ARTICLE

Model separability of robustness diagram with loop and time controls

Jasmine A. Malinao*[1] and Richelle Ann B. Juayong[2]

[1]Division of Natural Sciences and Mathematics, University of the Philippines Tacloban College, Tacloban City, Leyte, Philippines
[2]Department of Computer Science, College of Engineering, University of the Philippines, Diliman, Quezon City, Metro Manila, Philippines

ABSTRACT

In this study, we introduced and formalized the concept of separability of a multidimensional workflow model known as Robustness Diagrams with Loop and Time Controls. Separability reflects structures and behaviors in such models that are usable in effective and efficient modular design and analysis and establishing parallelizable activities of the system they represent. Through this research, we were also able to identify maximal substructures and activities that can become representatives of smaller ones with similar features within their respective activity groups. These maximal profiles, along with the relevant requirements for sharing of components, establish separable workflows. Furthermore, we also established the requirements of impedance-free workflows where activities therein do not interfere with each other from initiation to their completion. Then, we established the relationship between the separability and impedance-freeness of these workflows. Additionally, we provided proof of the correctness and time and space complexity of the algorithm and verification strategies that are needed or relevant in establishing separable and impedance-free workflows. Lastly, we posed recommendations for future research on extending this study towards the domain of parallelization of activities in these workflows.

*Corresponding author
Email Address: jamalinao1@up.edu.ph
Date received: February 28, 2024
Date revised: May 17, 2024
Date accepted: June 01, 2024
DOI: https://doi.org/10.54645/2024172CRV-31

1. Introduction

Real-world systems can be represented by models in various formats, levels of detail, abstractions, and complexity. These systems are analyzed for their satisfaction with some quality criteria or well-known workflow properties with the hopes of profiling what they can do or solve. In the field of system modeling and model verification, there is a rich and diverse set of frameworks and tools that aid in tasks from system planning, design and analysis, execution, maintenance, and evolution. Workflows such as Petri Nets, Business Process Modelling and Notation (BPMN)(Ko et al. 2009), WOODSS (WOrkflOw-based Spatial Decision Support System(Medeiros et al. 2005), the Unified Modelling Language (UML), Robustness Diagrams with Loop and Time Controls (RDLT)(Malinao 2017), among others, are such type of tools. Workflows are founded upon system representation built under three dimensions (van der Aalst 1996), namely, (a) resource, (b) process, and (c) case. Each of these dimensions would capture different sets of information

KEYWORDS

model verification, RDLT, separability, soundness, workflows

about the execution of activities that systems as described through their models. For example, UML Class Diagrams are uni-dimensional models that only focus on conveying the resource dimension of a system, i.e. objects that act upon their designated roles. Meanwhile, Petri Nets can express information under the process dimension and case dimensions. They are used to show tasks that are designed in a sequential, conditional, iterative, and parallel manner. Furthermore, they provide a way for modelers to simulate such tasks to verify if the given Net satisfies proper termination and if there are no unusable components, i.e. *the soundness property* (van der Aalst 1996). At times, workflows such as BPMN and RDLTs can help system analysts express system information using all three workflow dimensions with some level of care and consciousness to not overwhelm users, induce workflow errors, and maintain a verifiable and scalable workflow model. With this power of representation, RDLTs, for example, have been used to model and analyze real-world systems, such as HVAC systems (Malinao 2017), integrated disease surveillance and response systems (Lopez et al. 2020), and Fujitsu Ten's Computer Aided Multi-Analysis System Auto Test Tool (Malinao et al. 2013).

Whenever models evolve in size and complexity to keep up with their representation of a reference system, the inherent question of its maintainability, composability, scalability, or verifiability arises. For example, when a Petri Net is formed as a set of smaller modules, wherein each of its modules is sound, soundness at the level of the integrated Net is not guaranteed (van der Aalst 2000). Workflow models that represent activities that share components and/or resources can experience deadlocks or withdrawn active processes that can induce their non-completion of tasks. These can happen whenever submodules inside a model, or its entirety, are not properly designed or configured such that, regardless of the serialization or parallelization of such activities, this erroneous behavior would still manifest (Hauser et al. 2006, Kotb and Baumgart 2005, van Hee et al. 2003). In the context of RDLTs and their nature on multidimensional system representation, approaches have been introduced to either decompose an input RDLT into simpler models having lower dimensions, e.g. Sequence Diagrams (Eclipse and Malinao 2023a), Petri Nets (Sulla and Malinao 2023), Class Diagrams (Calvo and Malinao 2023), for information management with cognizance of loss of information, or transformed into matrix representations (Delos Reyes et al, 2018), along with the required matrix operations for activity simulation, for model verification. However, no literature to date has provided the concepts and techniques to obtain separable substructures that can facilitate effective and efficient module-based analysis and model verification. Moreover, none of them has also analyzed the impact of multiple modules and/or system activities having shared components such that the completion or possible parallelization of processes therein comes into question.

In this study, we address the abovementioned gaps in module-based representation and analysis of RDLTs. More specifically, we establish the definitions, requirements, efficient algorithms, and design strategies, and prove theorems regarding the separability of substructures and activity profiles in RDLTs regardless of the sharing of components and complex attributes innate to these models such as its reset structures and behavior.

Section 1.1 provides the basic notations, definitions, algorithms, and strategies for system representation and verification of RDLTs. We emphasize the concept and strategies around activity group (Eclipse and Malinao 2023b); structure-based computations on the reusability of components in activities in RDLTs that can have reset profiles; contraction paths to establish reachable components during process execution; and model simplification of RDLTs via vertex simplification for abstracted and level-based views of an input RDLT.

Section 2 provides our proposed methodology for establishing separable structures and profiles in RDLTs. We first establish the role of maximal activities of activity groups relative to other (maximal) activities in the same or different activity groups. We also improve the contraction path discovery in RDLTs to effectively extract minimal substructures, appending looping information in them as a post-processing step, to be able to generate maximal substructures in RDLTs. These maximal substructures are then used to generate their corresponding activity groups. By combining our knowledge of the reusability of components in activities, as well as our results on these substructures, we establish separability for RDLTs and its complexity of verification. In Section 2, we also introduce the concept of a composite activity that is built upon smaller activities, albeit maximal within their respective groups, and reuse the activity extraction algorithm in literature (Malinao 2017) on a transformed version of the input RDLT to determine impedance between/among the latter activities. With this, we establish impedance-free RDLTs. Through these results, we relate separable structures and maximal activity profiles of RDLTs concerning their impedance on activity completion, accounting for shared components between and/or among them.

Finally, Sections 3 and 4 provide and summarize our results and contributions, as well as establish an initial approach and recommendation to extend our work in the domain of parallel profiles in RDLTs.

1.1. Robustness Diagram with Loop and Time Controls

We provide the definition of RDLT below. In the context of this study, the set $\mathbb{N}$ of natural numbers is the set of positive integers.

Definition 1. (Robustness Diagram with Loop and Time Controls) (Malinao 2017)

A Robustness Diagram with Loop and Time Controls (RDLT) is a graph representation R of a system that is defined as $R = (V, E, T, M)$ where:

- V is a is a finite set of vertices. Each vertex can be of two types: an object or a controller. An object can be of two subtypes: a boundary or an entity.
  An object corresponds to a resource, e.g. person, file system, table, etc., while a controller corresponds to a task in a system.

- E is a finite set of arcs such that no two objects are connected. Furthermore, every arc (x, y) has the following attributes:
  - $C: E \to \Sigma \cup \{\varepsilon\}$ where Σ is a finite non-empty set of symbols and ε is the empty symbol. $C(x, y) \in \Sigma$ means that $C(x, y)$ is a symbol corresponding to a condition that is required to be satisfied, e.g. input/output requirement, to proceed from x to y. Meanwhile, $C(x, y) = \varepsilon$ means that there is no condition imposed by (x, y) or signifies that x is the owner object of the controller y.
  - $L: E \to \mathbb{N}$ is the maximum number of traversals allowed on the arc.

- Let T be a mapping such that $T(x, y) = (t_1, \dots, t_n)$ for every $(x, y) \in E$ where $n = L(x, y)$ and $t_i \in \mathbb{N}$ is the time a check or traversal is done on (x, y) by some algorithm's walk on R.

- $M: V \to \{0,1\}$ indicates whether $u \in V$ is a center of an RBS. An RBS is a substructure of G_u of R that is induced by a **center** $u \in V$, i.e. if $M(u) = 1$, and the set of controllers owned by u. (x, y) is an in-bridge of

G_u if x is not a vertex in G_u but y is. Conversely, (x, y) is an out-bridge of G_u if x is a vertex in G_u, but y is not. Lastly, a pair of arcs (a, b) and (c, d) are **type-alike** with respect to y if (a, b) and (c, d) are both in/out-bridges of y, or both are not. Note that an RBS has only one center.

Shown in Figure 1 is an RDLT R with $x_1(x_9)$ as its source(sink) and has one RBS whose center is x_3. The center x_3 has two owned controllers, i.e. x_2 and x_4. The entire RBS is annotated with a circle with dashed lines for emphasis. The in-bridge of x_3 is (x_1, x_3), while x_4 has the out-bridge (x_4, x_9). The arcs (x_2, x_4) and (x_3, x_4) are type-alike relative to x_4 since they are both not in/out-bridges of x_4. Meanwhile, (x_2, x_4) and (x_4, x_9) are not type-alike with respect to x_4 since (x_2, x_4) is not an in/out-bridge of x_4 while (x_4, x_9) is its out-bridge. By looking at the C-values, it can be realized that x_8 is an AND-JOIN(Malinao 2017), i.e. it requires that the conditions 'a' and 'b' are both satisfied before reaching x_8. Meanwhile, x_4 forms an OR-JOIN(Malinao 2017) since there are no conditions that need to be satisfied to reach x_4 from either of its parents x_2 and x_3. Had there been an in-bridge (x_8, x_4) for x_4 with a Σ-condition, e.g. 'a', this OR-JOIN is unaffected by this condition due to type-alikeness of these arcs; thus a traversal to x_4 from either x_2 or x_3 pushes on as long as the maximum allowable times, e.g. $L(x_2, x_4)$, has not been exhausted during the ongoing process executions of an activity inside the RBS.

