R will output 0 as it violates the definition of parallel activities, where more than one maximal activity is implied in R. However, the algorithm still generates a set of reachability configurations of the maximal activity. This is also the case if R has unparallel activities, of which such activities are still generated but not output.

Aside from the PAE algorithm, Doñoz and Malinao defined what constitutes parallel activities by listing four requirements given a pair of activities S and S' in each maximal activity in the RDLT. First, they must share the same input and output vertices. Second, they must not interrupt each other, i.e., S must not exit the RBS while still executing S'. This process interruption causes S' within the RBS to possibly exist longer than normally due to the reset in L-values after S exits. Third, they must not compete with each other. That is, S cannot

complete in the RDLT, which occurs if at least one of its processes completes, if S' is allowed to complete. Lastly, they must complete simultaneously.

Malinao and Juayong observed that the outputted maximal activities parallel to each other impede other activities within the same activity group [12]. In other words, parallel activities that have shared resources, such as arcs common between them, may compete with each other on the L-values, resulting in barring some activities from traversing such arcs at all. Thus, the set of maximal activities outputs determines whether the input RDLT is impedance-free, i.e, all activities terminate properly even with shared arcs.

2.5 Soundness Notions of RDLTs

For relevance, only classical and weak soundness notions will be discussed.

Definition 4 (Classical Soundness of RDLTs [10]). *An RDLT R is classical sound iff each activity profile S, composed of reachability configurations $S(1), S(2), \ldots, S(k)$, where k is lesser than or equal to the longest path of R, of a set of sources I and a sink f, where all arcs from each source is in $S(1)$ and all arcs to the sink is in $S(k)$, satisfies these conditions:*

1. ***Proper termination.*** *For every activity profile $S = \{S(1), S(2), \ldots, S(k)\}$.*
   - *All arcs in $S(k)$ point to f.*
   - *If $k \geq 2$, every arc incident to a vertex y in a reachability configuration $S(i)$ has an arc incident from vertex y in a succeeding reachability configuration $S(j)$.*
2. ***Liveness.*** *Every arc is traversed in some activity profile.*

Defined by Malinao and Juayong [10], classical soundness requires that a given RDLT uphold the sub-properties of proper termination and liveness, as done by the RDLT shown in Figure 2. In detail, the sink must be reachable from the source, the vertex reached at the last step must be the sink, and the vertices of the RDLT must all be traversed in at least one activity. These conditions imply that the model has no deadlocks and finishes all its processes during termination.

Definition 5 (Weak Soundness of RDLTs [14]). *An RDLT R is classical sound iff each activity profile S, composed of reachability configurations $S(1), S(2), \ldots, S(k)$, where k is lesser than or equal to the longest path of R, of a set of sources I and a sink f, where all arcs from each source is in $S(1)$ and all arcs to the sink is in $S(k)$, satisfies this condition:*

1. ***Proper termination.*** *For every activity profile $S = \{S(1), S(2), \ldots, S(k)\}$.*
   - *All arcs in $S(k)$ point to f.*
   - *If $k \geq 2$, every arc incident to a vertex y in a reachability configuration $S(i)$ has an arc incident from vertex y in a succeeding reachability configuration $S(j)$.*

Weakened from classical soundness, weak soundness only requires proper termination [14]. That is, the RDLT is considered weak sound if every reached vertex has a path to the sink and terminates simultaneously. This notion allows for deadlocks to exist in the model, given that it does not block the process from properly terminating. Specifically, deadlocks are allowed given that a separate path of vertices that observes proper termination from the parent of the vertex causing such a deadlock exists. Figure 3 visualizes a weak sound RDLT.

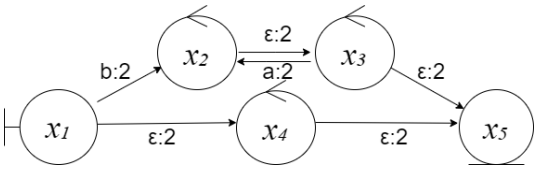

Fig. 3. A weak sound RDLT (based on [14]).

These definitions were based on those derived for WF-Nets by comparing the similar behaviors of both models and then establishing structures that enable such behaviors [10,14]. It also implies the possibility of defining looser notions by removing or changing some required conditions of formalized RDLT soundness notions. However, these studies that formalized such notions of soundness mainly focused on defining them sequentially. However, with the proposal of PAE [5], RDLTs can now be analyzed in a parallel context, i.e., a single RDLT model may represent the simultaneous traversal of more than one case.

3 Review of Related Works

To determine the soundness notion of an RDLT, verification strategies were introduced for the formalized RDLT soundness notions, along with their formal definitions [10,11,14]. Listed below are the methods that check for any deadlocks through unresolved constraints from either the L- or C-attributes. On the other hand, livelocks are unlikely due to the eventual exhaustion of L-values.