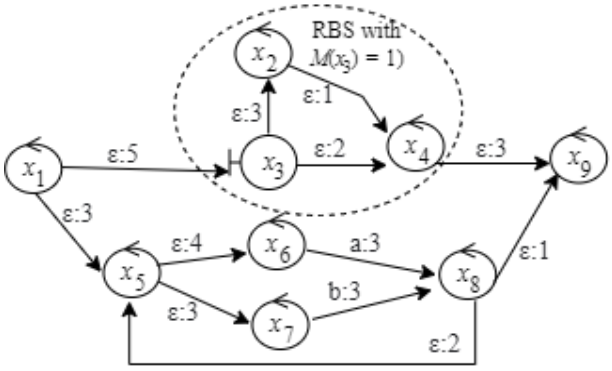

Figure 1: RDLT with a reset-bound subsystem with center at x_3, where $M(x_3) = 1$.

For simplicity of discussions, we shall focus on RDLTs with one source and one sink vertex. Nevertheless, readers can use the concept of an **extended RDLT** (Malinao 2017) for RDLTs with more than one of these vertices and apply the concepts and strategies of this paper as per usual. An extended RDLT of an input RDLT is adopted from the concept of a Workflow Net (van der Aalst 1996) of a given Petri Net. That is, the extended RDLT adopts all the information of the input RDLT as well as it contains one dummy source i (sink o) vertex that is connected to each original source s (sink f) vertex of the input R, establishing an OR-SPLIT(AND-JOIN)(Eclipse and Malinao 2023a) structure in this connection, i.e. $C(i, s) = \varepsilon$ (distinct $C(f, o) \in \Sigma$) relative to other members of the AND-JOIN), and L-value set to 1. In the context of our research, we modify this extension by setting an OR-JOIN at the sink o, rather than the usual AND-JOIN, and the L-value to a positive integer n. We call this modification the ***weakly-extended RDLT***. We shall discuss in later sections an optimal value for n with respect to RDLT model separability.

The so-called *activity profile* (Malinao 2017) describes an activity that a system, represented by an RDLT R, performs through its set of components and requirements for execution that they have. An **activity profile** $S = \{S(1), S(2), \ldots, S(k)\}$, $k \in \mathbb{N}$, is a set of **reachability configurations** $S(j)$, such that every $S(j)$ contains the set of components (i.e. arcs) that is reached/used at time step j starting from the source towards the sink of R. A depth-first search algorithm called the **activity extraction algorithm**(Malinao 2017) is used to extract one activity from R. This algorithm considers the arc attributes of R to know if a component can be (re)used in an activity, e.g. if its reuse has already reached the maximum allowable times of use as per its L-attribute values. Although this L-value of each arc (x, y) is already known by design-time, the reusability of (x, y) can be extended by placing this arc inside an RBS. Whenever an activity exits from the boundary of an RBS, the dynamic information stored in its T-attribute value that indicates the actual number of uses of (x, y) inside the recent execution of tasks inside the RBS is reset to 0. Thus, if this RBS is reused as part of the activity, (x, y) can be used again at most $L(x, y)$ times. As an example, if the activity traverses (x_4, x_9) in Figure 1, (x_2, x_4) is reusable even if it was used in a prior time step and its L-value is just 1.

From Figure 1, we can have an activity profile S with its reachability configurations $S(1) = \{(x_1, x_5)\}$, $S(2) = \{(x_5, x_6), (x_5, x_7)\}$, $S(3) = \{(x_6, x_8), (x_7, x_8)\}$, $S(4) = \{(x_8, x_5)\}$, $S(5) = \{(x_5, x_6), (x_5, x_7)\}$, $S(6) = \{(x_6, x_8), (x_7, x_8)\}$, $S(7) = \{(x_8, x_9)\}$.

An RDLT can have multiple activities that can be extracted from it using the activity extraction algorithm. Through these activities, an input RDLT can be checked for certain behavior or satisfaction of model properties, e.g. proper termination, and utility of all its components – i.e. *soundness* (van der Aalst 1996). However, this checking can take a significant amount of time and space because there are as many activity profiles as there are paths from the source s to the sink f, inclusive of repeatable paths induced from loops and the L-values of the arcs of R. With this, previous literature would introduce verification techniques for RDLT properties by use of the structural information in RDLTs. At times, this structural information would also be used to decompose RDLTs and transform the decomposed components into uni- or bi-dimensional models such as Petri Nets, the UML's Sequence Diagram, Class Diagrams, etc. This decomposition can aid in a more targeted analysis of some aspects of the input RDLT using some of its underlying substructures. As the latter models have limited syntax to represent all three workflow dimensions, it is expected that such limitation would also be present in fully representing the RDLT or its substructures.

Paper (Eclipse and Malinao 2023b) introduced the concept of *minimal activity* and *activity groups* in RDLTs as an aid for structural analysis and RDLT decomposition as presented in Definition 2. Every minimal activity is then used by a mapping in the paper to produce a set of sequence diagrams, and then integrated with loop fragments to be able to represent the activity group of each of these minimal activities.

Roughly speaking, a **minimal activity** in an RDLT is an activity for a given source $s \in V$ and an output sink $f \in V$ for which there are no other activities for s and f would have a smaller set of arcs participating in an activity. Moreover, an *activity group* for an activity S, denoted as $ActGr(S)$, is a set of activities that would have the same set of arcs as used in S and/or with the addition of the looping arcs whose endpoints are visited using the arcs of S. A looping arc (x, y) of a vertex y, as introduced in (Malinao 2017), is an arc in an RDLT that, when traversed, causes a reuse of y and its descendants. Note that every minimal activity for $[s, f]$ would have no arc component that is reused in the activity, thus, no loops can be found among its components.

Definition 2. (Activity Group, Minimal Activity) (Eclipse and Malinao 2023b)

Let $S = \{S(1), S(2), \dots, S(k)\}$, $k \in \mathbb{N}$, be an activity profile for the input-output pair $[s, f]$ of V in RDLT R. An **activity group** of S for $[s, f]$, denoted as $ActGr(S)$, is a set of activities in R where for every $S' \in ActGr(S)$, $S' = \{S'(1), S'(2), \dots, S'(k')\}$, $k' \in \mathbb{N}$, the following hold:

2. $A \cap B \neq \emptyset$, where $A = \bigcup_{i=1}^{k} S(i)$ and $B = \bigcup_{j=1}^{k'} S'(j)$, and,
3. without loss of generality, $\forall (y, x) \in B \setminus (A \cap B)$, (y, x) is a looping arc of x and $\exists (a, b), (c, d) \in A$ such that $x = \{a, b, c\}$ and $y = d$.

$ActGr_{max}(S)$ is a **maximal activity group** of S if there is no activity group $ActGr'(S)$ of S such that $ActGr_{max}(S) \subset ActGr'(S)$. Meanwhile, the activity $S_{min} \in ActGr(S)$, where $S_{min} = \{S_{min}(1), S_{min}(2), \dots, S_{min}(k_{min})\}$, $k_{min} \in \mathbb{N}$, is called a **minimal activity** of $ActGr(S)$ if $\forall S' \in ActGr_{max}(S)$, $\bigcup_{i=1}^{k_{min}} S_{min}(i) \subseteq \bigcup_{j=1}^{k'} S'(j)$.

From Figure 1, a minimal activity S_{min} is composed of $S_{min}(1) = \{(x_1, x_5)\}$, $S_{min}(2) = \{(x_5, x_6), (x_5, x_7)\}$, $S_{min}(3) = \{(x_6, x_8), (x_7, x_8)\}$, $S_{min}(4) = \{(x_8, x_9)\}$; one activity group for S_{min} is $ActGr(S_{min}) = \{S_{min}, S_1, S_2\}$, where;

1. activity S_1 is composed of $S_1(1) = \{(x_1, x_5)\}$, $S_1(2) = \{(x_5, x_6), (x_5, x_7)\}$, $S_1(3) = \{(x_6, x_8), (x_7, x_8)\}$, $S_1(4) = \{(x_8, x_5)\}$, $S_1(5) = \{(x_5, x_6), (x_5, x_7)\}$, $S_1(6) = \{(x_6, x_8), (x_7, x_8)\}$, $S_1(7) = \{(x_8, x_9)\}$. S_1 iterates through the AND-SPLIT and AND-JOIN substructure only once, along the components of S_{min} and the looping arc (x_8, x_5).
2. activity S_2 is composed of $S_2(1) = \{(x_1, x_5)\}$, $S_2(2) = \{(x_5, x_6), (x_5, x_7)\}$, $S_2(3) = \{(x_6, x_8), (x_7, x_8)\}$, $S_2(4) = \{(x_8, x_5)\}$, $S_2(5) = \{(x_5, x_6), (x_5, x_7)\}$, $S_2(6) = \{(x_6, x_8), (x_7, x_8)\}$, $S_2(7) = \{(x_8, x_5)\}$, $S_2(8) = \{(x_5, x_6), (x_5, x_7)\}$, $S_2(9) = \{(x_6, x_8), (x_7, x_8)\}$, $S_2(10) = \{(x_8, x_9)\}$. S_2 iterates through the AND-SPLIT and AND-JOIN substructure twice, along the components of S_{min} and the looping arc (x_8, x_5).

Note that the activity extraction algorithm cannot iterate through the split-join substructure the third time since it has exhausted the maximum allowable use of (x_8, x_5), i.e. $L(x_8, x_5) = 2$, and it is not inside an RBS so a reset thereof can extend the reusability of its components. There is no other activity group $ActGr'(S_{min})$ of S_{min} such that $ActGr(S_{min}) \subset ActGr'(S_{min})$, thus $ActGr(S_{min})$ is a maximal activity group of S_{min}.