3.1 Compositional Analysis of RDLTs

To help understand the profiles of RDLTs with RBSs, vertex simplification is applied to separate the RBS from the other portions of the model [9]. Specifically, this process yields two subgraphs: level-1, which contains the system without the RBS, and level-2, which contains the RBS portion. With these outputs, algorithms can be applied to each subgraph individually. However, this process does not include the L-attributes. As a solution, Malinao and Juayong formulated the expanded vertex simplification [11].

Using the outputs of the vertex simplification process, graph contraction can be applied to ensure proper termination and liveness based on the C-attributes of

the model [9]. Essentially, this contraction process merges the vertices connected by an arc, given that traversal is possible. It starts from the source and continues until merging is no longer possible. If only one vertex remains, then C-attribute-based proper termination and liveness, subsequently, classical soundness, are observed by the model. If not, then only the option to complete is observed, given that the sink is also merged with the other vertices. This result can be used to distinguish weak sound RDLTs as shown in Figure 4.

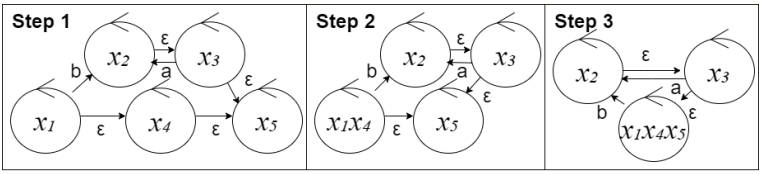

Fig. 4. Graph contraction of a weak sound RDLT (based on [14]).

3.2 L-Safeness of RDLTs without Resets

Similar to graph contraction, this method is used to verify various soundness notions of RDLTs. However, this method focuses on the L-attributes of the model, instead of the C-attributes, by checking deadlocks invoked through traversing looping arcs and the reuse of components [10,11,14].

Definition 6 (Cycles, Critical Arcs, and Escape Arcs [10]). *A cycle is a sequence of vertices, where an arc connects each consecutive pair of vertices, and no two vertices in the cycle are the same except for the first and last ones.*

An arc (x, y) in a cycle is called a critical arc (CA) if its L-value is the smallest one compared to all other arcs. If there exists an arc (x, v), where v is not a part of the cycle, it is considered an escape arc (EA) of (x, y) in the cycle. Self-loops, i.e., (x, x), are cycles that are themselves entirely composed of one CA that does not affect the (re)use of other arcs in an RDLT.

Definition 7 (Loop-Safe Arcs [10]). *Let R be an RDLT with some path from a source s to every other vertex in R. A non-critical arc (NCA) (x, y) is loop-safe if its L-value is larger than its reusability ($RU((x, y))$). That is, $RU((x, y)) = \Sigma_{k=1^{|Cycles(x,y)|}}(I * L((u, v)))$, for any arc (u, v) in the set of arcs which have the smallest L-value in other cycles that intersect with the current cycle being focused on. Such a set determines the value of I, where it is equal to 1 if an intersection exists between two cycles or if $k = 1$. Otherwise, $I = 0$.*

Definition 8 (Safe Critical Arcs [10]). *A CA (x, y) in E is safe if there is an escape, non-critical arc (x, z) in E, where (x, z) is loop-safe.*

These four types of arcs were focused on with a set of criteria known as join-safe L-values to ensure classical and relaxed soundness. However, such a set was weakened to permit deadlocks in the model to help verify weak RDLT soundness. Listed below are such criteria as well as other structures to be considered.

Definition 9 (Weakened Join-safe L-values and RDLT R [14]). *For every split-join pair of arcs (u, y), $(v, y) \in E$ in an RDLT R, (u, y) and (v, y) have weakened join-safe L-values if the following conditions are met:*

- *For every process with $C((u, y))$ or $C((v, y)) = \varepsilon$*
  1. *CAs are allowed if safe, i.e., $\exists$ a loop-safe non-critical EA for each CA.*
  2. *Branching out is allowed, as resolving its join is not affected by the arc that has a C-value of ε.*
  3. *Process interruptions are allowed for any C-value of the interrupting arc.*
- *For every process with $C((u, y))$ or $C((v, y)) = \Sigma$*
  1. ***One-split origin for sibling processes merging at an AND- or MIX-join.*** *Vertices u and v have one same ancestor vertex x, such that there are is a pair of sibling processes $P_u = a_1 \dots a_n$, where $a_i = x$, $a_{n-1} = u$, $a_n = y$, $a_i \in V$, $1 < I < n - 2$, and $P_v = b_1 \dots b_m$, where $b_j = x$, $b_{m-1} = v$, $b_m = y$, $b_j \in V$, $1 < I < m - 2$. They must have no arcs in common, i.e., they do not intersect except at x at y; and*
  2. ***No unrelated processes.*** *For every arc $(a, b) \in E$, if $a = x$, where x is the one common ancestor of u and v, then there is no path from a to another vertex $r \in V$ such that for some $(r, w) \in E$, $w \neq y$.*
  3. ***No branching out from every related process.*** *$\forall q_k \in Lit(P_q)$, where $q \in \{u, v\}$, $1 \leq k < |Lit(P_q)|$, $\nexists (q_{k,s}) \in E$ where $s \in V \backslash Lit(P_q)$. This is to ensure that the join is resolved.*
  4. ***Process interruptions.*** *With loss of generality, if the interrupting arc $(w, v) \in E$ has a C-value of ε, then interruptions are allowed.*
  5. ***Duplicate values.***
     - *If $C((u, y))$, $C((v, y)) \in \Sigma$, i.e., an AND-join at y, then there is no process $P = [x_1 x_2 \dots x_p]$ in R, where $x_1 = x$, $x_p = y$, such that $C((x_{p-1}, x_p)) = C((u, y))$ (or $C((v, y))$).*
     - *Without any loss of generality, if $C((u, y)) = \varepsilon$ and $C((v, y)) \in \Sigma$, i.e., a MIX-join at y, then any process $P = x_1 x_2 \dots x_p$ in R, where $x_1 = x$, $x_p = y$, can have $C((x_{p-1}, x_p)) \in \{\varepsilon, C((v, y))\}$, i.e., duplicate C-values are allowed, but no two arcs must have different C-values.*
  6. ***Given an AND-join, the L-values of arcs that comprise such a join must be the same.***
  7. ***No CAs exist within the process.***

If every pair has weakened join-safe L-values, then R is weakened join-safe.

Definition 10 (Point-of-Delay, Deadlock Point [9,14]). *A vertex x is a* Point-of-Delay (POD) *if it is connected to two preceding vertices by two type-alike arcs that do not have equal C-values. A deadlock point is each vertex $x \in dV$ in an RDLT R where dV is a set of PODs in R and $dV \subset V$, s.t. $x \in dV$ is unreachable at any time step during the extraction of activities.*

Definition 11 (Escape Contraction Path [14]). *An escape contraction path P is a sequence of vertices in R if these conditions are satisfied:*

1. *The first vertex of P is a parent of a deadlock point q.*
2. *P is a contraction path, where the last vertex of P is R's sink.*

Definition 12 (Deadlock-Resolving RDLT R [14]). *R is deadlock-resolving if every deadlock point has one or more escape contraction paths from its parent to the sink of the model.*

Definition 13 (Deadlock-Tolerant RDLT R [14]). *R is deadlock-tolerant if all critical and non-critical arcs are respectively safe and loop-safe. Additionally, R must be weakened join-safe and deadlock-resolving as well.*

Definitions 10 and 11 describe structures that must be present in a weak sound RDLT, as they allow proper termination of the activity even with the presence of deadlocks. Figure 4 shows the escape contraction path for the weak-sound RDLT in Figure 3, where it offers every reached vertex a path leading towards the sink. With this, such an RDLT is deadlock-resolving as defined in Definition 12. Subsequently, Figure 3 is also deadlock-tolerant as it observes the other requirements and is thus considered a weak sound RDLT structurally [14].

4 Methodology

This section contains the definitions, profiles, and verification methods of k-soundness of RDLTs based on the WF-Net soundness notions. However, the definition for k-soundness will focus on the maximal activities of the RDLT, as it encompasses all behavioral and structural profiles of a given activity group. To avoid confusion, this notion will be referred to as k^{max}-soundness.

Definition 14 (k^{max}-RDLT Soundness). *An RDLT R is k^{max}-**Sound** iff every set of maximal activities $S = \{A_1, A_2, \ldots, A_n\}, 1 \leq n \in \mathbb{N} \leq k$ where $k = |S|$, where $A_i = \{S(1), S(2), \ldots, S(n_i)\}$, $n_i \in \mathbb{N}$ and is equal to or less than the length of the longest path in R, of a set of sources I and a sink f, where all arcs from each source is in $S(1)$ and all arcs to the sink is in $S(k)$, satisfies these conditions:*

1. ***Proper Termination.*** *Every activity A_i in S completes at the sink and $|S| = |E_i|$ where E_i is the set of arcs $(x, y) \in E$ and $E_i \subseteq S(1)$ for all A_i where $x = s$, $y \in V$, i.e., A_i exists such that:*
   - *All arcs in $S(n_i)$ point to f.*
   - *If $n_i \geq 2$, every arc incident to a vertex y in a reachability configuration $S(i)$ has an arc incident from vertex y in a succeeding reachability configuration $S(j)$, i.e., for every $(x, y) \in S(i)$, there exists another arc $(y, z) \in S(j)$, for all $1 \leq i < k$, and for some j in the range $i + 1 \leq j \leq k$.*
   - *The number of activities A_i must be equal to the number of traversable outgoing arcs of the source vertex s.*