Definition 2 opens the possibilities of a more efficient structural and behavioral analysis of RDLTs by simply looking at a set of the representatives of each activity group in an RDLT, rather than looking at *every* activity therein to conclude some properties in a model. However, rather than focusing on each of their minimal activities, we focus on each of their *maximal activity* – the activity in an activity group that contains the superset of arcs among all other activities therein. In Section 2, we shall show that this choice of representation is optimal as every maximal activity is usable for testing RDLTs at its limits as well as it is able to represent all the structures and behaviors of the other activities in its groups. Furthermore, we shall also show how they can be used as a helpful reference in determining parallelizable activities in RDLTs via some separation technique, with considerations of shared resources among separate activities as well as the presence of reset-bound subsystems.

1.2. Extracting Minimal Activities in RDLTs

A minimal activity for $[s, f]$ in R can be generated by identifying a substructure of R for which a contraction path(Malinao 2017) from s to f can be established through this substructure. This contraction path accounts for the reachability/usability of an arc in R for an activity profile by solely looking at its condition, i.e. C-value, apart from graph connectivity.

Roughly speaking, a contraction of an arc (x, y) collapses x and y into one merged vertex z. This contraction will only be possible if there is no other arc (u, y), where $x \neq u$, $C(u, y) \subseteq \bigcup_{\forall i} \{C(x, y)_i\} \cup \{\varepsilon\}$, and (x, y) and (u, y) are type-alike. Note that the contraction process can result in having multiple edges connecting two vertices. Thus, the notation $C(x, y)_i$ is the C-value of the i^{th} arc connecting x to y.

Figure 2 shows a series of contractions from the source x_1 to the sink x_9 of R in Figure 1.

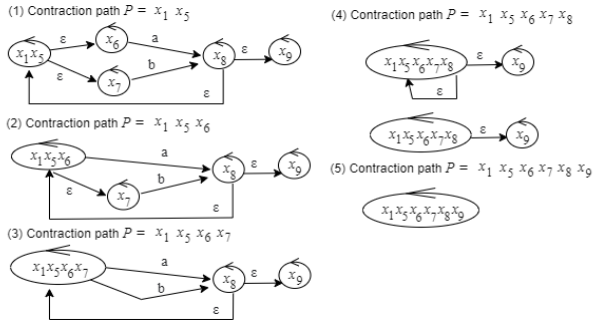

Figure 2: A contraction path from source x_1 to sink x_9 of R in Figure 1.

1.3. Expanded Reusability of Arcs

When determining various profiles of an input RDLT R, such as a substructure where there is a contraction path or behaviors such as system tasks that can be built from minimal activities, it had been demonstrated in the literature(Malinao and Juayong 2023a, Malinao and Juayong 2023b, Malinao 2017) that the (expanded) vertex simplification of R is highly useful. This technique has also been used to help prove various model properties such as the relaxed and classical soundness of RDLTs.

The expanded vertex simplification algorithm (EVSA) (Malinao and Juayong 2023a) creates at least two RDLTs known as the level 1 and level 2 vertex simplified multi-graph (Cormen et al. 2009) of controllers R_1 and R_2, respectively. R_1 is composed of a set of vertices and (abstract) arcs corresponding to the vertices and arcs of R that are found outside of or have at least one bridge in every RBS of R. Meanwhile, each R_2 is composed of a set of vertices and arcs corresponding to the vertices and arcs of R that are found inside one RBS of R. Since R_1 only retains those vertices that have at least one bridge, rather than all vertices inside an RBS, R_1 loses some of the details of the C- and L-attributes of arcs found in paths internal to this RBS. Nevertheless, by establishing an *abstract arc* (x', y') in R_1 between these retained vertices x' and y', where their respective vertices x and y in R have an internal path in the RBS of R, the reachability of y from x is still accounted in R_2 via (x', y'). EVSA then computes for the derived L-value of (x', y') by determining the maximum, possible number of times that it is used by an activity in R. This computation looks at the set of paths and cycles, if any, that can involve/reach (x, y) in this activity. Since (x', y') represents components inside the RBS, this computation also considers resets that can extend the reusability of (x, y) in this activity. The overall sum for the reusability of each arc (x, y) in R is called the **expanded reusability** of (x, y), denoted as $eRU(x, y)$ (Malinao and Juayong 2023b).

R_1 (and R_2) inherits the C- and L-values of the arcs in R to its own arcs, except for each abstract arc (x', y') where $C(x', y')$ is set to ε, and $L(x', y')$ the minimum expanded reusability of all the arcs along the path inside the RBS of R represented by (x', y') plus 1. Moreover, the M-values that establish centers and their induced RBS structures in R are not taken into $R_1(R_2)$ since $R_1(R_2)$ already represents connectivities outside(inside) each RBS.

Figure 3 shows the level 1 and level 2 expanded vertex simplification of R in Figure 1. Each arc has a label with the format $C(x, y)$:(derived) $L(x, y)$ $(eRU(x, y))$. We have two abstract arcs in R_1, i.e. $(x_3, x_4)_1$ and $(x_3, x_4)_2$ representing the internal paths $x_1 \to x_2 \to x_4$ and $x_3 \to x_4$, respectively, in the input R. Since R has no looping involved in its RBS, the level 2 simplification R_2 reflects the reusability of (x_3, x_4) as 0. (Malinao and Juayong 2023a) considers that each abstract arc (x', y') should not control the reusability of other non-RBS arcs, thus the computation of $eRU(x', y')$ adds 1 to the original reusability of (x, y) in R before extended reusability via its in-bridges of its RBS are considered. With this, the $eRU(x_3, x_4)$ is equal to 1 in R_1, and the derived L-value $L(x_3, x_4) = 2$. Meanwhile, the split-join substructure in R from x_5 to x_8 has each of its arcs (u, v) to have $eRU(u, v) = 2$ since looping within this substructure is only possible twice due to the controlling of iteration via the $L(x_8, x_9) = 2$.

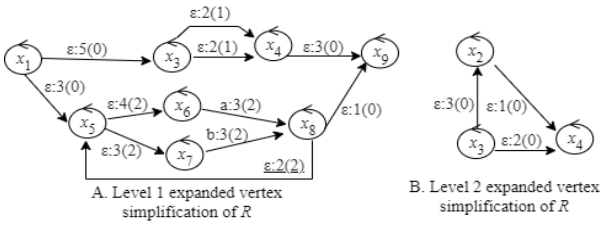

Figure 3: Expanded Vertex Simplifications of $\boldsymbol{R}$ in Figure 1. Each arc $(\boldsymbol{x}, \boldsymbol{y})$ in this figure has a label with the format $\boldsymbol{C(x, y)}$:(derived) $\boldsymbol{L(x, y)}$ $(\boldsymbol{eRU(x, y)})$.

With EVSA, we would be able to realize the reusability of RDLT components alongside the reset profiles of RDLT models. By combining the strategy on establishing maximal activity structures in RDLTs, we can then progress to establishing concepts and techniques to separate an input RDLT into fragments without loss of local and global information of systems as represented by their RDLT models. Such separation can be helpful whenever we want to study in isolation some substructures of a system, relate them against each other, and/or build more efficient mechanisms to study, observe, and simulate system behaviors in a parallel fashion.

2. Methodology for Establishing Separable RDLTs

In this section, we formalize the concepts and techniques relating to separability of RDLTs with a focus on maximal activities and structures that support their execution. We begin by looking at how individual maximal activities in activity groups and sharing of components between and among them influence the separability of an RDLT model, as well as their activity completion or impedance thereof, possibly incurred through these shared components. Subsequently, we establish the structural requirements and introduce algorithms that build and/or prove profiles of *separable RDLTs* with the help of the concept of maximal activity structures, composite activities, and looped RDLTs, among others. Thereafter, we prove relationships between separable and impedance-free RDLTs.

Definition 3. (Maximal Activity)

Let $S = \{S(1), S(2), \ldots, S(k)\}$, $k \in \mathbb{N}$, be an activity for the input-output pair $[s, f]$ of V in R.

Let $ActGr_{max}(S)$ be a maximal activity group of S for $[s, f]$ in R.

The activity $S_{max} \in ActGr(S)$, $S_{max} = \{S_{max}(1), S_{max}(2), \ldots, S_{max}(k_{max})\}$, $k_{max} \in \mathbb{N}$, is called a **maximal activity** for $[s, f]$ in R if $\forall S' \in ActGr(S)$, where $S' = \{S'(1), S'(2), \ldots, S'(k')\}$, $k' \in \mathbb{N}$, $\bigcup_{i=1}^{k_{max}} S_{max}(i) \supseteq \bigcup_{j=1}^{k'} S'(j)$.

If we analyze S_{max} of $ActGr(S)$, it would be an activity profile that is composed of the arcs in the minimal activity $S_{min} \in ActGr(S)$, along with all the looping arcs $(x, y) \in E$ where x and y are vertices found in S_{min}.

From the RDLT R in Figure 1, we have three(3) maximal activities S^1_{max}, S^2_{max}, and S^3_{max} for its input-output pair $[x_1, x_9]$, as follows:

1. S^1_{max} that is composed of $S^1_{max}(1) = \{(x_1, x_5)\}$, $S^1_{max}(2) = \{(x_5, x_6), (x_5, x_7)\}$, $S^1_{max}(3) = \{(x_6, x_8), (x_7, x_8)\}$, $S^1_{max}(4) = \{(x_8, x_5)\}$, $S^1_{max}(5) = \{(x_5, x_6), (x_5, x_7)\}$, $S^1_{max}(6) = \{(x_6, x_8), (x_7, x_8)\}$, $S^1_{max}(7) = \{(x_8, x_5)\}$, $S^1_{max}(8) = \{(x_5, x_6), (x_5, x_7)\}$, $S^1_{max}(9) = \{(x_6, x_8), (x_7, x_8)\}$, $S^1_{max}(10) = \{(x_8, x_9)\}$;

2. S^2_{max} that is composed of $S^2_{max}(1) = \{(x_1, x_3)\}$, $S^2_{max}(2) = \{(x_3, x_2)\}$, $S^2_{max}(3) = \{(x_2, x_4)\}$, $S^2_{max}(4) = \{(x_4, x_9)\}$;

3. S^3_{max} that is composed of $S^3_{max}(1) = \{(x_1, x_3)\}$, $S^3_{max}(2) = \{(x_3, x_4)\}$, $S^3_{max}(3) = \{(x_4, x_9)\}$;

Remark 1: In view of a weakly-extended RDLT, since our maximal activities share the arcs leading from(to) the dummy source(sink), it is easy to see that a good L-value of these arcs is at least the number of maximal activities in this RDLT.