Similar to weak RDLT soundness, k^{max}-soundness only requires that every reached vertex has a path that eventually leads to the final vertex, i.e., proper termination. Another similarity is the allowance of untraversed arcs for every possible activity. In other words, an RDLT may not necessarily be live to be considered as k^{max}-sound. Thus, a k^{max}-sound RDLT can contain anomalies that may block traversal, such as deadlock, within the model.

Weak soundness only considers one activity per traversal of the RDLT, as it was defined sequentially. However, through its parallel nature, k^{max}-soundness considers that more than one activity may be traversed within the model at a given time. Specifically, the model must allow k^{max} number of activities to initiate, traverse, and terminate, similar to the WF-Net definition [3]. This creates the need to require every activity to terminate properly explicitly. With this, the k^{max} number of activities to be completed can be determined by the number of outgoing arcs from the source s. Similar to tokens in WF-Nets, these arcs represent activities initiated at the start of the system.

Figure 3 visualizes an example of an RDLT that exhibits k^{max}-soundness, where $k^{max} = 1$, as it only contains one maximal activity. In detail, this activity, A_1, is composed of $A_1(1) = \{(x1, x4)\}$ and $A_1(2) = \{(x4, x5)\}$. The upper portion of the model is not considered to be a maximal activity as it is not traversable. This incapacity of traversal is on account of the self-controlling arc, a profile that triggers a deadlock due to the violation of the unconstrainedness criterion [14], with the deadlock point x_2. This example also follows the requirements of weak soundness as was discussed in [14].

Figure 5 shows an RDLT that exhibits k^{max}-soundness, where $k^{max} = 2$. Like the earlier figure, it retains A_1. However, a new activity, A_2, composed of $A_2(1) = \{(x1, y1)\}$ and $A_2(2) = \{(y1, x5)\}$. These two processes do not merge at y due to the join they merge with, i.e., OR-join. With this, two activities are initiated and terminated, making the model 2-sound instead of 1-sound.

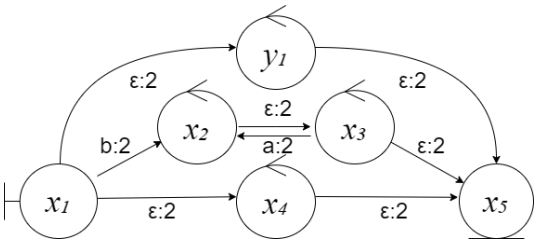

Fig. 5. An RDLT that satisfies k^{max}-Soundness where $k^{max} = 2$.

4.1 Profiles of k^{max}-Sound RDLTs

Since k^{max}-soundness accepts all completed activities, whether parallel or not, a weakened type of parallel activities is defined in Definition 15. Simultaneous completion is removed here to allow proper termination of non-parallel activities.

Definition 15 (Weakly Parallel Activities S_{weak}). *A set of maximal activities S of an RDLT R is weakly parallel, denoted as S_{weak}, if for every pair of activities A, $A' \in S$, the following conditions are met:*

1. *A and A' have the same set of input and output vertices*
2. *A and A' do not interrupt each other*
3. *A and A' are not competing activities*

Deadlock tolerance, used to verify weak RDLT soundness [14], ensures that the input RDLT observes proper termination even without liveness. This method can be used to ensure k^{max}-soundness in RDLTs, given the analogous nature of requiring proper termination. Through this, the Weak RDLT Soundness Verification Algorithm (WRSVA), as formulated in [14] to systematically verify the weak soundness through deadlock tolerance, can be used to check the k^{max}-soundness of an input RDLT. It must be noted that the WRSVA, listed in Appendix A.1, does not consider impedance, as this concept does not exist sequentially. With this, Definition 16 ensures the traversal of every arc in the RDLT.

Definition 16 (Impedance-Free L-values). *For each arc $(x, y) \in E$ and $A_i \in S$, $i \leq k$, $L((x, y)) \geq (|A_i| = n)$, i.e, all arcs in each derivable activity from R must have an L-value larger, if not equal, to the number of extractable activities.*

Even with the given concepts, there is still no guarantee that k^{max} activities within the RDLT are completed. To help ensure that the RDLT only contains k^{max} activities, a new requirement is introduced. Depending on the number of initiated activities, the needed number and type of splits and joins within the RDLT, as they determine the splitting and merging of activities, are specified to ensure that the number of completed activities matches.

Definition 17 (k^{max}-Ensuring RDLT R). *An RDLT R is k^{max}-ensuring if the number of joins merging activities J_m, joins separating activities J_s, and E_{split}, a set of traversable arcs $(x, y) \in E$ that form splits separating activities, i.e., OR- and AND- splits that do not have a looping arc, follow these constraints:*

- *$J_m = ceil(\frac{|E_{split}|}{2}) - k^{max} + 1$, i.e., the number of joins merging activities must be equal to the difference between the number of traversable arcs that separate activities and the number of initiated activities plus one.*
- *$J_s = k^{max} - 1$, i.e., the number of joins that separate activities must be equal to the number of initiated activities minus one.*

Figures 3 and 5 are examples of RDLTs that are k^{max}-ensuring, considering the number of initiated activities for each one. Lastly, Definition 18 describes an RDLT that encompasses the discussed profiles, which subsequently constitutes a k^{max}-sound RDLT. With this definition, a verification algorithm for k^{max} soundness, which checks for each aspect of a k^{max}-safe RDLT, is formulated and listed in Appendix A.3 along with its time and space complexities.

Definition 18 (k^{max}-Safe RDLT R). *An RDLT R with a set of weakly parallel activities S_{weak} is k^{max}-safe if R is deadlock-tolerant, k^{max}-ensuring, and has impedance-free L-values.*

5 Results and Discussion

Like most workflow models, the RDLT can represent real-world systems efficiently and in detail. This fact holds because this model encompasses all three workflow dimensions as mentioned earlier. One such usage is the visualization of the 2-bed adsorption chiller system [15]. Multiple parts in this system connect through various circumstances and loops. By representing the components, functions, and their respective interactions using the RDLT model, this results in a representation of the adsorption system as formulated and presented by Malinao [9]. Eclipse et al. adopted such a system and identified a substructure of the chiller system through an RDLT representation shown in Figure 6 [6].

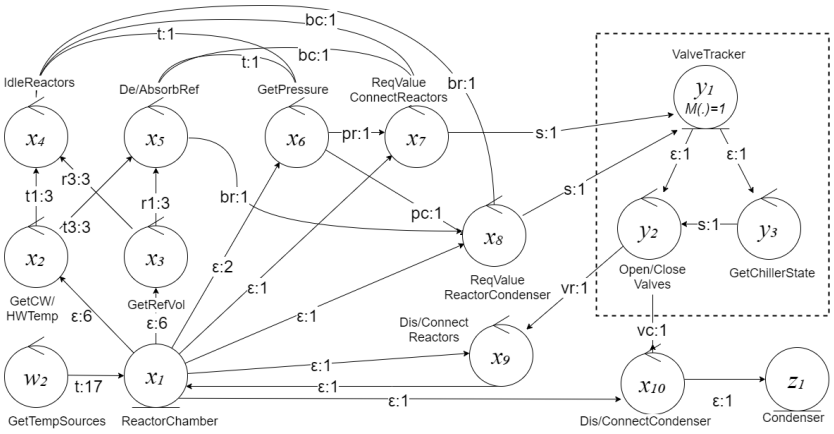

Fig. 6. An isolated RDLT subgraph of the adsorption chiller system (based on [6]).

5.1 Mapping of Real-World Systems

By isolating a subsystem from the portion in Figure 6, a k^{max}-sound RDLT was identified by selecting x_1 as the source, x_6 as the sink, and including the arcs and vertices interacting between x_1 and x_6. In this case, $k^{max} = 3$ as there are both three initiated and completed activities. However, some L-values were adjusted to adhere to the requirement of impedance-freeness of such values, i.e., changing the minimum L-value to 3, given $k^{max} = 3$.

From a realistic parallel standpoint, one can surmise from this chiller subsystem that three possible activities exist that use the components fully in their respective processes. Coming from the reactor chamber x_1, one activity is that the system traverses the path directly to the GetPressure function, x_6. This activity waits for the other two activities currently reaching the GetCW/HWTemp and GetRefVol vertices, x_2 and x_3, before performing the IdleReactors and De/AbsorbRef functions at x_4 and x_5. Lastly, all three activities terminate simultane-

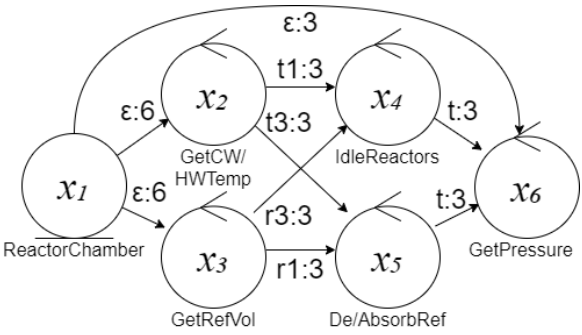

Fig. 7. A k^{max}-sound RDLT from a subsystem of the chiller system where $k^{max} = 3$.

ously at x_6, at which point the pressure levels of the entire system are retrieved. Thus, this subsystem is indeed considered as k^{max}-sound where $k^{max} = 3$.