2.1. Composite Activities and Activity Impedance

As a preliminary step to establishing profiles and techniques that help separate substructures of an input RDLT R based on its (maximal activities, we first introduce the concept of composite activities and impeding activities in R as shown below. Then, we look at the implications of shared components among these activities.

Definition 4. (Looped RDLT)

Given an RDLT $R = (V, E, T, M)$ with one source s and sink vertex f.

A **looped RDLT** $R_{loop} = (V_{loop}, E_{loop}, T_{loop}, M_{loop})$ of R is an RDLT derived from R where:

1. V_{loop} and E_{loop} are sets of vertices and arcs corresponding to V' and E', respectively, that inherit the same values of the vertex and arc attributes of R. Additionally, V'' contains a dummy source controller i and dummy sink controller o such that $(i, s'), (f', o) \in E_{loop}$ where $s'(f')$ correspond to $s(f)$ of R, with $C(i, s') = C(f', o) = \varepsilon$, $L(i, s'), L(f', o) \in \mathbb{N}$,

   and

2. $(f', s') \in E_{loop}$, with $C(f', s') = \varepsilon$, and $L(f', s') \in \mathbb{N}$.

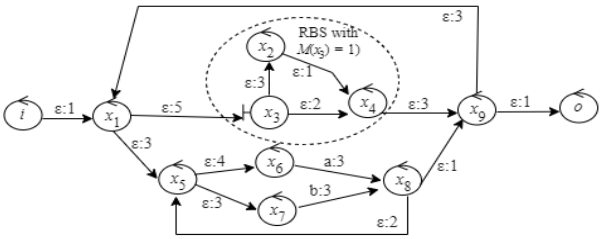

Figure 4: The looped RDLT $\boldsymbol{R_{loop}}$ of the input $\boldsymbol{R}$ in Figure 1.

Definition 5. (Composite Activity S')

Let $\boldsymbol{R}$ be an RDLT with its looped RDLT $\boldsymbol{R_{loop}}$ with $\boldsymbol{i}$ and $\boldsymbol{o}$ as its dummy source and sink controllers (see Definition 4).

An activity $\boldsymbol{S'}$ of $\boldsymbol{R_{loop}}$ is called a **composite activity** in $\boldsymbol{R_{loop}}$ for a set of activities $\boldsymbol{A} = \{\boldsymbol{S_1}, \boldsymbol{S_2}, \dots, \boldsymbol{S_n}\}$ in $\boldsymbol{R}$, $n \in \mathbb{N}$, $\boldsymbol{S_q} = \{\boldsymbol{S_q}(\boldsymbol{1}), \boldsymbol{S_q}(\boldsymbol{2}), \dots, \boldsymbol{S_q}(\boldsymbol{k_q})\}$, $\boldsymbol{k_q} \in \mathbb{N}$, such that $\boldsymbol{S'}$ is the **concatenation of the activities** in $\boldsymbol{A}$, denoted as $\boldsymbol{S'} = \boldsymbol{S_1} \oplus \boldsymbol{S_2} \oplus \dots \oplus \boldsymbol{S_n}$, where $\boldsymbol{S'} = \{\boldsymbol{S'}(\boldsymbol{1}), \boldsymbol{S'}(\boldsymbol{2}), \dots, \boldsymbol{S'}(\boldsymbol{k'})\}$, $k' \in \mathbb{N}$, with

1. $S'(1) = \{(i, s')\}$,
2. Let $\beta = \sum_{q=1}^{n}(k_q + 1)$ and $\Omega = \{\sum_{q=1}^{m}(k_q + 1) \mid 1 \le m \le n\}$.
   For each $t_{<q,p>} \in \{1, 2, \dots, \beta\} \backslash \Omega$ where $1 \le q \le n$ and $1 \le p \le k_q$, set
   $S'(1 + t_{<q,p>}) = \{(x', y') \in E_{loop} \mid (x', y') \in E_{loop}$ corresponds to $(x, y) \in E$, and $(x, y) \in S_q(p)\}$,
3. For each $t_{<q,p>} \in \Omega$ where $1 \le q \le n$ and $1 \le p \le k_q$, set
   $S'^{(1+(t_{<q,p>}+q))} = \{(f', s')\}$
4. $S'(k') = \{(f', o)\}$, where $k' = 2 + \sum_{q=1}^{n}(k_q + 1)$

In essence, the activity S' resulting from the concatenation of activities in A simulates the execution of S_1 first in R_{loop}, initiating it via i at first, and then from s' to f' (via the components of S_1). Thereafter, this activity goes back to s' from f', and then simulates S_2 from there in R_{loop}, and so on. After iterating through these simulations until the last activity S_n, the activity S' completes in R_{loop} by traversing (f', o) at time step k'.

Remark 2. The composite activity S' for A will facilitate the checking of the possibility that every activity $S_q \in A$ completes in full, with respect to the arcs involved in S_q. With this, $L(f', s')$ should be at least $|A|$, i.e. the number of activities in A which are tested for their completion relative to each other.

In addition, whenever R is a multi-source(multi-sink) RDLT, simply connect $i \in V_{loop}(f' \in V_{loop})$ to every $s' \in V_{loop}(o \in V_{loop})$ where $s'(f')$ corresponds to a source $s \in V$ (sink $f \in V$) of R, where $C(i, s') = \varepsilon$ and $L(i, s') = 1$ ($C(f', o) = \varepsilon$ and $L(f', o) = 1$). Similarly, set every $L(f', s')$ as described above.

Definition 6. (Non-impeding Activities)

Let R be an RDLT with its source and sink vertices s and f, respectively, and R_{loop} be its looped RDLT with i and o be its dummy source and sink vertices.

Let A be a set of activities in R for the pair $[s, f]$.

We say that A has no activities that impede each other in R if there exists a composite activity C in R_{loop} for the activities in A.

Building a composite activity S' from a set of activities of R in Figure 1, say its maximal activities S^1_{max}, S^2_{max}, and S^3_{max}, we have $S' = \{S'(1), \dots, S'(21)\}$ where $S'(1) = \{(i, x_1)\}$, $S'(2) = \{(x_1, x_5)\}$, $S'(3) = \{(x_5, x_6), (x_5, x_7)\}$, $S'(4) = \{(x_6, x_8), (x_7, x_8)\}$, $S'(5) = \{(x_8, x_5)\}$, $S'(6) = \{(x_5, x_6), (x_5, x_7)\}$, $S'(7) = \{(x_6, x_8), (x_7, x_8)\}$, $S'(8) = \{(x_8, x_5)\}$, $S'(9) = \{(x_5, x_6), (x_5, x_7)\}$, $S'(10) = \{(x_6, x_8), (x_7, x_8)\}$, $S'(11) = \{(x_8, x_9)\}$, $S'(12) = \{(x_9, x_1)\}$, $S'(13) = \{(x_1, x_3)\}$, $S'(14) = \{(x_3, x_2)\}$, $S'(15) = \{(x_2, x_4)\}$, $S'(16) = \{(x_4, x_9)\}$, $S'(17) = \{(x_9, x_1)\}$, $S'(18) = \{(x_1, x_3)\}$, $S'(19) = \{(x_3, x_4)\}$, $S'(20) = \{(x_4, x_9)\}$, $S'(21) = \{(x_9, o)\}$. Note that S' can be completely simulated in R_{loop} from i until it reaches its intended sink o. Thus, we say that S^1_{max}, S^2_{max}, and S^3_{max} do not impede each other in R.

Theorem 1. Given an activity S and its maximal activity group $ActGr(S)$, let S_{max} be the maximal activity of this group. For every activity $S' \in ActGr(S)$, where S' has components that are involved in a cycle, S' and S_{max} impede each other.

Proof. Suppose that S' and S_{max} do not impede each other in R, where S' has components that are involved in a cycle. That is, there exists a composite activity $S_{com} = S_{max} \oplus S'$ that is derivable from the looped RDLT R_{loop} of R. Note that the set of components of S_{max} is a superset of the components of S'. Thus, we can select some (x, y) of S_{max} and analyze its use inside S_{com}.

Suppose (x, y) is involved in some cycle in S_{max}. Using R_{loop}, we shall simulate S_{com} using S_{max} first. We simulate S_{max} and pass through (x, y) by the number of times it is reused in S_{max}. Since S_{max} is a maximal activity of R, this number is the actual reusability $eRU(x, y)$ of (x, y). Upon the completion of the simulation of S_{max} in R_{loop}, the activity proceeds with traversing (f', s') of R_{loop}. From there, we simulate S' and pass through (x, y) again using the number of times n that S' reuses (x, y). However, since we have already exhausted the allowable number of times that (x, y) is used through the simulation of S_{max} on R_{looped}, we can never pass through (x, y) again when we try to simulate S', regardless if (x, y) is inside or outside of an RBS. Thus, the composite activity S_{com} is not realizable in R_{loop}, ergo S_{max} and S impede each other. With this, we arrive at a contradiction.

□

With Theorem 1, since the set of components of S_{max} is a superset of the components of $S' \in ActGr(S)$, we can use S_{max} to analyze activity flows in any such S' within the group, or in general, analyze the structure and behavior of R through the relationships of its set of maximal activities. Furthermore, we set our goal in this paper to analyze these relationships across different maximal groups for the purpose of giving system designers and analysts the concepts and techniques that would help them fragment an input RDLT into separable (and manageable) substructures without compromising and omitting its innate system information. With this, we establish *impedance-free* RDLTs in the context of their maximal activities as shown in Definition 7.

Definition 7. (Impedance-free RDLT)

An input RDLT R is called **impedance-free** if every pair of its maximal activities for its source and sink do not impede each other.

The RDLT R in Figure 1 is impedance-free since every pair of its maximal activities S^1_{max}, S^2_{max}, and S^3_{max} do not impede each other.