5.2 Summary of Theorems and Algorithms

In this study, the proven theorems are listed below. A verification algorithm for k^{max}-soundness RDLTs was also formulated in the paper. The proofs for such theorems and the algorithm are listed in Appendices A.2 and A.3, respectively.

Theorem 1 (Implication of k^{max}-Soundness from the Weak Soundness of an RDLT). *If R is weak sound, then it is k^{max}-sound, where R has one traversable arc from the source vertex, and is impedance-free and k^{max}-ensuring, where $k^{max} = 1$.*

Theorem 2 (Implication of k^{max}-Soundness from the Classical Soundness of an RDLT). *If R is classically sound, then it is k^{max}-sound when $k^{max} = 1$.*

Theorem 3 (Proper Termination of a k^{max}-Sound RDLT). *A k^{max}-sound R properly terminates even without liveness.*

Theorem 4 (Structural Verification of k^{max}-RDLT Soundness). *R is k^{max}-sound iff both of its expanded vertex simplification levels R_1 and R_2 are k^{max}-safe.*

Theorem 5 (Time Complexity of k^{max}RSVA). *Verifying that R is k^{max}-sound uses $O(c|E|^4)$ time, where c and E represent the number of cycles and set of arcs in the model, respectively.*

Theorem 6 (Space Complexity of k^{max}RSVA). *Verifying that R is k^{max}-sound uses $O(|E|^2)$ space, where E represents the set of arcs in the model.*

6 Conclusion

In this study, k^{max}-soundness has been formalized in the context of RDLTs, including its profiles, which structurally show how a k^{max}-soundness RDLT behaves and is formed. The relationships between k^{max}-soundness and other formalized RDLT soundness notions have also been discussed in the paper. An algorithm, k^{max}RSVA, a verification strategy for k^{max}-soundness, along with its asymptotic analysis, is presented in this study. Lastly, a real-world system that satisfies k^{max}-soundness, where $k^{max} = 3$, was identified in this paper.

Future work focusing on k-soundness of RDLTs that do not necessarily only consider maximal activities must be advanced. Thus, it is recommended that the remaining soundness notions that have not yet been formalized in the context of RDLT be focused on. Namely, these are the generalized soundness and up-to-k-soundness notions, which have not been formalized due to their inherent parallel nature and the recentness of the Parallel Activity Extraction algorithm. Furthermore, the profile relationships between such remaining notions and k^{max}-soundness must also be analyzed further in terms of implication.

Acknowledgements R.I. Ramirez thanks the Department of Science and Technology - Science Education Institute (DOST-SEI), which supported his studies at the University of the Philippines Diliman through the Engineering Research and Development for Technology (ERDT) scholarship, during which the work presented here was formulated. R.A. Juayong acknowledges Robert Cheng, the URATEX Professional Chair in Engineering. F.G. Cabarle thanks the Scientific Productivity System of the University of the Philippines (2021-2023) for its support.

References

1. Aalst, van der, W.M.P.: Structural characterizations of sound workflow nets, Computing Science Reports, vol. 9623. Technische Universiteit Eindhoven (1996)
2. Aalst, van der, W.M.P.: Verification of workflow nets. In: Application and theory of Petri nets 1997. vol. 1248, pp. 407–426 (1997)
3. Aalst, van der, W.M.P., Hee, K.M., Hofstede, A.H.M., Sidorova, N., Verbeek, H.M.W., Voorhoeve, M., Wynn, M.T.: Soundness of workflow nets: classification, decidability, and analysis. Formal Aspects of Computing 23 (2011)
4. Afable, M.E., Malinao, J.: On Lazy Soundness of Robustness Diagrams with Loop and Time Controls. Bachelor's thesis, University of the Philippines Tacloban College, Tacloban City, Leyte, Philippines (2025)
5. Doñoz, R., Malinao, J.: Parallel Activities in Robustness Diagram with Loop and Time Controls. Bachelor's thesis, University of the Philippines Tacloban College, Tacloban City, Leyte, Philippines (2024)
6. Eclipse, K., Malinao, J.: Model decomposition of robustness diagram with loop and time controls to sequence diagrams. In: Proceedings of the Workshop on Computation: Theory and Practice (WCTP 2023) (2023)
7. Hollingsworth, D.: Workflow management coalition the workflow reference model (1995)

8. Lopez, J., Bayuga, M., Juayong, R., Malinao, J., Caro, J., Tee, M.: Workflow models for integrated disease surveillance and response systems. Theory & Practice of Computation, 2021 Taylor & Francis Group, ISBN 978-0-367-41473-3 (2020)

9. Malinao, J.: On Building Multidimensional Workflow Models for Complex Systems Modelling. Ph.D. thesis, Fakultät für Informatik (Pattern Recognition and Image Processing Group) Institute of Computer Graphics and Algorithm, Technische Universität Wien, Vienna, Austria (2017)

10. Malinao, J., Juayong, R.A.: Classical soundness in robustness diagram with loop and time controls. Philippine Journal of Science 152(6B) pp. 2327–2342 (2023)

11. Malinao, J., Juayong, R.A.: Reset profiles and classical soundness in robustness diagrams with loop and time controls. Pre-proceedings of the 12th Workshop on Computation: Theory and Practice (WCTP 2023) pp. 371–394 (2023)

12. 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)

13. Ramirez, R.I., Malinao, J.: On Weak Soundness in Robustness Diagrams with Loop and Time Controls. Bachelor's thesis, University of the Philippines Tacloban College, Tacloban City, Leyte, Philippines (2024)

14. Ramirez, R.I., Malinao, J.: On weak soundness in robustness diagrams with loop and time controls. In: Krouska, A., Mylonas, P., Caro, J. (eds.) Novel and Intelligent Digital Systems: Proceedings of the 5th International Conference (NiDS 2025). pp. 299–316. Springer Nature Switzerland, Cham (2026)

15. Rezk, A.R.M.: Theoretical and Experimental Investigation of Silica Gel/Water Adsorption Refrigeration Systems. Ph.D. thesis, University of Birmingham (2012)

16. Sulla, C.N., Malinao, J.: Mapping of robustness diagram with loop and time controls to petri net with considerations on soundness. In: Kabassi, K., Mylonas, P., Caro, J. (eds.) Novel & Intelligent Digital Systems: Proceedings of the 3rd International Conference (NiDS 2023). pp. 338–353. Springer Nature Switzerland, Cham (2023)

A Appendices

A.1 Weak RDLT Soundness Verification Algorithm [14]

Algorithm 1 Weak RDLT Soundness Verification Algorithm (WRSVA) [14]

Input: RDLT R with or without RBS
Output: True, false otherwise

Apply EVSA [11]
if R contains an RBS **then**
    $i = 2$
else
    $i = 1$
end if
for every vertex simplification R_i **do**
    for every cycle c **do**
        Determine if every non-critical arc is loop-safe and the critical arc is safe
    end for
    for every vertex $v \in V$ **do**
        Store deadlock point y
    end for
    for every deadlock point $y \in V$ **do**
        Determine if non-critical loop-safe escape contraction paths exist from a parent vertex x of y
        Determine if the split-join pair with y has weakened join-safe L-values
    end for
end for
if R_1 is deadlock-tolerant **then**
    if R contains an RBS **then**
        if R_2 deadlock-tolerant **then**
            return true
        else
            return false
        end if
    end if
    return true
else
    return false
end if

A.2 Structural and Behavioral Profiles of k^{max}-Sound RDLTs

Theorem 7 (Implication of k^{max}-Soundness from the Weak Soundness of an RDLT). *If R is weak sound, then it is k^{max}-sound, where R has one traversable arc from the source vertex, and R is impedance-free and k^{max}-ensuring, where $k^{max} = 1$.*

Proof. Let R be an impedance-free and 1-ensuring weak sound RDLT. Let R also have one arc $(x, y) \in E$ where $x = s$ and $(x, y) \subseteq S(1)$ for the sole A. Assume that R is not k^{max}-sound. Thus, the following claim must hold:

1. There exists an activity $A_1 = \{S(1), S(2), \ldots, S(k)\}, (q, o) \in S(k), q \in V, k \in \mathbb{N}$, in R, where $\exists (p, u) \in S(i), 1 \leq i < k$, s.t. $\nexists (u, y) \in \bigcup_{j=i+1}^{k} S(j)$ i.e.S does not properly terminate through $(p, u) \in S(i)$, specifically, the child $y \in V$ of u is a pending task of an unfinished process in S.

The sole requirement of k^{max}-soundness, proper termination, states that every reached vertex within every activity must have a path leading to the final vertex. When $k^{max} = 1$, there can only be one activity that traverses and completes in R, equivalent to the requirement of weak soundness. Theorem 3 also proves that a k^{max}-sound RDLT terminates properly even without liveness. Thus, a k^{max}-sound RDLT satisfies the proper termination that an RDLT of weak soundness must have.

Remark 2. With Theorem 1, RDLTs of weak soundness were proven to imply k^{max}-soundness. Conversely, however, the theorem cannot be proven as not all k^{max}-sound RDLTs are weak sound, specifically due to the weak soundness requirement that all activities must terminate at the same timestep, which may not be the case for k^{max}-sound RDLTs when $k^{max} > 1$.