2.2. Techniques on RDLT Separation

2.2.1. *Extracting Maximal Activities*

To extract maximal activities from a given RDLT R (with 1 input and output vertices), we shall first generate its Level 1 and Level 2 expanded vertex simplification R_1 and R_2, respectively. If R_2 has multiple source/sink vertices, update R_2 as its weakly-extended RDLT so that it only has one dummy source and sink vertex with an OR-JOIN at the dummy sink.

Then, we establish a contraction path P from a source vertex s_{R_i} of R_i to its sink vertex f_{R_i}. The set of vertices and edges that are involved in P may not compose a minimal activity in R_i. That is, this set can include looping arcs or can include a sub-path P' from s_{R_i} to f_{R_i} where at least one of its vertex v that is involved in a MIX-JOIN(Malinao 2017) or an OR-JOIN has (u,y) and (v,y) have duplicate C-values, i.e. $C(u,y) = C(v,y)$. Note that for the latter case, y can be used(reached) by any activity by satisfying $C(u,y)$ without the need to satisfy $C(v,y)$ (or vice versa). Hence, we shall prune out such duplications without compromising the required paths along the contraction path P from the source to the sink of R_i.

To execute the above-mentioned pruning, we shall assign a positive integer weight to each arc that is necessary to reach the sink from the source of R_i along P. Initially, this weight per arc is set to 0. After initialization, we collect every merge point y along P. For every arc (x,y) whose C-value is different from the other arcs (u,y), we add 1 to each arc involved along the path Q from the source or another merge point along Q, whichever is nearer to y, to (x,y). We do this too for one path Q' whose component (x,y) has a duplicated C-value relative to another arc (v,y) along P. After this process, arcs that are duplicates of others with respect to their C-values, along with their ancestors which are unnecessary to reach the sink from the source of R_i, i.e. with final weights of 0 as well, are removed from P. (Looping arcs are also removed from P in this process of pruning.) The other arcs with a weight of at least 1 are deemed to be necessary and the minimum set to reach the sink from the source.

For brevity, we call the substructure R_{min} of R_i as a *Minimal Contraction Structure (MinCS)* where R_{min} is induced by the arcs composing the (pruned) contraction path P in R_i.

With respect to getting a minimal activity of the entire input RDLT R, we consolidate the components of R that compose a pruned contraction path P from s to f of R_1, and a set of pruned contraction paths in R_2 from a set of sources I_{R_2} and set of sink vertices O_{R_2} in R_2 such that every $o_{R_2} \in O_{R_2}$ appears as part of the contraction path P of R_1, and at least one $i_{R_2} \in I_{R_2}$ has a path towards $o \in O_{R_2}$.
Algorithm 1 below shows the Modified Contraction Algorithm (MCA) that extracts a minimal contraction path P in R_i.

Algorithm 1: The Modified Contraction Algorithm (MCA)

Input: RDLT R with one source s and sink f

Output: A minimal contraction path P for the source and sink of R_i, $i \in \{1,2\}$.

Steps:

Get Level-i vertex simplification of R by EVSA(Malinao and Juayong 2023a).
Select i, $i \in \{1,2\}$ to build P
If $i = 2$ AND R_2 has more than 1 source/sink **then**
    Update R_2 as a weakly-extended RDLT. //sets R_2 to have 1 dummy source/sink with an OR-JOIN at the sink
Let $s' \in V_i$ and $f' \in V_i$ be the source and sink of R_i
Let $x = s'$. //x shall act as the dummy node representing merged vertices in P
Initialize $P = \{x\}$.
From x to f'
    Select $y \in V_i$ where $(x,y) \in E_i$
    Select $y \in V_i$ where $(x,y) \in E_i$
    If $\bigcup_{\forall j}\{C(x,y)_j\} \cup \{\varepsilon\} \supseteq \bigcup_{\forall (u,y) \in E_i, u \neq x}\{C(u,y)\}$
        Update $C(u,y) = \varepsilon, \forall (u,y) \in E_i, u \neq x.$
        Merge x to y.
        Update $P = P \cup \{y\}$.
Induce R_{min} from R_i using the vertices in P.
//Prune duplicate paths within the contraction path P
For each arc (p,q) in R_{min}
    Initialize $weight(p,q) = 0$.
Let MP be the set of vertices in R_{min} where each vertex is a merge point of a JOIN
For each $y \in MP$
    //Add 1 to the weight of arcs in every(one) path that has a
    //distinct(duplicated) C-value at merge point
    Let $Q = v_1 v_2 \ldots v_n$ be an elementary path from the source s'' of R_{min} to y where $\nexists (u,y)$ in R_{min} such that $C(v_{n-1}, v_n) \neq C(u,y)$.
    From $j = n$ to 2
        Update $weight(v_{j-1}, v_j) = weight(v_{j-1}, v_j) + 1$.
        If $v_{j-1} \in MP$ **then *break***.
    Let $Q' = u_1 u_2 \ldots u_m$ be an elementary path from the source s'' of R_{min} to y where $\exists (v,y)$ in R_{min} such that $C(u_{m-1}, u_m) = C(v,y), u_{m-1} \neq v$.
    From $j = m$ to 2
        Update $weight(u_{j-1}, u_j) = weight(u_{j-1}, u_j) + 1$.
        If $u_{j-1} \in MP$ **then *break***.
Remove from R_{min} any arc (u,v) with $weight(u,v) = 0$.
Update P to be the set of the remaining vertices of R_{min}
Output P.

Lemma 1. MCA produces a minimal contraction path P whose MinCS R_{min} composes a minimal activity S_{min} in the expanded level-i vertex simplification R_i, $i \in \{1,2\}$, of R.
Proof. We prove Lemma 1 by contradiction. That is, let S be an activity in R_i where the components of $S = \{S(1), S(2), \ldots, S(k)\}$ are derived from using MinCS R_i' obtained through such contraction path P of R_i. Furthermore, suppose $\bigcup_{j=1}^{k} S(j) \subset \bigcup_{j'=1}^{k'} S_{min}(j')$. With this, there exists an arc

(x, y) that was traversed in S_{min} but not in S.

1. **Case 1:** If $C(x, y) \in \Sigma$. This case implies that there is either an MIX- or AND-JOIN that merges at y. Note that for a MIX-JOIN, any (u, y) such that $C(u, y) = C(v, y)$ is pruned out from R_{min} at steps 16-35, $C(x, y)$ already represents every such (u, y) in the evaluation of the reachability of y in R_i. Moreover, if P contains the other component of this MIX-JOIN, i.e. (v, y) where $C(v, y) = \varepsilon$, (v, y) can never be contracted towards y without the required condition $C(x, y)$. Thus, P without (x, y) is not a contraction path for R_i, thus S cannot exist on R_{min}, where $\bigcup_{j=1}^{k} S(j) \subset \bigcup_{j'=1}^{k'} S_{min}(j')$.

   Meanwhile, if y is a merge point for an AND-JOIN involving (x, y) and some (u, y) in R_i, i.e. $C(u, y) \in \Sigma$ and $C(x, y) \neq C(u, y)$, y can never be contracted within P if (x, y) is missing. Similarly, P without (x, y) is not a contraction path for R_i, ergo, such S cannot exist on R_{min}.

   Thus, in both scenarios, we arrive at a contradiction.

2. **Case 2:** If $C(x, y) = \varepsilon$. This case implies that there is an OR-JOIN, or simply a sequential process, that merges at y. By the pruning step of MCA, all other (u, y) with the same C-value, i.e. ε, is removed from R_{min}. This step would then imply that there is exactly one path from x to y, as well as from the source of R_i to y, and then to its sink in R_{min}. In essence, the arc (x, y) acts as a bridge in R_{min} such that R_{min} becomes a disconnected graph with the absence of (x, y). Thus, P is not a contraction path from its source towards its sink without (x, y). With this, thus we proved that S cannot exist on R_{min}, where $\bigcup_{j=1}^{k} S(j) \subset \bigcup_{j'=1}^{k'} S_{min}(j')$ – a contradiction.

□

Lemma 2. MCA builds a minimal contraction path P, and its corresponding MinCS R_{min} in the expanded level-i vertex simplification R_i, $i \in \{1,2\}$, of the RDLT R, using $O(|V|^3)$ and $O(|V|^2)$ time and space complexity.

Proof. For Line 1, EVSA builds R_i from R with $O(|V|^2)$ in time and space complexity as per (Malinao and Juayong 2023a). Extending R_i at Line 4 takes constant time and space. To build a contraction path P from the source s to f of R_i from Lines 8-14, it can take the diameter $d \in \mathbb{N}$ of R_i to build P, and then multiplied with $O(|V|^2)$, or $O(d|V|^2)$ in time and space complexity. This diameter is computed solely using the elementary paths of R_i, thus $d \in O(|V|)$. This complexity accounts for the worst-case wherein each contraction may require the inclusion of all paths from a split point towards its corresponding merge point y, i.e. merging via an AND- or MIX-JOIN. For the latter type of JOIN, contraction may first go through all the paths ending with ε at the merge point before going through a Σ-condition arc (v, y) to resolve this JOIN. For the pruning step of MCA, i.e. from Lines 16 and onwards, we have the following steps and their corresponding time and space complexity:

1. listing the merge points in R_{min} in MP – this takes $O(|V|^2)$.

2. Pruning R_{min} for each elementary path $Q(Q')$, adding 1 to each necessary arc of R_{min} – This step takes $O(d|V|^2)$ or $O(|V|^3)$. This accounts that every merge point can have $O(|V|)$ connections. Since we do this for each merge point along the diameter d of R_i, we therefore obtain $O(|V|^3)$ and $O(|V|^2)$ in time and space complexity, respectively.

□

Definition 8. (Maximal Activity Structure)
A **Maximal Activity Structure(MAS)** $R_{MAS} = (V_{MAS}, E_{MAS}, T_{MAS})$ of $R_i = (V_i, E_i, T_i, M_i)$ for its source s_{R_i} and sink vertex o_{R_i} is a projection of R_i induced by the components of its MinCS R_{min} for and every looping arc (x, y) of R_i where x and y are vertices found in R_{min}, where for every edge $(u', v') \in E_{MAS}$ corresponding to $(u, v) \in E_i$ of R_i,

$$L(u', v') = \begin{cases} 1, & \text{if } (u', v') \text{ appears in } R_{min} \text{ and} \\ & (u, v) \text{ is not a part of a cycle in } R_i, \\ L(u, v), & \text{otherwise.} \end{cases}$$

Figure 5 shows the set of maximal activity structures in items (a) and (b) of R_1 and R_2, respectively, of the RDLT R in Figure 1. For item (a), note that there are two abstract arcs connecting x_3 and x_4 in R_1. Thus, due to the pruning step of MCA to extract the MinCS of R_1, it was able to find two MinCS substructures along these abstract arcs due to their duplicate C-values, i.e. $C(x_3, x_4)_1 = C(x_3, x_4)_2 = \varepsilon$. Thus, these substructures in item (a) eventually result to R^1_{MAS} and R^2_{MAS} as two of the three MAS of R_1. Furthermore, none of the components of these substructures are involved in a cycle, thus, their L-values are set to 1 by Definition 8. Meanwhile, R^3_{MAS} has each of the components of the split-join substructure having its L-value set to its expanded reusability, i.e. 2, as per Definition 8 as well.

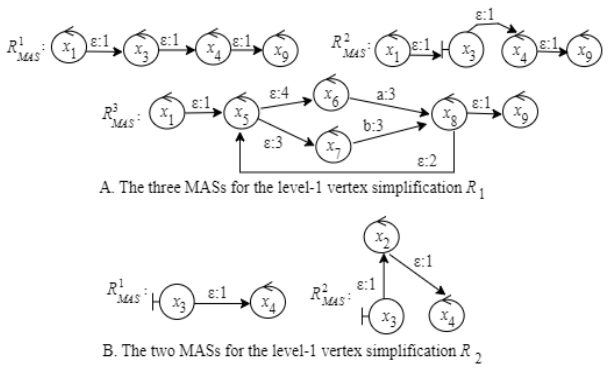

Figure 5: The Maximal Activity Structures (MASs) of R_1 and R_2 of R in Figure 1.

Theorem 2. For every MAS R_{MAS} of R_i, there exists a corresponding maximal activity $S_{max} = \{S_{max}(1), S_{max}(2), \ldots, S_{max}(k)\}$ of R_i whose components use R_{MAS}.

Proof. For this, we prove that there exists an activity $S = \{S(1), S(2), \ldots, S(k)\}$ derivable from R_{MAS} where $\forall j$, $S(j) = S_{max}(j)$ (or $S = S_{max}$).

Let R_{min} be the MinCS that is used to derive R_{MAS} as per Definition 8. Furthermore, let $S_{min} = \{S_{min}(1), S_{min}(2), \ldots, S_{min}(k')\}$ be a minimal activity in R_i(as proved in Lemma 1) that is derivable from R_{min}.

We construct S via the components in S_{min} such that we adopt the precedence of every pair $S_{min}(t)$ and $S_{min}(t + j)$, $1 \leq t \leq k'$, onto S, along with the incorporation of all loops involved

between two pairs of vertices that are found in R_{min}(and S_{min}). That is, we simulate the activity extraction algorithm on R_{MAS}, building S, by tracing the components of S_{min}, momentarily suspending this tracing if the algorithm sees a looping arc (x, y) along the previously traced components. The algorithm shall iterate on these components as long as $L(x, y)$ of R_{MAS} allows it to. Once this looping is exhausted, then, the algorithm continues to build using the subsequent components of S_{min} until the next looping arc is encountered. These processes would iterate as stated until the last components of S_{min} are simulated in R_{MAS} and adopted in S.

Since the L-value of each arc (x, y), where (x, y) is involved in a loop in R_{MAS}, is equal to the L-value of its corresponding arc (x', y') in R_i (by Definition 8), then the maximum number of times that the algorithm iterates on R_{MAS} to build S is the same as the maximum number of times that (x', y') is used in S_{max}. Furthermore, since S_{max} uses the arcs involved in R_{MAS}, the utility and progression of use of each arc in S_{max} is consistent with that of S. Thus, we proved that $S = S_{max}$.

□

2.2.2. *Separable RDLT*

From here on, we would start dealing with how to separate components or substructures of the RDLT based on their aggregation to form an activity in R. We shall separate an input RDLT based on its maximal activity structures where every such structure is usable to generate an activity group of a minimal activity in R. Identifying and separating such structures in an input RDLT can give us an insight into how shared components affect the parallelizability of activities if and when such activities need to be executed at the same time.

Definition 9. (Separable RDLT)

Given $R_i = (V_i, E_i, T_i)$ and a $k \in \mathbb{N}$, R_i is **k-separable** if there exists exactly k number of MAS $R^j_{MAS} = (V^j_{MAS}, E^j_{MAS}, T^j_{MAS})$, $j = 1,2, \dots, k$, where for every $(u, v) \in E_i$, and its corresponding arc $(u^j, v^j) \in E^j_{MAS}$,

$$L(u, v) \geq \sum_{j=1}^{k} L\left(u^j, v^j\right) \geq eRU(u, v).$$

We call the entire RDLT R as a **separable RDLT** if its expanded vertex simplifications R_i is k_i-separable, $k_i \in \mathbb{N}$, $i \in \{1,2\}$, and every $(x, y) \in E_i$ has a corresponding $(x^j, y^j) \in E^j_{MAS}$ of some R^j_{MAS}.

Definition 9 means that the consolidated set of MAS of each R_i would have each component $(u, v) \in E_i$ to have its (derived) L-value, in particular its expanded reusability $eRU(u, v)$, configured in a way that parts of it are distributed to each L-value of its corresponding component $(u^j, v^j) \in E_{MAS}$ where the sum of the latter does not exceed $L(u, v)$ of R_i, regardless of the fact that (u, v) is a shared component by these activities.

Finally, Definition 9 also provides that every arc in a R_i is involved in at least one of its set of MAS so the entire RDLT R is separable.

Referencing the expanded vertex simplification R_1 in Figure 3 and its respective MAS in Figure 5, item (a), R^1_{MAS} to R^3_{MAS}, R_1 is separable by Definition 9. For example, (x_1, x_3) of R_1 has $L(x_1, x_3) = 5$, where its set of corresponding arcs in R^1_{MAS}, R^2_{MAS}, and R^3_{MAS} (also labelled with (x_1, x_3)) have their L-values totalling to $2(< 5)$, i.e. $L(x_1, x_3)$ of $R^1_{MAS} + L(x_1, x_3)$ of $R^2_{MAS} = 1 + 1 = 2$. Note that (x_1, x_3) is not part of MAS R^3_{MAS}. Meanwhile, for the abstract arc (x_3, x_4) of R_1 in Figure 3 that corresponds to the path $(x_3, x_2), (x_2, x_4)$, its L-value in R_1 is 1; while its corresponding arcs $(x_3, x_4)_1$ in R^1_{MAS}, R^2_{MAS}, and R^3_{MAS} have their sum of L-values equal to 1 (< 2). In both arcs, $eRU(x_1, x_3) = 1$ and $eRU(x_3, x_4)_1$ are equal to 1, thus, $(5 \geq 2 \geq 1)$ and $(2 \geq 1 \geq 1)$ (as per Definition 9, respectively. If we look at the relationship of R_i and its MAS, substructures that involve at least one loop in R_i must not be part of more than 1 MAS for R_i, otherwise, R_i is not separable. That is, by Definition 8, an arc (x, y) of R_i would have its corresponding arcs in each of its MASs set to its $L(x, y)$, hence, for n MAS of R_i, where $n > 1$, we have $L(x, y) \not\geq n * L(x, y) \not\geq eRU(x, y)$ in Definition 9.

Additionally, R_2 is also separable based on the L-values of its components with relation to R^1_{MAS} and R^2_{MAS} in item (b) of Figure 5.

Theorem 3 poses the relationship of separable and impedance-free RDLTs.

Theorem 3. R_i is separable if and only if R is impedance-free, i.e. for every pair of its MAS $R^1_{MAS} = (V^1_{MAS}, E^1_{MAS}, T^1_{MAS}, M^1_{MAS})$ and $R^2_{MAS} = (V^2_{MAS}, E^2_{MAS}, T^2_{MAS}, M^2_{MAS})$ that induce their corresponding maximal activities S^1_{max} and S^2_{max}, respectively, S^1_{max} and S^2_{max} do not impede each other.

Proof. We first prove that if R_i is separable, then every such S^1_{max} and S^2_{max} do not impede each other.

Using Definition 9, we know that every such pair of MAS R^1_{MAS} and R^2_{MAS} of R_i follows the minimal contraction paths P and P', respectively, where either of the following cases holds:

- **Case 1: if P and P' intersect at some $y \in E_i$, but with no common arc between them.**

  Suppose that y forms a JOIN in R_i, i.e. $(u, y), (v, y)$, where at least one of these components uses at least one arc $(u^1, y^1) \in E^1_{MAS}$ in P, while at least one other arc $(v^2, y^2) \in E^2_{MAS}$, where these components correspond to the JOIN $(u, y), (v, y) \in E_i$, respectively. Since P and P' are minimal contraction paths from the source to the sink vertices of R_i, we know that $C(u^1, y^1) = C(v^2, y^2)$. With these, if we simulate the maximal activity S^1_{max} derivable from R^1_{MAS} through P, (u, y) will not be impeded by (v, y) so that y is reachable by the algorithm in terms of their C-values. Thus, the simulation of S^1_{max} can go on for every such JOIN that goes through/intersects with P until the last reachability profile of S^1_{max}. The same is true if we simulate the maximal activity S^2_{max} derivable from R^2_{MAS} through P'. Thus, we would be able to prove this case by building the looped RDLT R_{loop} of R_i as per Definition 4. Thereafter, we build a composite activity S'' from using S^1_{max} and S^2_{max} of these MAS, and therefore show that S^1_{max} and S^2_{max} do not impede each other.

- **Case 2: if P and P' share a common arc (x, y) in E_i.**

  To prove this case, we look at the execution of the activities S^1_{max} and S^2_{max} that are derivable from MAS R^1_{MAS} and R^2_{MAS} that use P and P', respectively. Let $(x^j, y^j) \in E^j_{MAS}$ be the arc in R^j_{MAS} that corresponds

to $(x,y) \in E_i$, $j = 1,2$. With this, we have the following sub-cases to prove:

– **if (x,y) is not part of at least one loop in R_i.**

For this case, we start by building the looped RDLT R_{loop} of R_i as per Definition 4 using R^1_{MAS} and R^2_{MAS} as in Case 1 above.

Similarly, we construct a composite activity S'' using S^1_{max} and S^2_{max} for R_{loop}. Suppose we simulate S^1_{max} in R_{loop} from its source until we use all its components. Since (x^1,y^1) is not part of a loop, its L-value is 1 in R^1_{MAS}. Note that S^1_{max} uses (x^1,y^1) exactly once. Then, the simulation of S'' moves into simulating S^2_{max} through the corresponding components in R_{loop} until the sink is reached. In the same manner, $L(x^2,y^2) = 1$ in R^2_{MAS}, as well as S'' uses its corresponding arc exactly once. Since R_i is separable, we know that $L(x,y) \geq 2$. Moreover, $L(x^1,y^1) + L(x^2,y^2) \geq eRU(x,y)$. Thus, we would be able to simulate the entire composite activity S'' since $L(x,y)$ allows it so.

– **if (x,y) is part of at least one loop in R_i.**

For this case, we show that R_i cannot be separable if there exists a shared (x,y) between R^1_{MAS} and R^2_{MAS} where (x,y) is part of at least one loop in R_i.

We prove this by contradiction. That is, R_i is separable if (x,y) is a shared component of S^1_{max} and S^2_{max} derivable from R^1_{MAS} and R^2_{MAS}, respectively, where (x,y) is part of at least one loop in R_i.

Assume that R_i is separable. Furthermore, let (x,y) be a (P)CA of R_i.

Let $(x^1,y^1) \in E^1_{MAS}$ and $(x^2,y^2) \in E^1_{MAS}$ both correspond to the shared (P)CA $(x,y) \in E_i$. Since R_i is separable, we know that $L(x,y) \geq L(x^1,y^1) + L(x^2,y^2)$ (Definition 9). However, using Definition 8, we know that $L(x^1,y^1) = L(x^2,y^2) = L(x,y)$. Note that by the construction of EVSA of R_i, $L(x,y) = eRU(x,y)$. With this, we see that $L(x,y) \geq 2L(x,y)$ which is a contradiction, thus our claim that R_i is separable is not true.

In a similar light, we will also prove that if S^1_{max} and S^2_{max} share (x,y), where (x,y) is part of at least one loop, there is no composite activity S'' that can be constructed from these activities, i.e. S^1_{max} and S^2_{max} impede each other. We prove this again by contradiction. That is, suppose that a composite activity S'' exists from S^1_{max} and S^2_{max} that shares (x,y) of R_i where (x,y) is part of at least one loop.

Let c_1 by the cycle that involves (x^1,y^1) in R^1_{MAS}. Similarly, let c_2 by the cycle that involves (x^2,y^2) in R^2_{MAS}. Since S^1_{max} is a maximal activity derivable from R^1_{MAS}, it would therefore use the maximum number of times that (x^1,y^1) is usable through c_1, i.e. $L(x^1,y^1) = eRU(x,y)$. By this time, we have already exhausted the usability of (x,y) such that S^2_{max} cannot be simulated in R_{loop} when we enter c_2 to use/reuse (x^2,y^2). Thus, S'' is not realizable as a composite activity for the source and sink of R_{loop}, i.e. a contradiction to our claim.

Next, we prove that if for every pair of MAS R^1_{MAS} and R^2_{MAS} of R_i, with their maximal activities S^1_{max} and S^2_{max}, S^1_{max} and S^2_{max} do not impede each other, then R_i is separable.

To prove this case, we look at every common arc $(x,y) \in E_i$ that has corresponding arcs $(x^j,y^j) \in E^j_{MAS}$, $j = 1,2$, that is common between R^1_{MAS} and R^2_{MAS}. Since each R^j_{MAS} is a maximal activity structure, we know that $L(x^j,y^j)$ is either 1 or $L(x,y)$. In either case, we know that S^1_{max} and S^2_{max} do not impede each other. In other words, if we now include all of the maximal activities R^j_{MAS} of R_i, we can build a composite activity $C = R^1_{MAS} \oplus R^2_{MAS} \oplus \dots \oplus R^{k_i}_{MAS}$, $k_i \in \mathbb{N}$, that can be simulated in the looped RDLT of R_{loop} of R_i. If we look at the number of times $n \in \mathbb{N}$ that (x^j,y^j) is used in C, we see that n is either (a) exactly the number of maximal activities that use (x,y) (i.e. for the first case – $(L(x^j,y^j) = 1)$, and with $eRU(x,y) = 0$, or (b) equal to $eRU(x,y)$ since there is exactly one MAS R^j_{MAS}, that use (x,y) (i.e. for the second case – $L(x^j,y^j) = eRU(x,y))$, where $1 \leq j \leq k_i$. In both cases, $L(x,y) \geq n \geq eRU(x,y)$. With this, we have proved that R_i is separable.

□

Corollary 1. Given two activities S and S' of R_i, let $ActGr(S)$ and $ActGr(S')$ be their maximal activity groups where $ActGr(S) \neq ActGr(S')$.

For every pair of activities $A \in ActGr(S)$ and $B \in ActGr(S')$ in R, R_i is separable if and only if A and B do not impede each other.

Proof. Follows from Theorem 3. □

Theorem 4. Given a set of MAS of size k_i for an expanded level-i vertex-simplification R_i of an RDLT R, $i \in \{1,2\}$, it takes $O(|E|^3)$ time and space complexity to determine if R is separable.

Proof. We initially express the problem of determining a distribution of the L-value of each arc in R_i from among its given set of MAS. This can be expressed as a system of linear equations, as shown below. Thereafter, we solve this system such that we are able to realize if the input RDLT R is indeed separable.

$$\begin{pmatrix} L(e_{1,1}) & L(e_{1,2}) & \dots & L(e_{1,k_i}) \\ L(e_{2,1}) & L(e_{2,2}) & \dots & L(e_{2,k_i}) \\ \vdots & \vdots & \dots & \vdots \\ L(e_{|E_i|,1}) & L(e_{|E_i|,2}) & \dots & L(e_{|E_i|,k_i}) \end{pmatrix} \begin{pmatrix} I^1(e_r) \\ I^2(e_r) \\ \vdots \\ I^{k_i}(e_{|E_i|}) \end{pmatrix} \leq \begin{pmatrix} L(e_1) \\ L(e_2) \\ \vdots \\ L(e_{|E_i|}) \end{pmatrix},$$

where $L(e_{r,s}) \in \{0,1,L(e_r)\}$, $1 \leq r \leq |E_i|$, $1 \leq s \leq k_i$, $L(e_r) \geq 1$, and

$$I^s(e_r) = \begin{cases} 1, & \text{if } \exists (u,v) \in E^s_{MAS} \text{ where } (u,v) = e_r \\ 0, & \text{otherwise.} \end{cases}$$

The vector $[I^1(e_r), I^2(e_r), \dots, I^s(e_r), \dots, I^{k_i}(e_r)]$ accounts for the inclusion or exclusion of e_r in the pruned contraction path

from the source to the sink of with respect to the MAS R^s_{MAS} of R_i, $1 \leq s \leq k_i$. Whenever $e_r \in E_i$ has its corresponding arc $e_{r,s} \in E^s_{MAS}$ included in R^s_{MAS}, i.e. $I^s(e_r) = 1$, its $L(e_{r,s})$ is either 1 or $L(e_r)$ as per Definition 8, otherwise 0(i.e. excluded from R^s_{MAS}, with $I^s(e_r) = 0$).

With these, our system of linear equations accounts for the connectivity, L-values, and C-values of R_i and its set of MAS. Thus, a solution to this system establishes a set of L-values of the arcs for each MAS of R_i that establishes the separability of R_i as per Definition 9.

Solving a system of linear equations is known to be polynomial-time and -space solvable, i.e. $O(n^3)$, where n is the number of linear equations involved in the problem(Golub and Van Loan 1996). For our second problem of determining if R is separable, n corresponds to $|E_i|$ which is $O(|E|)$. Thus, this problem entails $O(|E|^3)$ time and space complexity.

■

From Theorem 4, we were able to establish that should a system designer start with a known k_i-sized set of MAS for R_i, realizing if each R_i is separable runs in polynomial time. Regarding the hardness of determining whether an RDLT R is separable, a solution to this problem would entail that we first determine the set of MAS for each of its expanded vertex simplifications R_1 and R_2. Thereafter, we can determine if R is separable by looking at the relationships of their L-values. In particular, the first problem may need us, at worst, to enumerate all possible (elementary) paths from the source to the sink vertex of each R_i. This takes an exponential number of such paths.

Thus, this would also be the number of contraction paths that the MCA can establish for R_i. Thus, our initial problem in identifying all the MAS is reducible to a well-known NP-hard problem(Cormen et al. 2009) of such path enumeration. Given this insight, we pose the following conjecture:

Conjecture 1. Determining if an input RDLT R is separable is an NP-hard problem.

3. Results and Discussion

Through this research, we were able to establish new concepts and techniques to separate an input RDLT into coherent substructures that are usable to represent or generate sets of activities by some group similarities; help in targeted model decomposition or transformation; facilitate efficient workflow analysis; and provide insights on how these are useful in building parallel profiles in RDLTs. We were able to also establish representatives of each of these groups for examining the impact of shared components and their presence in cycles within and across such groups. In summary, we established and proved the following results as shown in Table 1.

Table 1: Summary of the established definitions, algorithms, and proved theorems of this paper.

#	Result	Reference
A. Introduced Definitions		
1	Maximal Activity	Definition 3
2	Looped RDLT	Definition 4
3	Composite Activity	Definition 5
4	Non-impeding Activities	Definition 6
5	Impedance-free RDLT	Definition 7
6	Maximal Activity Structure (MAS)	Definition 8
7	Separable RDLT	Definition 9
B. Algorithm		
1	The Modified Contraction Algorithm (MCA)	Algorithm 1
C. Proved Theorems		
1	Impedance between a maximal activity ar activity (with a loop) that belong to the same activ	Theorem 1
2	MCA produces a minimal contraction structure in R_i	Lemma 1
3	Time and space complexity of MCA	Lemma 2
4	Each MAS in R_i generates a maximal activity therein in R_i	Theorem 2
5	Relationship of Impedance-free and separable RDLTs	Theorem 3
6	Separable RDLTs and the impedance between their MASs	Corollary 1
7	Time and space complexity of verifying separability of RDLTs given a set of MAS for its set of expanded vertex simplifications	Theorem 4

Some Notes on RBS and Parallel Activities

As we have established above in our results, the separability of RDLTs mainly focuses on the feasibility of fragmenting them through their C- and L-attributes as reflected in the looped versions of the vertex simplifications of R. That is, the C-values drive this fragmentation by imposing that the necessary set of conditions for reachability is accounted for in entire fragments. Meanwhile, L-values drive it by imposing that the arcs inside each fragment are reusable despite that such arcs appear in separate fragments due to multiple (maximal) activities sharing them.

With our use of looped RDLTs to sequentially simulate sets of activities to verify impedance with respect to each other, we have momentarily excluded the case wherein such activities may affect each other in terms of the M-attribute, i.e. their use of RBS. That is, we pose as a future research endeavor the parallelized simulation of activities, alongside their sharing and use of RBS components in overlapping time intervals. For this case, resets may cause the cancellation of ongoing processes in at least one of these activities, thereby such activity never completes and/or may trigger unexpected behavior in R. As a preliminary step to manage these issues on parallel activities, we offer the initial definition below.

Definition 10. (Reset-safe RDLT)

An impedance-free RDLT R is **reset-safe** if for every pair of maximal activities S and S' of the input-output pair $[s, f]$, either of the following holds:

- if S and S' are checking/traversing arcs inside an RBS G of R, both will exit an outbridge of a vertex in G at the same time,
- if S and S' do not use G at the same time, or
- if S' and S' do not use G at all.

In addition to Definition 10, we have the following conjecture with regard to the relationship of impedance-free and separable RDLTs in the context of parallel activities:

Conjecture 2: R is reset-safe if R is separable and if for every pair of MAS R_{MAS} and R'_{MAS} of R_i of R, the following hold:

1. for every non-RBS $(x, y) \in E$, every path leading from (x, y) to an in-bridge of an RBS G is an elementary path,

2. for every out-bridge $(u, v) \in E$ of $u \in V$ of G, and its corresponding arc $(u_1, v_1) \in E_{MAS}$ $((u_2, v_2) \in$

E'_{MAS}), the number of contraction steps from the s_1 to v_1 in R_{MAS} is equal to the number of contraction steps from s_2 to v_2 in R'_{MAS}, where s_1 and s_2 are both sources or both targets of looping arcs in R_{MAS} and R'_{MAS}, respectively.

4. Conclusions and Future Work

Since RDLTs were introduced in (Malinao 2017), there have been approaches in subsequent literatures that extract sub-profiles in such models either via model decomposition or model transformations. These approaches aim to extract smaller sets of information from an input RDLT to help generate simpler and/or smaller models. These were previously done with manual labor and human intervention to choose a substructure or behavior (via activity extraction) from the input RDLT. Through the concepts and techniques of our study, we are now able to isolate these substructures and sets of behaviors through the extraction of MAS and their respective maximal activity groups. These MAS and maximal activity groups, and particularly their maximal activities, can be then used to analyze the RDLT in a more efficient manner rather than by looking at individual substructures or activities.

One of the most important contributions of this paper is that the results herein usher the concept of shared components between and among different (maximal) activities in $R(R_i)$. It also offers a view of the impact of having shared components that are involved in loops. We can realize that this can result in deadlocks in workflows despite a sequential yet consecutive execution of activities that share such components. That is, activity completion can never be accomplished for at least one of these activities. This information can now serve as a valuable input to the goal of parallelization of activities in RDLTs that represent real-world systems with parallelizable structures and behaviors.

Lastly, we have also provided in this paper a jumpstart to modeling and analyzing RDLTs with parallel profiles that include reset structures and behaviors. With this, we foresee and recommend as future work to extend the activity extraction algorithm in literature as a parallel algorithm. This would also mean opening the field of multidimensional workflow modeling to properties related to having parallel activities such as generalized soundness and its weakened notions(van der Aalst 1996); benchmarking and transformations of models that exhibit maximally parallelizable profiles; among others.

CONFLICT OF INTEREST

The authors of this paper do not have any financial, personal, or professional relationship with other individuals or organizations that could be construed as conflict of interest in accomplishing this study.

CONTRIBUTIONS OF INDIVIDUAL AUTHORS

Dr. Malinao is the main contributor with respect to the formulation of the concepts, techniques, and proofs of this paper. Meanwhile, Dr. Juayong ensured and verified the thoroughness, completeness, and correctness of these contents.

REFERENCES

(Calvo and Malinao 2023) Calvo G, Malinao J. Mapping Hierarchies and Dependencies from Robustness Diagram with Loop and Time Controls to Class Diagram. In: Kabassi K, Mylonas P, Caro J. (eds) Novel & Intelligent Digital Systems: Proceedings of the 3rd International Conference (NiDS 2023). Lecture Notes in Networks and Systems. Vol. 783. Cham: Springer, 2023:23-42.

(Cormen et al. 2009) Cormen T, Leiserson C, Rivest R, and Stein C. Introduction to Algorithms. 3rd Edition. MIT Press, 2009. ISBN-13: 978-0262033848.

(Delos Reyes et al. 2018) Delos Reyes R, Agnes K, Malinao J, Juayong RA. Matrix Representation and Automation of Verification of Soundness of Robustness Diagram with Loop and Time Controls. Proceedings of the Workshop on Computation, Theory, and Practice (WCTP), 2018.

(Eclipse and Malinao 2023a) Eclipse K, Malinao, J. Model Decomposition of Robustness Diagram with Loop and Time Controls to Sequence Diagrams. In: Kabassi K, Mylonas P, Caro J. (eds) Novel & Intelligent Digital Systems: Proceedings of the 3rd International Conference (NiDS 2023). Lecture Notes in Networks and Systems. Vol. 784. Cham:Springer, 2023:40-54.

(Eclipse and Malinao 2023b) Eclipse, K, Malinao, J. Model Decomposition of Robustness Diagram with Loop and Time Controls to Sequence Diagrams using Activity Groups. In: Pre-proceedings of the Workshop on Computation: Theory and Practice(WCTP), Hokkaido, Japan. 2023.

(Golub and Van Loan 1996) Golub G, Van Loan C. Matrix Computations. 3rd edition. Johns Hopkins University Press. 1996. ISBN 978-0-8018-5414-9.

(Hauser et al. 2006) Hauser R, Friess M, Kuster J, Vanhatalo J. Combining Analysis of Unstructured Workflows with Transformation to Structured Workflows. In: Proceedings of the 2006 10th IEEE International Enterprise Distributed Object Computing Conference (EDOC'06), 2006. ISBN:0-7695-2558-X.

(Ko et al. 2009) Ko R, Lee S, Lee EW. Business process management (BPM) standards: a survey. Business Process Management Journal. 2009; 15 (5):744-491.

(Kotb and Baumgart 2005) Kotb YT, Baumgart AS. An extended Petri net for modeling workflow with critical sections. IEEE International Conference on e-Business Engineering (ICEBE'05), Beijing, China, 2005: 134-141.

(Leopold et al. 2015) Leopold H, Mendling J, Gunther O. What we can learn from quality issues of BPMN models from industry, IEEE Software 33(4). 2015;1-9.

(Lopez et al. 2020) Lopez JCL, Bayuga, MJ, Juayong RA, Malinao J, Caro J, Tee M. Workflow models for integrated disease surveillance and response systems, Theory and Practice of Computation, London: Taylor and Francis Group, 2020.

(Malinao 2017) Malinao J. On building multidimensional workflow models for complex systems modeling. Dissertation, Technische Universität Wien. reposiTUm, 2017.

(Malinao and Juayong 2023a) Malinao J, Juayong, RA. Reset Profiles and Classical Soundness in Robustness Diagrams with Loop and Time Controls. Pre-proceedings of the Workshop on Computation: Theory and Practice(WCTP), Hokkaido, Japan, 2023.

(Malinao and Juayong 2023b) Malinao J, Juayong RA. Classical Soundness in Robustness Diagram with Loop and Time Controls. Philippine Journal of Science. Vol. 152(6B), 2023: 2327–2342.

(Malinao et al. 2013) Malinao J, Lozano LM, Pascua S, Chua RB, Magboo MS, Caro J. A Metric for User Requirements Traceability in Sequence, Class Diagrams, and Lines-of-Code via Robustness Diagrams. Proceedings in Information and Communications Technology. Vol. 7, Springer, 2013: 50-63.

(Medeiros et al. 2005) Medeiros CB, Perez-Alcazar J, Digiampietri L, Pastorelo GZ Jr, Santanche A, Torres RS, Madeira E, Bacarin E. WOODSS and the Web: Annotating and Reusing Scientific Workflows. SIGMOD Record. 34, 2005:18-23.

(Sulla and Malinao 2023) Sulla, CN, and 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). Lecture Notes in Networks and Systems, vol 784. Cham: Springer, 2023:338-353.

(van der Aalst 1996) van der Aalst WMP. Structural Characterizations of Sound Workflow Nets. Computing Science Reports 96/23. Einhoven University of Technology, 1996.

(van der Aalst 2000) van der Aalst WMP. Workflow Verification: Finding Control-Flow Errors using Petri-net-based Techniques. In Business Process Management: Models, Techniques, and Empirical Studies. Springer Berlin Heidelberg, 2000:161-184

(van Hee et al 2003) van Hee K, Sidorova N, Voorhoeve M. Soundness and Separability of Workflow Nets in the Stepwise Refinement Approach. In: van der Aalst, W.M.P., Best, E. (eds) Applications and Theory of Petri Nets 2003. ICATPN 2003. Lecture Notes in Computer Science. Vol 2679. Springer Berlin Heidelberg, 2003.