Theorem 8 (Implication of k^{max}-Soundness from the Classical Soundness of an RDLT). *If R is classically sound, then it is k^{max}-sound, where R has one traversable arc from the source vertex, and R is impedance-free and k^{max}-ensuring, where $k^{max} = 1$.*

Proof. Follows from Theorem 1 and Theorem 3.3.1 of [14].

Theorem 9 (Proper Termination of a k^{max}-Sound RDLT). *A k^{max}-sound R properly terminates even without liveness.*

Proof. Follows from Theorem 1 and Theorem 3.4.1 of [14].

Theorem 10 (Structural Verification of k^{max}-RDLT Soundness). *R is k^{max}-sound iff both of its expanded vertex simplification levels R_1 and R_2 are k^{max}-safe.*

A.3 k^{max}-RDLT Soundness Verification Algorithm and Complexities

Algorithm 2 k^{max}-RDLT Soundness Verification Algorithm (k^{max}RSVA)

Input: RDLT R with or without RBS
Output: True, false otherwise

Apply Weak RDLT Soundness Verification Algorithm [14]
if R is not weak sound **then**
    return false
end if
Apply Expanded Vertex Simplification Algorithm [11]
if R contains an RBS **then**
    $i = 2$
else
    $i = 1$
end if
for every vertex simplification R_i **do**
    for every arc $(x, y) \in E$ **do**
        Determine if (x, y) has L-values greater than the number of activities k^{max}
    end for
    for every split-join pair $(x, y) \in E$ **do**
        Determine if each pair is k^{max}-ensuring
    end for
end for
if R_1 is k^{max}-ensuring **then**
    if R contains an RBS **then**
        if R_2 k^{max}-ensuring **then**
            return true
        else
            return false
        end if
    end if
    return true
else
    return false
end if

Theorem 11. ***Time Complexity of*** k^{max}***RSVA*** *Verifying that R is k^{max}-sound uses $O(c|E|^4)$ time, where c and E represent the number of cycles and set of arcs in the model, respectively.*

Proof. The algorithm is divided into four main processes: (1) WRSVA [14], (2) expanded vertex simplification [11], (3) verifying impedance-free L-values, and (4) verifying that the input is k^{max}-ensuring. First, WRSVA has a time complexity of $O(c|E|^4)$ according to [13]. For the second process, it takes $O(|V|^2)$ time, which corresponds to the largest number of arcs the input model can have.

Because the algorithm visits every arc of the diagram to replicate them or create abstract versions for the outputs R_1 and R_2 (if an RBS exists), the algorithm's computational complexity peaks when the number of arcs is at its largest, hence $O(|V|^2)$. Next, the third process uses $O(|V^2|)$ time because it goes through each arc and checks if its L-value is greater than or equal to k^{max}. The fourth process takes $O(|V| + |E|)$, corresponding to the total number of vertices V and arcs E in R. This process uses the depth-first or breadth-first search algorithm to solve the problem, hence $O(|V| + |E|)$.

Out of all the processes, WRSVA has the greatest complexity, making the time complexity of determining k^{max} soundness of an RDLT $O(c|E|^4)$.

Theorem 12. ***Space Complexity of*** k^{max} ***RSVA*** *Verifying that R is k^{max}-sound uses $O(|E|^2)$ space, where E represents the set of arcs in the model.*

Proof. As mentioned earlier, the algorithm is divided into four main processes. For the first process, the space needed for WRSVA is $O(|E|^2)$ [14]. Both the second and third processes have a space complexity of $O(|V|^2)$, matching the upper bound of arcs that R can contain. Because the algorithm stores every arc of the diagram to replicate them or create abstract versions for the outputs R_1 and R_2 (if an RBS exists), the algorithm's computational complexity peaks when the number of arcs is at its largest, hence $O(|V|^2)$. For the last process, it has a space complexity of $O(|V| + |E|)$ similar to its time complexity. It stores the vertices and arcs traversed to create the contraction path as it uses the depth-first or breadth-first search algorithm to solve the problem, hence $O(|V| + |E|)$.

Because verifying weak soundness has the greatest space complexity out of all the processes, the space needed to verify k^{max} soundness is $O(|E|^2)$.

Open Access This chapter is licensed under the terms of the Creative Commons Attribution-NonCommercial 4.0 International License (http://creativecommons.org/licenses/by-nc/4.0/), which permits any noncommercial use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons license and indicate if changes were made.

The images or other third party material in this chapter are included in the chapter's Creative Commons license, unless indicated otherwise in a credit line to the material. If material is not included in the chapter's Creative Commons license and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder.