NAIST-IS-DD2221428

Doctoral Dissertation

On the Applications of the Recursive Property of the Zero-Suppressed Binary Decision Diagram

Brian Godwin S. Lim

September 25, 2025

Division of Information Science
Graduate School of Science and Technology
Nara Institute of Science and Technology

A Doctoral Dissertation
submitted to Graduate School of Science and Technology,
Nara Institute of Science and Technology
in partial fulfillment of the requirements for the degree of
Doctor of ENGINEERING

Brian Godwin S. Lim

On the Applications of the Recursive Property of the Zero-Suppressed Binary Decision Diagram[*]

Brian Godwin S. Lim

Abstract

The zero-suppressed binary decision diagram (ZDD) is a powerful data structure for the compact representation of sparse combinatorial families. Its recursive structure and effective reduction rules enable efficient diagram manipulation, family operations, and extraction of basic family information, making it widely applicable in discrete optimization. Given its utility and unique properties, this dissertation explores novel applications of the recursive property of the ZDD. Specifically, a framework bridging statistical moments and the ZDD is first introduced, facilitating the extraction of distributional properties from arbitrary families. A corresponding recursive algorithm is then developed and validated against three classical combinatorial problems, demonstrating orders-of-magnitude improvements in computational efficiency over conventional approaches. The recursive framework is further adapted to reliability engineering by introducing moment-based network reliability measures, called path survival reliabilities, which are efficiently calculated using the ZDD. Through theoretical discussions and experimental validation on three real-world water distribution networks, the new measures are demonstrated to be well-grounded in engineering principles, offering a more nuanced characterization of network reliability compared to conventional measures. Overall, the utility of the ZDD is extended by introducing efficient recursive algorithms for extracting non-trivial family information.

Keywords:

Combinatorial problems, discrete optimization, path survival reliabilities, recursive algorithms, statistical moments, zero-suppressed binary decision diagram

[*]Doctoral Dissertation, Graduate School of Science and Technology, Nara Institute of Science and Technology, September 25, 2025.

ゼロサプレス型二分決定グラフの再帰的性質の応用に関する研究 [*]

リム ブライアン ゴッドウィン シー

内容梗概

ゼロサプレス型二分決定グラフ（zero-suppressed binary decision diagram; ZDD）は、疎な組合せ族をコンパクトに表現するための強力なデータ構造である。その再帰的構造と効果的な簡約規則により、図の操作、族の演算、および基本的な族情報の抽出を効率的に行うことが可能であり、離散最適化分野において広く応用されている。本論文では、ZDD の有用性および特異な性質に着目し、その再帰的性質に基づく新たな応用可能性を探究する。具体的には、統計的モーメントとZDD を結びつける枠組みを導入し、任意の族から分布的性質を抽出する手法を提示する。対応する再帰的アルゴリズムを開発し、古典的な三つの組合せ問題に対して検証を行った結果、従来手法と比較して計算効率において桁違いの改善が得られたことを示す。さらに、この再帰的枠組みを信頼性工学に応用し、経路生存信頼度（path survival reliabilities）と呼ばれるモーメントに基づくネットワーク信頼性指標を導入する。これらの指標はZDD を用いて効率的に計算され、三つの実世界の配水ネットワークに対する理論的考察および実験的検証を通じて、工学的原理に基づいた妥当性を有し、従来の指標と比較してネットワーク信頼性をより精緻に特徴づけるものであることが示された。以上により、非自明な族情報を抽出するための効率的な再帰的アルゴリズムの導入を通じて、ZDD の応用可能性がさらに拡張されることが示された。

キーワード

組合せ問題、離散最適化、経路生存信頼度、再帰的アルゴリズム、統計的モーメント、ゼロサプレス型二分決定グラフ

[*] 奈良先端科学技術大学院大学 先端科学技術研究科 博士論文 2025 年 9 月 25 日

Contents

List of Figures

List of Tables

1. Introduction

1.1 Background

The zero-suppressed binary decision diagram (ZDD) is a data structure widely employed for the compact representation of families (sets) of sparse subsets [34]. Notably, it improves upon the original binary decision diagram (BDD) [4] by incorporating more effective reduction rules that better handle sparsity, leading to improved efficiency and unique family representation. Additionally, the recursive property of the ZDD also facilitates efficient diagram manipulation and family operations, such as obtaining the union and intersection of two diagrams [36], enabling it to represent more complex families. Furthermore, the number of represented subsets and the optimal subset with respect to a linear cost function may also be extracted from the diagram with ease.

The unique properties of the ZDD position it as an ideal data structure for combinatorial applications, as it enables the compact and efficient representation of families that may contain an exponential number of subsets. Despite this potential combinatorial explosion, the number of ZDD nodes required to represent the family is generally orders of magnitude smaller due to its effective reduction rules. This results in a significant reduction in memory requirements compared to explicitly enumerating and storing all subsets. Additionally, the recursive property of the ZDD also enables the extraction of basic family information through a single traversal of the diagram. This provides a substantial computational advantage over a naive brute-force approach.

Within the field of discrete optimization, the ZDD is widely used to represent all feasible solutions to a problem. It offers a good representation for solution sets in combinatorial problems, including the knapsack problem, the eight queens problem, and the knight's tour problem [36]. Its flexibility has also led to increasing adoption in diverse applications, such as polynomial manipulation [35], regular expressions [21], and Boolean satisfiability [29]. Moreover, the ZDD also finds practical applications in graph theory, particularly in arc routing [42, 44], network design [40, 41], and network reliability [27, 43], where subgraphs are uniquely represented by their corresponding edge sets [22, 26]. Recent studies also continue to explore the applications of the ZDD across various fields [3, 11, 13].

1.2 Research Objectives

Given the utility and unique properties of the ZDD, this dissertation focuses on developing recursive algorithms for extracting key information about the family represented by a (constructed) ZDD. Specifically, the objectives are as follows:

1. Leverage the ZDD and its recursive property to propose novel recursive algorithms for combinatorial applications;
2. Develop a framework to extract distributional properties, via statistical moments, from an arbitrary family represented by a ZDD; and
3. Craft new network reliability measures grounded in engineering principles while utilizing the ZDD for efficient calculation.

1.3 Overview

This dissertation is organized as follows. Chapter 2 provides a formal introduction to the ZDD and the frontier-based search (FBS) used to construct the diagram for graphs. Chapter 3 presents a recursive framework that bridges statistical moments and the ZDD, providing additional utility to the diagram. Chapter 4 then leverages the ZDD to develop novel moment-based network reliability measures, called path survival reliabilities, addressing challenges in reliability engineering. To conclude, Chapter 5 summarizes the dissertation and presents recommendations for future work.

2. Preliminaries

2.1 Zero-Suppressed Binary Decision Diagram

The mathematical definition of the zero-suppressed binary decision diagram (ZDD) is formally introduced below [34].

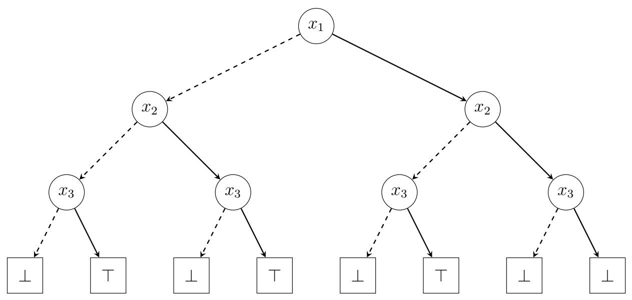
Definition 2.1 (Zero-Suppressed Binary Decision Diagram)**.** *Let U be a finite ordered universe where for $x_i, x_j \in U$, $x_i < x_j$ if and only if $i < j$. A reduced zero-suppressed binary decision diagram is a labeled directed acyclic graph satisfying the following properties:*

1. *There is exactly one node with an indegree of 0 called the root;*
2. *There are exactly two nodes with an outdegree of 0 called the 0-terminal and the 1-terminal, denoted by $\perp$ and $\top$, respectively;*
3. *Every non-terminal node has exactly two outgoing arcs, labeled by 0 and 1, called the 0-arc and 1-arc, respectively;*
4. *The destination of the 0-arc and the 1-arc of a non-terminal node is called the 0-child and 1-child, respectively;*
5. *Every non-terminal node is labeled by an element in U;*
6. *The label of a non-terminal node is strictly smaller than those of its children;*
7. *There is no node whose 1-child is the 0-terminal; and*
8. *There are no distinct nodes that have the same label, 0-child, and 1-child.*

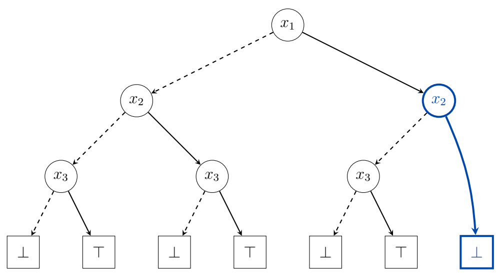
Properties 7 and 8 of Definition 2.1 above highlight the unique reduction rules of the ZDD, where the reduction complexity is proportional to the number of ZDD nodes [28]. Moreover, a reduced ZDD $\mathcal{D}$ may uniquely and efficiently represent any family of subsets $\mathcal{F}$ from the finite ordered universe U [34]. Accordingly, a route R from the root to the 1-terminal in the diagram corresponds to a subset $S \in \mathcal{F}$ if and only if for each $x \in S$, there exists a node labeled x whose 1-arc is in R [36].

As an illustration, consider the family $\mathcal{F} = \{\{x_1, x_3\}, \{x_2, x_3\}, \{x_3\}\}$ from the finite ordered universe $U = \{x_1, x_2, x_3\}$ as presented in Fig. 2.1. The binary decision tree representing $\mathcal{F}$ is presented in Fig 2.1a, where the nodes in the diagram are labeled with an element from U. Moreover, a 0-arc is represented by a dashed line while a 1-arc is represented by a solid line. Following Property

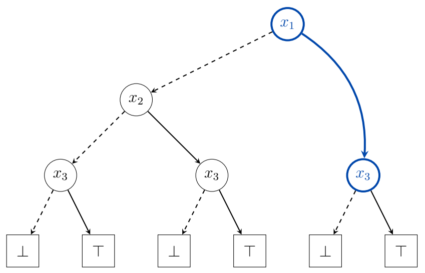
7, the rightmost node labeled x_3 is deleted since its 1-child is the 0-terminal $\perp$, obtaining the diagram in Fig. 2.1b. This process is repeated for the rightmost node labeled x_2, obtaining the diagram in Fig. 2.1c. Following Property 8, the three nodes labeled x_3 are merged since they share the same 0-child and 1-child, obtaining the diagram in Fig. 2.1d. Since both Properties 7 and 8 are satisfied, the diagram in Fig. 2.1e thus corresponds to the reduced ZDD representing $\mathcal{F}$.



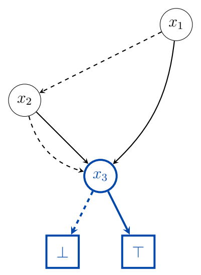
(a) The binary decision tree representing $\mathcal{F}$



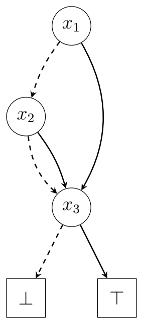
(b) Deleting node labeled x_3 with $\perp$ as the 1-child following Property 7



(c) Deleting node labeled x_2 with $\perp$ as the 1-child following Property 7



(d) Merging nodes labeled x_3 with the same 0 and 1-child following Property 8



(e) The reduced ZDD representing $\mathcal{F}$

Figure 2.1: Illustration for the ZDD representing $\mathcal{F} = \{\{x_1, x_3\}, \{x_2, x_3\}, \{x_3\}\}$

Furthermore, starting from the root labeled x_1 in Fig. 2.1e, one proceeds to the 0-child if the element is to be excluded from the subset and to the 1-child otherwise until the 1-terminal $\top$ is reached. For instance, taking the 0-arc of the root labeled x_1, the 1-arc of the node labeled x_2, and the 1-arc of the node labeled x_3 corresponds to the subset $\{x_2, x_3\} \in \mathcal{F}$. The theorem below then highlights the recursive structure of the ZDD [34].

Theorem 2.1. *Consider the zero-suppressed binary decision diagram $\mathcal{D}$ representing the family $\mathcal{F}$. The root r is either a terminal node or a non-terminal node.*

1. *If r is the* 0*-terminal, then $\mathcal{F} = \emptyset$, the empty family.*
2. *If r is the* 1*-terminal, then $\mathcal{F} = \{\emptyset\}$, the family containing the empty set.*
3. *If r is a non-terminal node, then it has two children. Suppose r_0 is the* 0*-child and r_1 is the* 1*-child of r. Denote the family with diagram rooted at r_i by $\mathcal{F}_i$, for $i = 0, 1$. If the root has label $x_r \in U$, then*

$$\mathcal{F} = \mathcal{F}_0 \cup \left(\mathcal{F}_1 \bowtie \{\{x_r\}\}\right), \tag{2.1}$$

where $\mathcal{A} \bowtie \mathcal{B} := \{A \cup B \mid A \in \mathcal{A}, B \in \mathcal{B}\}$.

From Theorem 2.1 above, subsets in $\mathcal{F}$ that do not contain x_r are connected to r through its 0-arc while subsets in $\mathcal{F}$ that do contain x_r are connected to r through its 1-arc. This may be written mathematically as

$$\mathcal{F}_0 = \{S \mid S \in \mathcal{F}, x_r \notin S\}, \tag{2.2}$$

$$\mathcal{F}_1 = \{S \setminus \{x_r\} \mid S \in \mathcal{F}, x_r \in S\}. \tag{2.3}$$

Moreover, the recursive property of the ZDD also facilitates the efficient execution of family operations [34]. In particular, consider two ZDDs $\mathcal{D}_1$ and $\mathcal{D}_2$ representing the families $\mathcal{F}_1$ and $\mathcal{F}_2$ with elements from the universes U_1 and U_2, respectively. Obtaining the number of subsets represented by $\mathcal{D}_1$, denoted by $|\mathcal{D}_1|$, or equivalently the number of subsets in $\mathcal{F}_1$, denoted by $|\mathcal{F}_1|$, has time complexity $\mathcal{O}\left(Z\left(\mathcal{D}_1\right)\right)$, where $Z\left(\mathcal{D}\right)$ denotes the number of ZDD nodes in $\mathcal{D}$. Obtaining the intersection $\mathcal{F}_1 \cap \mathcal{F}_2$ or the union $\mathcal{F}_1 \cup \mathcal{F}_2$ as a ZDD has time complexity $\mathcal{O}\left(Z\left(\mathcal{D}_1\right) \cdot Z\left(\mathcal{D}_2\right)\right)$. The time complexity of other common operations is provided in [47].

2.2 Frontier-Based Search

One method for constructing a ZDD representing a family of subgraphs is the frontier-based search (FBS) [26]. It constructs the diagram representing a class of subgraphs, such as paths, matchings, and trees, in a top-down breadth-first manner while employing node sharing and pruning for efficient construction. An outline of the construction algorithm for an s-t path is provided below.

Suppose $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ is a simple undirected graph without multi-edges, *i.e.*, each edge $e = \{v_e, v'_e\} \in \mathcal{E}$ is uniquely defined by a pair of distinct vertices $v_e, v'_e \in \mathcal{V}$. Denote $\mathcal{G}' = (\mathcal{V}', \mathcal{E}')$ a subgraph of $\mathcal{G}$, where $\mathcal{E}' \subseteq \mathcal{E}$ and $\mathcal{V}' = \bigcup_{e' \in \mathcal{E}'} e' \subseteq \mathcal{V}$ with the notation $\bigcup_{i=1}^{|\mathcal{E}'|} e_{a_i}$ representing the set of vertices to which at least one of $e_{a_1}, e_{a_2}, \ldots, e_{a_{|\mathcal{E}'|}}$ is incident. Hence, no vertex with degree 0 may exist in $\mathcal{G}'$. Consequently, each subgraph $\mathcal{G}'$ is uniquely determined by the subset of edges $\mathcal{E}'$. In the context of the ZDD, one sets $\mathcal{E}$ as the universe. The elements of $\mathcal{E}$ are then ordered such that $e_1 < e_2 < \cdots < e_{|\mathcal{E}|}$. With this, every node in the ZDD is labeled with an edge $e \in \mathcal{E}$.

FBS constructs the ZDD in a top-down breadth-first manner, creating and labeling nodes in the ZDD starting from the smallest element in the universe. Initially, the root is labeled e_1. For $i = 1, 2, \ldots, |\mathcal{E}| - 1$, a ZDD node labeled e_{i+1} is generated only after all ZDD nodes labeled e_i are generated. Moreover, the 0-arc and 1-arc of a ZDD node labeled e_i may only point to the 0-terminal, the 1-terminal, or a ZDD node labeled e_j, for $i < j$.

For systematic and efficient construction, information on subgraph specifications from previous edge selection is maintained on each ZDD node for vertices incident to both a processed edge and an unprocessed edge. This set of vertices is called the frontier [26]. By convention, $F_0 = F_{|\mathcal{E}|} = \emptyset$. For $j = 1, 2, \ldots, |\mathcal{E}| - 1$, the jth frontier is defined as

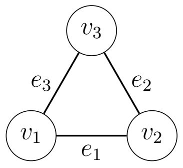
$$F_j \coloneqq \left(\bigcup_{i=1}^{j} e_i \right) \cap \left(\bigcup_{i=j+1}^{|\mathcal{E}|} e_i \right). \tag{2.4}$$

For vertices in the frontier, each ZDD node n maintains information on vertex degrees with an array $n.\texttt{deg}$. The partition of the frontier is also recorded on each ZDD node as $n.\texttt{comp}$, which determines if two vertices belong to the same connected component.

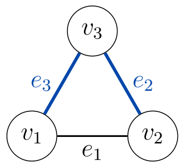
For a vertex $v \in \mathcal{V}$ entering the frontier, *i.e.*, the current edge is the smallest incident edge, one sets $n.\texttt{deg}[v] = 0$ and updates $n.\texttt{comp}$ to $n.\texttt{comp} \cup \{\{v\}\}$. When processing the 1-arc of a ZDD node labeled $e = \{v_e, v'_e\} \in \mathcal{E}$, if vertices v_e and v'_e already belong to the same component in $n.\texttt{comp}$, then the ZDD node points to the 0-terminal $\bot$ since a cycle has occurred. Otherwise, the degrees $n.\texttt{deg}[v_e]$ and $n.\texttt{deg}[v'_e]$ are incremented by one and the partitions of v_e and v'_e are joined in $n.\texttt{comp}$. Moreover, for a vertex $v \in \mathcal{V}$ leaving the frontier, *i.e.*, the current edge is the largest incident edge, $n.\texttt{deg}[v]$ is fixed and need not be updated further as it is independent of the remaining edges. Hence, if $v \in \{s, t\}$ and $n.\texttt{deg}[v] \neq 1$ or $n.\texttt{deg}[v] \notin \{0, 2\}$ for $v \notin \{s, t\}$, then an s-t path cannot be formed regardless of the selection from the remaining edges so the ZDD node points to the 0-terminal $\bot$. Nonetheless, if the final edge $e_{|\mathcal{E}|} \in \mathcal{E}$ is processed and all constraints are satisfied, the ZDD node then points to the 1-terminal $\top$.

Furthermore, during construction, the node sharing strategy is employed, merging two ZDD nodes $n \neq n'$ if $n.\texttt{deg} = n'.\texttt{deg}$ and $n.\texttt{comp} = n'.\texttt{comp}$. Node pruning, as described above, is also performed when an s-t path cannot be formed by adding a subset of the remaining edges. With node sharing and pruning, FBS efficiently constructs the ZDD representing the family of s-t paths. A complete discussion of the construction algorithm using recursive specifications [23] may be found in [26].

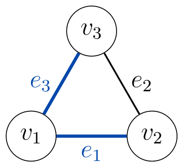
As an illustration, consider the sample graph $\mathcal{G}$ in Fig. 2.2a and the three paths $\mathcal{P}^1 = \{e_1, e_3\}$, $\mathcal{P}^2 = \{e_2, e_3\}$, and $\mathcal{P}^3 = \{e_3\}$ shown in Figs. 2.2b to 2.2d, respectively. The ZDD representing this family of paths $\mathcal{F}_{\mathcal{P}} = \{\mathcal{P}^1, \mathcal{P}^2, \mathcal{P}^3\} = \{\{e_1, e_3\}, \{e_2, e_3\}, \{e_3\}\}$ is presented in Fig. 2.2e, where the nodes in the diagram are labeled by an edge $e \in \mathcal{E}$ from graph $\mathcal{G}$. Similarly, a 0-arc is represented by a dashed line while a 1-arc is represented by a solid line. A traversal from the root labeled e_1 to the 1-terminal $\top$ thus corresponds to a path in the graph. Specifically, the 0-arc represents the exclusion of the edge while the 1-arc represents the inclusion of the edge. For instance, taking the 0-arc of the root labeled e_1, the 1-arc of the node labeled e_2, and the 1-arc of the node labeled e_3 corresponds to path $\mathcal{P}^2 = \{e_2, e_3\} \in \mathcal{F}_{\mathcal{P}}$.



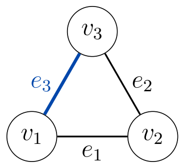
(a) A sample graph $\mathcal{G}$



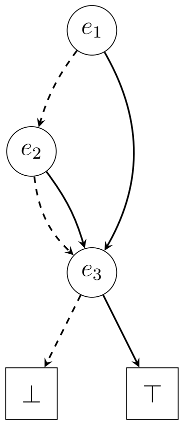
(c) Path $\mathcal{P}^2 = \{e_2, e_3\}$



(b) Path $\mathcal{P}^1 = \{e_1, e_3\}$



(d) Path $\mathcal{P}^3 = \{e_3\}$



(e) The ZDD representing $\mathcal{F}_{\mathcal{P}}$

Figure 2.2: Illustration for the ZDD representing $\mathcal{F}_{\mathcal{P}} = \{\mathcal{P}^1, \mathcal{P}^2, \mathcal{P}^3\}$

3. Statistical Moments

The unique properties of the ZDD, along with its efficient representational capabilities, have led to its extensive application in discrete optimization, where it can compactly represent families containing an exponential number of feasible solutions. Extracting distributional properties from these feasible solutions hence provides valuable insights into the characteristics of the family. Toward this objective, the chapter introduces a novel recursive algorithm for extracting statistical moments, such as the mean and variance, from an arbitrary family represented by a (constructed) ZDD.

Specifically, given a value mapping $v : U \to \mathbb{R}$ that assigns a value to every element in the universe U, the value of a subset $S \subseteq U$, denoted by $V(S)$, is first formulated as the sum of the values of its elements, similar to existing works [37]. Taking inspiration from the method of moments framework, for an arbitrary ZDD $\mathcal{D}$ representing the family $\mathcal{F}$, the kth moment of $\mathcal{F}$, denoted by $M^{(k)}(\mathcal{F})$, is calculated as the mean of the exponentiated subset values. Under this new moments framework for the ZDD, an algorithm leveraging its recursive property along with memoization techniques is developed. In particular, this novel recursive algorithm may efficiently extract the statistical moments of a family with just a single traversal of the corresponding ZDD. Its utility is then demonstrated with three classical problems in combinatorics—power sets, the knapsack problem, and paths in graphs—to highlight the orders-of-magnitude increase in computational efficiency relative to a naive brute-force approach.

3.1 Moments of a Family

As a prerequisite for defining the moments of a family $\mathcal{F}$, the value of a subset $S \subseteq U$ is first formally defined.

Definition 3.1 (Value of a Subset)**.** *Consider the subset* $S = \left\{x_{s_1}, x_{s_2}, \ldots, x_{s_{|S|}}\right\} \subseteq U$ *with a value mapping* $v : U \to \mathbb{R}$ *that assigns a real number to every element of* U*. The value of* S *is defined as*

$$V(S) \coloneqq \sum_{x \in S} v(x). \tag{3.1}$$

The value of a subset $S \subseteq U$ is the sum of the values of its elements. This formulation provides a good starting point for practical applications [37]. Nevertheless, other subset value formulations may be easily adapted to this framework. Following the method of moments framework, the moment of a family $\mathcal{F}$ is defined formally below.

Definition 3.2 (Moment of a Family)**.** *Consider the non-empty family $\mathcal{F}$ from the finite ordered universe U with a value mapping $V : 2^U \to \mathbb{R}$ that assigns a real number to every subset of U, where 2^U denotes the power set of U. The kth moment of family $\mathcal{F}$ is defined as*

$$M^{(k)}(\mathcal{F}) \coloneqq \frac{1}{|\mathcal{F}|} \sum_{S \in \mathcal{F}} [V(S)]^k . \tag{3.2}$$

The kth moment of a family $\mathcal{F}$ is the mean of the kth power of the values of each subset $S \in \mathcal{F}$. The two definitions above may then be combined compactly as follows.

Remark 3.1. *Consider the non-empty family $\mathcal{F}$ from the finite ordered universe U with a value mapping $v : U \to \mathbb{R}$ that assigns a real number to every element of U. The kth moment of family $\mathcal{F}$ is defined as*

$$M^{(k)}(\mathcal{F}) \coloneqq \frac{1}{|\mathcal{F}|} \sum_{S \in \mathcal{F}} \left[\sum_{x \in S} v(x) \right]^k . \tag{3.3}$$

The inner summation of Eqn. 3.3 above is raised to the kth power; hence, the expanded summation is a multinomial expansion with terms $v(x)$ for $x \in S$.

3.2 Methodology

Returning to the ZDD, consider the (constructed) diagram $\mathcal{D}$ representing the family $\mathcal{F}$ from the universe U rooted at node r. For $k \geq 0$, denote by $\texttt{val}^{(k)}$ a value assigned to each ZDD node given by

$$\texttt{val}^{(k)}(r) \coloneqq \sum_{S \in \mathcal{F}} \left[\sum_{x \in S} v(x) \right]^k \tag{3.4}$$

$$= \sum_{S\in\mathcal{F}} \sum_{\substack{\sum_{x\in S} p(x)=k \\ p(x)\geq 0,\ \forall x\in S}} \frac{k!}{\prod_{x\in S} p(x)!} \prod_{x\in S} [v(x)]^{p(x)}, \tag{3.5}$$

where the inner summation of Eqn. 3.5 represents a multinomial expansion of power k with $p(x)$ as the power assigned to $v(x)$ in a given term.

Note that $\texttt{val}^{(0)}(r) = |\mathcal{F}|$, $\texttt{val}^{(k)}(\perp) = 0$ for all $k \geq 0$, $\texttt{val}^{(0)}(\top) = 1$, and $\texttt{val}^{(k)}(\top) = 0$ for all $k \geq 1$. Leveraging the recursive structure of the ZDD highlighted in Eqns. 2.1, 2.2, and 2.3, for a non-terminal node r with label $x_r \in U$ and i-child r_i representing the family $\mathcal{F}_i$, for $i = 0, 1$, $\texttt{val}^{(k)}(r)$ may be decomposed as

$$\texttt{val}^{(k)}(r) = \texttt{val}^{(k)}(r_0) + \sum_{p=0}^{k} \binom{k}{p} [v(x_r)]^p \, \texttt{val}^{(k-p)}(r_1). \tag{3.6}$$

Appendix A.1 presents the formal proof of the recursive formulation of $\texttt{val}^{(k)}(r)$. Using $\texttt{val}^{(k)}(r)$, it is easy to see that

$$M^{(k)}(\mathcal{F}) = \frac{\texttt{val}^{(k)}(r)}{\texttt{val}^{(0)}(r)}. \tag{3.7}$$

Hence, $\texttt{val}^{(k)}(r)$ may be interpreted as the moment-generating function for the family represented by the ZDD rooted at node r. Moreover, a single execution to obtain $\texttt{val}^{(k)}(r)$ for fixed k already contains all of the moments $M^{(k')}(\mathcal{F})$ for $k' \leq k$. This demonstrates the efficiency of the proposed methodology, as there is no need for multiple executions to obtain different orders of moments. Algorithm 3.1 presents the pseudocode for the proposed algorithm. For efficient evaluation, $\textsc{MomentsEval}(r, v, k)$ is cached to memory and computed only once for every combination of r, v, k.

In summary, the proposed algorithm recursively traverses a given (constructed) ZDD to extract the first k moments of the corresponding family. The efficiency of the algorithm is attributed to its recursive nature, allowing it to employ classical techniques from dynamic programming while also effectively handling non-linearities in the multinomial expansion. In particular, the time complexity for extracting the first k moments from the family $\mathcal{F}$ represented by the ZDD $\mathcal{D}$ is $\mathcal{O}\left(k^2 \cdot Z\left(\mathcal{D}\right)\right)$. Since k is usually small in practice and the number of ZDD

Algorithm 3.1 $\textsc{MomentsEval}(r,\ v,\ k)$

```
1:  Assume node $r$ has label $x_r \in U$
2:  if $r = \bot$ then
3:      return 0
4:  else if $r = \top$ and $k = 0$ then
5:      return 1
6:  else if $r = \top$ and $k > 0$ then
7:      return 0
8:  else
9:      moment ← $\textsc{MomentsEval}\left(r.\textsc{child}(0), v, k\right)$
10:     for $p \leftarrow 0, 1, \ldots, k$ do
11:         moment ← moment + nCr$(k, p)\times$
12:                     pow$\left(v(x_r), p\right) \times \textsc{MomentsEval}\left(r.\textsc{child}(1), v, k - p\right)$
13:     return moment
```

nodes is significantly less than the number of subsets it represents $Z(\mathcal{D}) \ll |\mathcal{F}|$ [28], the proposed algorithm offers orders-of-magnitude reduction in computational time relative to a naive brute force approach enumerating all subsets in $\mathcal{F}$. Nonetheless, it is worth noting that the proposed algorithm operates on a given (constructed) ZDD $\mathcal{D}$ and value mapping v. The construction of the diagram is beyond the scope of the algorithm since it requires a specialized algorithm for each application [13, 21, 26, 35, 40]. Consequently, the suitability of the proposed algorithm also depends on the suitability of the ZDD for a given problem.

3.3 Results

To illustrate the utility and efficiency of the proposed algorithm, consider three classical problems with known moments information. First, power sets are investigated as a trivial example of a discrete optimization problem. The knapsack problem is then explored as a combinatorial application. Finally, the calculation of moments is analyzed on paths in graphs.

Across all reported results, the ZDD representing the family described is first constructed. Afterward, Algorithm 3.1 is applied to the constructed diagram with the corresponding value mapping to extract the first k moments of the corresponding family. Finally, validation against known moments information is performed to ensure the correctness of the algorithm.

Concerning reproducibility, the C++ programming language, with version 9.4.0 of g++, is chosen for implementation. Fundamental diagram manipulation is carried out through the `TdZdd` library [23]. The complete code is available in the `TdZdd Utility` repository[1]. The reported results use a machine with the Ubuntu 22.04 Long Term Support operating system, the AMD EPYC$^{\text{TM}}$ 7402P processor, and a memory of 256 GB.

As the primary contribution is the proposed algorithm, the main results focus on the execution time of the experiments to highlight the utility and efficiency of the algorithm. Specifically, the time taken to construct the ZDD, the number of subsets represented by the ZDD, the number of ZDD nodes, the time taken, and the number of recursive calls using Algorithm 3.1, and the total memory required for ZDD construction and moments evaluation are presented for completeness. As a benchmark, the time taken following a naive approach for moments evaluation—brute-force enumeration of subsets in the family and subsequent computation of moments following Eqn. 3.2—is also presented since the exact moments information, via analytic or programmatic solutions, is often unavailable or impractical, but potentially feasible with the ZDD.

3.3.1 Power Sets

Consider the universe U containing n elements. The ZDD representing the power set of U may be constructed following [23]. For simplicity, suppose the value mapping $v : U \to \mathbb{R}$ assigns a value 1 to every element in U, *i.e.*, $v(x) = 1$ for every $x \in U$. The kth moment of the family $\mathcal{F} = 2^U$ can be computed analytically as

$$M^{(k)}(\mathcal{F}) = \frac{1}{2^n} \sum_{p=0}^{n} \binom{n}{p} p^k. \tag{3.8}$$

Table 3.1 presents the summary of the results for varying cardinalities n. In all cases, the moments are calculated until $k = 5$, where both the proposed algorithm and the analytical solution yield the same values. The results show

[1]Accessible at `https://github.com/briangodwinlim/TdZdd-Utility`.

Table 3.1: Summary of Results for Power Sets

n	ZDD Construction Time	$\lvert\mathcal{D}\rvert$	$Z(\mathcal{D})$	k	Naive Calculation Time	Proposed Calculation Time	Proposed Recursive Calls	Memory Used
12	< 0.01 s	4.1×10^3	12	1	< 0.01 s	< 0.01 s	60	4 MB
				2	< 0.01 s	< 0.01 s	108	4 MB
				3	< 0.01 s	< 0.01 s	168	4 MB
				4	< 0.01 s	< 0.01 s	240	4 MB
				5	< 0.01 s	< 0.01 s	324	4 MB
16	< 0.01 s	6.6×10^4	16	1	0.03 s	< 0.01 s	80	4 MB
				2	0.03 s	< 0.01 s	144	4 MB
				3	0.03 s	< 0.01 s	224	4 MB
				4	0.04 s	< 0.01 s	320	4 MB
				5	0.04 s	< 0.01 s	432	4 MB
20	< 0.01 s	1.0×10^6	20	1	0.60 s	< 0.01 s	100	4 MB
				2	0.68 s	< 0.01 s	180	4 MB
				3	0.72 s	< 0.01 s	280	4 MB
				4	0.71 s	< 0.01 s	400	4 MB
				5	0.76 s	< 0.01 s	540	4 MB
24	< 0.01 s	1.7×10^7	24	1	11.35 s	< 0.01 s	120	4 MB
				2	12.03 s	< 0.01 s	216	4 MB
				3	12.46 s	< 0.01 s	336	4 MB
				4	12.96 s	< 0.01 s	480	4 MB
				5	13.48 s	< 0.01 s	648	4 MB
28	< 0.01 s	2.7×10^8	28	1	209.89 s	< 0.01 s	140	4 MB
				2	217.27 s	< 0.01 s	252	4 MB
				3	224.77 s	< 0.01 s	392	4 MB
				4	232.45 s	< 0.01 s	560	4 MB
				5	242.49 s	< 0.01 s	756	4 MB
32	< 0.01 s	4.3×10^9	32	1	> 1 h	< 0.01 s	160	4 MB
				2	> 1 h	< 0.01 s	288	4 MB
				3	> 1 h	< 0.01 s	448	4 MB
				4	> 1 h	< 0.01 s	640	4 MB
				5	> 1 h	< 0.01 s	864	4 MB

that a ZDD with just 32 nodes may represent a family containing $2^{32} \approx 4.3 \times 10^9$ subsets, highlighting the efficiency of the ZDD. Furthermore, the proposed algorithm is able to extract the moments from the families in less than 0.01 seconds, in stark contrast to the naive approach, which required significantly more computational time for the same task. This underscores the scalability of the proposed algorithm, offering an exponential reduction in computational time. These results demonstrate the utility and efficiency of the proposed algorithm with respect to conventional methods.

3.3.2 Knapsack Problem

Next, consider the 0-1 knapsack problem concerning the set U containing n items with corresponding weights $w(x) > 0$ for every $x \in U$. A subset of items $S \subseteq U$ is valid if and only if the total weight of the items in S does not exceed a given capacity C. Mathematically,

$$\sum_{x \in S} w(x) \leq C. \tag{3.9}$$

The ZDD representing the family of valid subsets of a knapsack problem may be constructed following [23]. Moreover, suppose a value mapping $v : U \to \mathbb{R}$ that assigns a reward to every item in U.

The knapsack problem is traditionally solved using dynamic programming to obtain the optimal subset S^* that maximizes the total reward. This approach may be easily modified to obtain the kth moment of the total rewards across all valid subsets as presented in the function `dp` in Appendix A.2. It is worth noting that the `@memoize` decorator is used to handle the memoization procedure of dynamic programming.

Table 3.2 presents the summary of the results for set U of varying sizes n. For simplicity, the weight w and value mapping v are randomly assigned integers from 1 to 10 with a fixed capacity $C = 5n$. Similarly, the moments are calculated until $k = 5$ with the dynamic programming approach validating the correctness of the proposed algorithm. The results show that a ZDD with just 175 nodes may represent a family containing 4.3×10^9 subsets satisfying a given constraint, highlighting the utility of the ZDD in combinatorial problems. Additionally, the proposed algorithm extracts the moments from the families in less than 0.01 sec-

Table 3.2: Summary of Results for the Knapsack Problem

n	ZDD Construction Time	$\lvert\mathcal{D}\rvert$	$Z(\mathcal{D})$	k	Naive Calculation Time	Proposed Calculation Time	Proposed Recursive Calls	Memory Used
12	< 0.01 s	4.1×10^3	12	1	< 0.01 s	< 0.01 s	60	4 MB
				2	< 0.01 s	< 0.01 s	108	4 MB
				3	< 0.01 s	< 0.01 s	168	4 MB
				4	< 0.01 s	< 0.01 s	240	4 MB
				5	< 0.01 s	< 0.01 s	324	4 MB
16	< 0.01 s	6.6×10^4	16	1	0.04 s	< 0.01 s	80	4 MB
				2	0.03 s	< 0.01 s	144	4 MB
				3	0.04 s	< 0.01 s	224	4 MB
				4	0.04 s	< 0.01 s	320	4 MB
				5	0.04 s	< 0.01 s	432	4 MB
20	< 0.01 s	1.0×10^6	62	1	0.61 s	< 0.01 s	310	4 MB
				2	0.64 s	< 0.01 s	558	4 MB
				3	0.66 s	< 0.01 s	868	4 MB
				4	0.70 s	< 0.01 s	1.2×10^3	4 MB
				5	0.73 s	< 0.01 s	1.7×10^3	4 MB
24	< 0.01 s	1.7×10^7	105	1	11.84 s	< 0.01 s	525	4 MB
				2	12.41 s	< 0.01 s	945	4 MB
				3	12.89 s	< 0.01 s	1.5×10^3	4 MB
				4	13.44 s	< 0.01 s	2.1×10^3	4 MB
				5	13.91 s	< 0.01 s	2.8×10^3	4 MB
28	< 0.01 s	2.7×10^8	56	1	217.12 s	< 0.01 s	280	4 MB
				2	225.78 s	< 0.01 s	504	4 MB
				3	233.72 s	< 0.01 s	784	4 MB
				4	241.68 s	< 0.01 s	1.1×10^3	4 MB
				5	249.82 s	< 0.01 s	1.5×10^3	4 MB
32	< 0.01 s	4.3×10^9	175	1	> 1 h	< 0.01 s	875	4 MB
				2	> 1 h	< 0.01 s	1.6×10^3	4 MB
				3	> 1 h	< 0.01 s	2.5×10^3	4 MB
				4	> 1 h	< 0.01 s	3.5×10^3	4 MB
				5	> 1 h	< 0.01 s	4.7×10^4	4 MB

onds, achieving a reduction in computational time by several orders of magnitude compared to the naive approach. The number of recursive calls further confirms the computational efficiency of the proposed algorithm, requiring significantly fewer operations than the total number of subsets in the family. These findings demonstrate the utility of the algorithm in practical combinatorial problems.

The example above highlights a potential application of the proposed algorithm in determining the distribution of solutions for a given discrete optimization problem. By calculating the moments of the value V assigned to candidate solutions, valuable insights into their distribution with respect to the value function may be obtained. This information may be further analyzed in conjunction with heuristic approaches, facilitating an exploration of the relative differences between heuristic solutions and the optimal solution. Such analysis may then enhance the understanding of solution quality and guide the development of more effective heuristics.

3.3.3 Paths in Graphs

Finally, consider the complete graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$. The ZDD representing the family of all undirected paths in $\mathcal{G}$ may then be constructed using the FBS [26]. For simplicity, suppose the value mapping $v : \mathcal{E} \to \mathbb{R}$ assigns a value 1 to every edge in $\mathcal{E}$, *i.e.*, $v(e) = 1$ for every $e \in \mathcal{E}$. The kth moment of the family $\mathcal{F}$ representing all undirected paths in $\mathcal{G}$ can be computed analytically as

$$M^{(k)}(\mathcal{F}) = \frac{\displaystyle\sum_{p=2}^{|\mathcal{V}|} \binom{|\mathcal{V}|}{p} \frac{p!}{2} (p-1)^k}{\displaystyle\sum_{p=2}^{|\mathcal{V}|} \binom{|\mathcal{V}|}{p} \frac{p!}{2}}. \tag{3.10}$$

Table 3.3 presents the summary of the results for complete graphs with varying edge sizes $n = \binom{|\mathcal{V}|}{2}$. The moments are also calculated until $k = 5$, where both approaches obtained similar values when a ZDD representing $\mathcal{F}$ can be constructed within the provided hardware constraints. In particular, when $|\mathcal{V}| > 20$, a ZDD cannot be constructed due to an out-of-memory (OOM) error attributed to combinatorial explosion [20]. Nevertheless, when $|\mathcal{V}| = 20$, the proposed algorithm extracts the moments within minutes from the family containing 3.3×10^{18}

Table 3.3: Summary of Results for Paths in Graphs

n	ZDD Construction Time	$\lvert\mathcal{D}\rvert$	$Z(\mathcal{D})$	k	Naive Calculation Time	Proposed Calculation Time	Proposed Recursive Calls	Memory Used
$\binom{4}{2}$	< 0.01 s	30	38	1	< 0.01 s	< 0.01 s	190	4 MB
				2	< 0.01 s	< 0.01 s	342	4 MB
				3	< 0.01 s	< 0.01 s	532	4 MB
				4	< 0.01 s	< 0.01 s	760	4 MB
				5	< 0.01 s	< 0.01 s	1.0×10^3	4 MB
$\binom{8}{2}$	< 0.01 s	5.5×10^4	3.4×10^3	1	0.04 s	< 0.01 s	1.7×10^4	4 MB
				2	0.03 s	< 0.01 s	3.1×10^4	4 MB
				3	0.03 s	< 0.01 s	4.8×10^4	4 MB
				4	0.03 s	< 0.01 s	6.8×10^4	4 MB
				5	0.03 s	< 0.01 s	9.2×10^4	4 MB
$\binom{12}{2}$	0.08 s	6.5×10^8	2.8×10^5	1	452.22 s	0.04 s	1.4×10^6	20 MB
				2	469.89 s	0.06 s	2.6×10^6	20 MB
				3	490.61 s	0.09 s	4.0×10^6	22 MB
				4	508.29 s	0.14 s	5.7×10^6	24 MB
				5	528.47 s	0.21 s	7.7×10^6	27 MB
$\binom{16}{2}$	6.67 s	2.8×10^{13}	2.5×10^7	1	> 1 h	4.14 s	1.3×10^8	1.2 GB
				2	> 1 h	6.34 s	2.3×10^8	1.3 GB
				3	> 1 h	9.50 s	3.5×10^8	1.5 GB
				4	> 1 h	14.54 s	5.1×10^8	1.7 GB
				5	> 1 h	21.13 s	6.8×10^8	1.8 GB
$\binom{20}{2}$	585.29 s	3.3×10^{18}	2.5×10^9	1	> 1 h	407.08 s	1.3×10^{10}	110 GB
				2	> 1 h	618.45 s	2.3×10^{10}	125 GB
				3	> 1 h	930.29 s	3.6×10^{10}	141 GB
				4	> 1 h	1431.38 s	5.1×10^{10}	157 GB
				5	> 1 h	2084.79 s	6.9×10^{10}	173 GB

undirected paths, whereas a naive approach failed to perform the same task within an hour. This stark difference underscores the utility of the proposed algorithm, offering more than 2000$\times$ reduction in computational time. Additionally, the algorithm also demonstrates moderate memory requirements, growing linearly with respect to k. These results highlight the significant advantages of the proposed algorithm in efficiently extracting moment information from a ZDD, even for large and complex problems.

The example above may be extended to investigate the distribution of different aspects of a graph. For instance, the vertex-to-vertex distance distribution of a graph is well-studied and typically follows a Gaussian distribution [46]. Using the proposed algorithm, describing such distributions becomes straightforward by calculating the first two moments. This capability enables efficient analysis of various graph properties, providing deeper insights into their structural characteristics and behaviors.

3.4 Summary

Overall, a novel recursive algorithm for extracting statistical moments from families represented by (constructed) ZDDs, rooted in the theoretical framework of the method of moments, is put forward. The efficiency of the proposed algorithm is achieved through a single recursive traversal of the given ZDD, leveraging its inherent structure to effectively handle the non-linearities in the multinomial expansion. The results across the three problems further complement this computational analysis as the algorithm extracts moments information from families containing an exponential number of subsets within seconds or minutes, offering a significant reduction in computational time compared to conventional methods. Validation against known moments information further confirms the correctness of the algorithm, underscoring its utility. It is worth noting, however, that floating point errors may occur when dealing with extreme values, which may produce inaccurate results. In such cases, the central or normalized moments in Appendices A.3 and A.4 may instead be considered. With the proposed algorithm, efficiently uncovering insights into the distribution of subset values from a given ZDD is made possible. This functionality may provide utility in analyzing complex problems where obtaining the statistical moments analytically is not feasible.

4. Path Survival Reliabilities

Reliability engineering plays a critical role across various domains in assessing the ability of complex systems to function satisfactorily after sustaining damage. Within the field, network reliability analysis encompasses a wide range of methods to quantify the capacity of networks to maintain adequate operation following destructive events [15, 39]. One of the most widely used measures is the K-terminal reliability, which evaluates probabilistic connectedness. This measure is readily applicable to a broad class of networks, including lifeline utilities, cloud computing, and mechanical systems [5, 7, 30, 38]. However, this convenience comes at the cost of oversimplification as the measure disregards critical network attributes for the sake of simplicity. Consequently, it fails to capture the nuances and complexities inherent in different types of networks. Given the persistent risk of natural disasters, this chapter focuses on the reliability of distribution networks for lifeline utilities, such as water, electricity, and gas [6, 18, 31].

Specifically, to address the limitations of existing reliability measures for lifeline utility networks, alternative and complementary moment-based reliability measures—path survival reliabilities—designed for stochastic graphs with designated sources, failure-probability-weighted edges, and capacitated edges are introduced. Crucially, these graphs retain the structural and topological properties of the networks. In brief, the per-vertex path survival reliability quantifies the average survival likelihood of single-source paths from a vertex to any source, weighted by maximal flow, in the event of damage. To facilitate comparisons across different network structures, the total path survival reliability is proposed as a consolidated measure, adjusting the per-vertex reliabilities based on the graph density and taking their weighted average with respect to the vertex-to-source shortest distance. The path survival reliabilities, calculated using the ZDD, translate edge-centric failure probabilities into vertex-centric and graph-centric reliability measures. Additionally, theoretical discussions on the key properties and distinctive advantages of the proposed measures demonstrate their strong foundation in engineering principles. These are further complemented by experimental validation on the Hanoi, Bursa, and Kobe water distribution networks, showcasing the ability of the path survival reliabilities in providing a more nuanced characterization of network reliability relative to conventional measures.

4.1 K-Terminal Reliability

Within the field of network reliability, the K-terminal reliability is a fundamental measure that quantifies the probability that a given set of K vertices (terminals) in a graph representing a network remain connected in the event of damage [7, 15, 39]. Its formal definition is given below following [19].

Definition 4.1 (K-Terminal Reliability)**.** *Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ be a stochastic graph with edge failure probabilities $0 \leq \pi(e) \leq 1$, for $e \in \mathcal{E}$. Denote $\mathcal{V}_K \subseteq \mathcal{V}$ a set of K vertices and $\mathcal{G}' = (\mathcal{V}', \mathcal{E}')$ a subgraph of $\mathcal{G}$. The K-terminal reliability is defined as*

$$R_K(\mathcal{G}, \mathcal{V}_K) := \sum_{\mathcal{G}' \subseteq \mathcal{G}} \mathbb{P}(\mathcal{G}') \cdot \mathbb{1}_{\mathcal{F}(\mathcal{G}, \mathcal{V}_K)}\big(\mathcal{G}'\big), \tag{4.1}$$

where

$$\mathbb{P}(\mathcal{G}') := \prod_{e \in \mathcal{E}} \big(\pi(e) \cdot \mathbb{1}_{\mathcal{E} \setminus \mathcal{E}'}(e) + [1 - \pi(e)] \cdot \mathbb{1}_{\mathcal{E}'}(e)\big), \tag{4.2}$$

$\mathcal{F}(\mathcal{G}, \mathcal{V}_K)$ is the family of functioning subgraphs, and $\mathbb{1}_U(x)$ is the indicator function that returns 1 if $x \in U$ and 0 otherwise. For the K-terminal reliability, a subgraph $\mathcal{G}'$ is functioning if and only if the vertices in $\mathcal{V}_K$ all belong to the same connected component in $\mathcal{G}'$.

Computing the K-terminal reliability involves checking the connection between all pairs of vertices in $\mathcal{V}_K$. This is known to be NP-hard [1]. One approach for calculating the exact K-terminal reliability is using the BDD for subgraph representation [17, 19, 27]. Other methods include using the principle of inclusion and exclusion, factoring theorem, and Monte Carlo simulations [7, 8]. More recently, deep neural networks have also been studied for accurate K-terminal reliability estimation, further demonstrating the timelessness of the latter [10].

Furthermore, the simplicity of the K-terminal reliability has made it the standard for measuring the reliability of a network. In fact, a recent review of reliability measures noted that developments in connectivity-based reliability measures primarily extend the K-terminal reliability to suit the specific needs of a network [31]. One such variant considers the probability of any k vertices being connected [9] while another formulation takes into account imperfect vertices and perfect edges [33]. Other efforts towards the development of reliability measures include considering vertex failures [12, 27, 45], ternary networks where edges can have limited operational state [16], and dependent edge failures [2, 32]. Despite these modifications, the underlying principle remains unchanged. Hence, a novel perspective on connectivity-based reliability measures is motivated by existing literature.

4.2 Path Survival Reliabilities

As a prerequisite for formulating the path survival reliabilities, the mathematical definition of a single-source path from a vertex to a source is introduced.

Definition 4.2 (Single-Source Path). *Given a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with a set of source vertices $\mathcal{S} \subset \mathcal{V}$, a single-source path from vertex $v \in \mathcal{V} \setminus \mathcal{S}$ to source $s \in \mathcal{S}$ is a sequence (set) of edges $\mathcal{P} = \left\{e_{p_1}, e_{p_2}, \ldots, e_{p_{|\mathcal{P}|}}\right\}$, where $e_{p_i} = \{v_{p_{i-1}}, v_{p_i}\} \in \mathcal{E}$ is the edge connecting vertices $v_{p_{i-1}}$ and $v_{p_i} \in \mathcal{V}$, for $i = 1, 2, \ldots, |\mathcal{P}|$, $v_{p_0} = v$, and $v_{p_{|\mathcal{P}|}} = s$. Moreover, $v_{p_i} \neq v_{p_j}$ if $i \neq j$ and $v_{p_i} \notin \mathcal{S}$, for $i = 1, 2, \ldots, (|\mathcal{P}| - 1)$.*

A single-source path from a vertex to a source is thus a path that does not traverse more than one source vertex. The formal definition of the path survival reliability is then given.

Definition 4.3 (Path Survival Reliability). *Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ be a graph with a set of source vertices $\mathcal{S} \subset \mathcal{V}$ and edge failure probabilities $0 \leq \pi(e) \leq 1$, for $e \in \mathcal{E}$. Suppose that $\mathcal{P}^i_{v,s} = \left\{e_{p_1}, e_{p_2}, \ldots, e_{p_{\left|\mathcal{P}^i_{v,s}\right|}}\right\}$ is the ith single-source path from $v \in \mathcal{V} \setminus \mathcal{S}$ to $s \in \mathcal{S}$. The path survival reliability of path $\mathcal{P}^i_{v,s}$ is defined as*

$$P\left(\mathcal{P}^i_{v,s}\right) \coloneqq \prod_{e \in \mathcal{P}^i_{v,s}} \left[1 - \pi(e)\right]. \tag{4.3}$$

The measure quantifies the probability that a given single-source path survives in the event of damage. This metric is then used to define the per-vertex path survival reliability, which translates edge failure probabilities into vertex-centric reliability measures.

Definition 4.4 (Per-Vertex Path Survival Reliability). *Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ be a graph with a set of source vertices $\mathcal{S} \subset \mathcal{V}$, edge failure probabilities $0 \leq \pi(e) \leq 1$, and* *edge capacities $\kappa(e) > 0$, for $e \in \mathcal{E}$. For $v \in \mathcal{V} \setminus \mathcal{S}$, the per-vertex path survival reliability is defined as*

$$\Pi^{(\mathcal{G})}(v) := \frac{\sum\limits_{s \in \mathcal{S}} \sum\limits_{\mathcal{P} \in \mathcal{F}(\mathcal{G}, \langle v, s \rangle)} \min\limits_{e \in \mathcal{P}} \kappa(e) \cdot P(\mathcal{P})}{\sum\limits_{s \in \mathcal{S}} \sum\limits_{\mathcal{P} \in \mathcal{F}(\mathcal{G}, \langle v, s \rangle)} \min\limits_{e \in \mathcal{P}} \kappa(e)} \tag{4.4}$$

$$= \frac{\sum\limits_{s \in \mathcal{S}} \sum\limits_{\mathcal{P} \in \mathcal{F}(\mathcal{G}, \langle v, s \rangle)} \min\limits_{e \in \mathcal{P}} \kappa(e) \cdot \prod\limits_{e \in \mathcal{P}} [1 - \pi(e)]}{\sum\limits_{s \in \mathcal{S}} \sum\limits_{\mathcal{P} \in \mathcal{F}(\mathcal{G}, \langle v, s \rangle)} \min\limits_{e \in \mathcal{P}} \kappa(e)}, \tag{4.5}$$

where $\mathcal{F}(\mathcal{G}, \langle v, s \rangle) := \left\{ \mathcal{P}_{v,s}^{1}, \mathcal{P}_{v,s}^{2}, \ldots, \mathcal{P}_{v,s}^{|\mathcal{F}(\mathcal{G}, \langle v, s \rangle)|} \right\}$ is the family of single-source paths from v to $s \in \mathcal{S}$.

The per-vertex path survival reliability $\Pi^{(\mathcal{G})}(v)$ gives the weighted average, by maximal flow, of the probability that single-source paths from $v \in \mathcal{V} \setminus \mathcal{S}$ to any source survive, assuming path and flow independence. The total path survival reliability is then proposed as a consolidated graph measure using the per-vertex path survival reliability. The formulation requires the definition below.

Definition 4.5 (Graph Density)**.** *The density of graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ is defined as*

$$\delta(\mathcal{G}) := \frac{|\mathcal{E}|}{\binom{|\mathcal{V}|}{2}} = \frac{2|\mathcal{E}|}{|\mathcal{V}|\,(|\mathcal{V}| - 1)}. \tag{4.6}$$

The density $\delta(\mathcal{G})$ represents the percentage of the maximum number of unique edges that are present in $\mathcal{G}$. Furthermore, denote the length of the shortest path from vertex $v \in \mathcal{V} \setminus \mathcal{S}$ to any source vertex by $\Delta^{(\mathcal{G})}(v)$. Finally, the total path survival reliability is formally proposed.

Definition 4.6 (Total Path Survival Reliability)**.** *Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ be a graph with a set of source vertices $\mathcal{S} \subset \mathcal{V}$. The total path survival reliability is defined as*

$$\Psi(\mathcal{G}) := \frac{\sum\limits_{v \in \mathcal{V} \setminus \mathcal{S}} \Delta^{(\mathcal{G})}(v) \cdot \left[\Pi^{(\mathcal{G})}(v)\right]^{\delta(\mathcal{G})}}{\sum\limits_{v \in \mathcal{V} \setminus \mathcal{S}} \Delta^{(\mathcal{G})}(v)}. \tag{4.7}$$

The total path survival reliability $\Psi(\mathcal{G})$ is the weighted average, with respect to the vertex-to-source shortest distance $\Delta^{(\mathcal{G})}(v)$, of the adjusted per-vertex path survival reliability. The adjustment to the per-vertex measure is done by raising it to the graph density $\delta(\mathcal{G})$.

4.3 Theoretical Discussion

The path survival reliabilities are crafted primarily for application in reliability engineering, specifically in assessing the ability of lifeline utility networks to withstand damage. Such an objective includes evaluating the ability of these networks to remain functioning satisfactorily after destructive events. In practice, the vertices of a network represent points of demand, such as residential or commercial areas requiring resources. By calculating the per-vertex and total path survival reliabilities from edge failure probabilities, edge-centric reliability information is translated into meaningful vertex-level and overall network reliability measures. Through the contribution, the reliability of a network may be better understood, enabling proactive mitigation strategies to ensure its continued service and resource availability.

To further establish the theoretical soundness of the proposed reliability measures, their mathematical relationship with the widely used K-terminal reliability is first explored. The technical foundation and key features of the path survival reliabilities are then examined through several properties, highlighting their distinctive advantages over the K-terminal reliability. Finally, a discussion on their relationship with probability measures is presented for completeness.

4.3.1 Path Survival Reliabilities and K-Terminal Reliability

The per-vertex path survival reliability may be interpreted as a normalized 2-terminal reliability, where a functioning subgraph is determined by the connectivity of vertex $v \in \mathcal{V} \setminus \mathcal{S}$ to any source vertex $s \in \mathcal{S}$, represented by the family $\mathcal{F}(\mathcal{G}, \langle v, \mathcal{S} \rangle) \coloneqq \bigcup_{s \in \mathcal{S}} \mathcal{F}(\mathcal{G}, \langle v, s \rangle)$. Specifically, this normalized reliability measure for vertex v is computed as the weighted average, with respect to the maximal flow, of the path survival reliabilities P.

$$\Pi^{(\mathcal{G})}(v) = \frac{\displaystyle\sum_{s\in\mathcal{S}} \sum_{\mathcal{P}\in\mathcal{F}(\mathcal{G},\langle v,s\rangle)} \min_{e\in\mathcal{P}} \kappa(e) \cdot P(\mathcal{P})}{\displaystyle\sum_{s\in\mathcal{S}} \sum_{\mathcal{P}\in\mathcal{F}(\mathcal{G},\langle v,s\rangle)} \min_{e\in\mathcal{P}} \kappa(e)} \tag{4.8}$$

$$= \frac{\displaystyle\sum_{s\in\mathcal{S}} \sum_{\mathcal{G}'\subseteq\mathcal{G}} \min_{e\in\mathcal{G}'} \kappa(e) \cdot P(\mathcal{G}') \cdot \mathbb{1}_{\mathcal{F}(\mathcal{G},\langle v,s\rangle)}(\mathcal{G}')}{\displaystyle\sum_{s\in\mathcal{S}} \sum_{\mathcal{G}'\subseteq\mathcal{G}} \min_{e\in\mathcal{G}'} \kappa(e) \cdot \mathbb{1}_{\mathcal{F}(\mathcal{G},\langle v,s\rangle)}(\mathcal{G}')} \tag{4.9}$$

$$= \frac{\displaystyle\sum_{\mathcal{G}'\subseteq\mathcal{G}} \min_{e\in\mathcal{G}'} \kappa(e) \cdot P(\mathcal{G}') \cdot \mathbb{1}_{\mathcal{F}(\mathcal{G},\langle v,\mathcal{S}\rangle)}(\mathcal{G}')}{\displaystyle\sum_{\mathcal{G}'\subseteq\mathcal{G}} \min_{e\in\mathcal{G}'} \kappa(e) \cdot \mathbb{1}_{\mathcal{F}(\mathcal{G},\langle v,\mathcal{S}\rangle)}(\mathcal{G}')} \tag{4.10}$$

$$=: \hat{R}_2(\mathcal{G}, \langle v, \mathcal{S}\rangle). \tag{4.11}$$

Inspired by the 2-terminal network resilience [8], the total path survival reliability may then be viewed as the weighted average of the adjusted and normalized 2-terminal reliabilities.

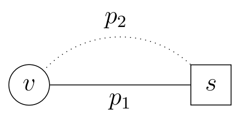
$$\Psi(\mathcal{G}) = \frac{\displaystyle\sum_{v\in\mathcal{V}\setminus\mathcal{S}} \Delta^{(\mathcal{G})}(v) \cdot \left[\Pi^{(\mathcal{G})}(v)\right]^{\delta(\mathcal{G})}}{\displaystyle\sum_{v\in\mathcal{V}\setminus\mathcal{S}} \Delta^{(\mathcal{G})}(v)} \tag{4.12}$$

$$= \frac{\displaystyle\sum_{v\in\mathcal{V}\setminus\mathcal{S}} \Delta^{(\mathcal{G})}(v) \cdot \left[\hat{R}_2(\mathcal{G}, \langle v, \mathcal{S}\rangle)\right]^{\delta(\mathcal{G})}}{\displaystyle\sum_{v\in\mathcal{V}\setminus\mathcal{S}} \Delta^{(\mathcal{G})}(v)}. \tag{4.13}$$

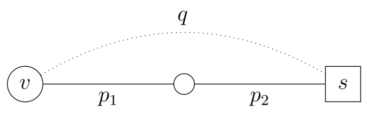
The path survival reliabilities hence adhere to standard reliability criteria. The proposed measures, nonetheless, emphasize normalization to accommodate diverse graph structures and enable comparability across different graphs. This is especially valuable for lifeline utility networks of varying sizes. Furthermore, the path survival reliabilities offer improvements over the K-terminal reliability in the context of lifeline utility networks as they better consider network attributes and properly distinguish between source and non-source vertices.

4.3.2 Key Properties of Path Survival Reliabilities

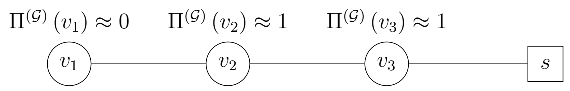
The proposed reliability measures also possess several desirable properties in the context of lifeline utility networks. These properties are highlighted in Fig. 4.1. For simplicity, existing single-source paths or edges are drawn as solid lines, while single-source paths or edges to be added are drawn as dotted lines.



(a) Property 1: Averaging Probabilities



(b) Property 2: Density Adjustment


$\Pi^{(\mathcal{G})}(v_1) \approx 0$
$\Pi^{(\mathcal{G})}(v_2) \approx 1$
$\Pi^{(\mathcal{G})}(v_3) \approx 1$
v_1
v_2
v_3
s

(c) Property 3: Weighted Aggregation

Figure 4.1: Key Properties of the Path Survival Reliabilities

Property 4.1 (Averaging Probabilities)**.** *In Fig. 4.1a, suppose the single-source path from vertex $v \in \mathcal{V} \backslash \mathcal{S}$ to source $s \in \mathcal{S}$ has p_1 probability of surviving after sustaining damage with a maximal capacity of c_1. Adding another single-source path with p_2 probability of surviving after sustaining damage and a maximal capacity of c_2 transforms the per-vertex path survival reliability from*

$$\frac{c_1 \cdot p_1}{c_1} \tag{4.14}$$

to

$$\frac{c_1 \cdot p_1 + c_2 \cdot p_2}{c_1 + c_2}. \tag{4.15}$$

From the property above, adding more single-source paths between a non-source vertex and a source vertex does not necessarily increase the per-vertex path survival reliability. The measure incentivizes adding resilient paths and paths with larger flow capacity and penalizes adding non-resilient paths and paths with smaller flow capacity. The property also highlights how a low per-vertex measure corresponds to a vertex prone to being inaccessible.

Compared to the K-terminal reliability, the per-vertex path survival reliability is more desirable as it incorporates a penalizing factor for adding non-resilient and small paths. In contrast, the addition of non-resilient paths will never decrease the value of the K-terminal reliability since the added path will never decrease the probability of a functioning state.

Property 4.2 (Density Adjustment)**.** *In Fig. 4.1b, suppose the single-source path from vertex $v \in \mathcal{V} \setminus \mathcal{S}$ to source $s \in \mathcal{S}$ has $p_1 \cdot p_2$ probability of surviving after sustaining damage with a maximal capacity of c_p. Making the graph denser by adding an edge with q probability of surviving after sustaining damage and a maximal capacity of c_q transforms the adjusted per-vertex path survival reliability from*

$$\left(\frac{c_p \cdot p_1 \cdot p_2}{c_p}\right)^{\frac{2}{3}} \tag{4.16}$$

to

$$\left(\frac{c_p \cdot p_1 \cdot p_2 + c_q \cdot q}{c_p + c_q}\right)^{1}. \tag{4.17}$$

The second property demonstrates the adjustment to the per-vertex path survival reliability, having incorporated the graph density. In a dense graph, source vertices are more accessible since there are more redundancies for alternative path connections; single-source paths traverse fewer edges on average. Consequently, the per-vertex measure would be in the magnitude of a product of fewer probabilities, making its value significantly larger. Conversely, in a sparse graph, source vertices are less accessible since there are fewer redundancies in place; single-source paths traverse more edges on average. The per-vertex measure would hence be in the magnitude of a product of more probabilities, making its value significantly smaller. The adjustment using the graph density normalizes the per-vertex measures of sparse graphs by increasing their values to approximately the same order of magnitude as the per-vertex measures of dense graphs. The property also highlights how a low per-vertex measure implies a sparse graph or a graph with many non-resilient edges.

Similarly, the adjusted per-vertex path survival reliability is more ideal as it accounts for the differences in graph structure. In contrast, denser graphs will always have a larger K-terminal reliability than sparser graphs since the additional edges increase the probability of a functioning state, *ceteris paribus.*

Property 4.3 (Weighted Aggregation)**.** *In Fig. 4.1c, suppose the per-vertex path survival reliabilities* $\Pi^{(\mathcal{G})}(v_1) \approx 0$ *and* $\Pi^{(\mathcal{G})}(v_2), \Pi^{(\mathcal{G})}(v_3) \approx 1$ *with unit edge length. Replacing the simple average with a weighted average of the adjusted per-vertex path survival reliabilities transforms the aggregated measure from*

$$\frac{\left[\Pi^{(\mathcal{G})}(v_1)\right]^{\delta(\mathcal{G})} + \left[\Pi^{(\mathcal{G})}(v_2)\right]^{\delta(\mathcal{G})} + \left[\Pi^{(\mathcal{G})}(v_3)\right]^{\delta(\mathcal{G})}}{3} \approx \frac{2}{3} \tag{4.18}$$

to

$$\frac{3 \cdot \left[\Pi^{(\mathcal{G})}(v_1)\right]^{\delta(\mathcal{G})} + 2 \cdot \left[\Pi^{(\mathcal{G})}(v_2)\right]^{\delta(\mathcal{G})} + 1 \cdot \left[\Pi^{(\mathcal{G})}(v_3)\right]^{\delta(\mathcal{G})}}{6} \approx \frac{1}{2}. \tag{4.19}$$

The third property demonstrates how the total path survival reliability punishes a graph for being unable to service remote vertices. Specifically, it penalizes a graph for having non-resilient edges critical in connecting source vertices to non-source vertices. The property thus highlights how a low total path survival reliability suggests difficulty in servicing demand points.

The total path survival reliability accounts for the edge failure probabilities, distances, and capacities in a graph. The measure thus takes on a holistic approach that encapsulates both the graph structure and network attributes.

4.3.3 Path Survival Reliability as Exponentiated Probability Measure

Additionally, the path survival reliability, while not directly a probability measure, is grounded in measure theory. Specifically, it may be shown that for $\mathcal{E}' \subseteq \mathcal{E}$, $\hat{P} : 2^{\mathcal{E}} \to [0,1]$ defined as

$$\hat{P}(\mathcal{E}') := \frac{\ln P(\mathcal{E}')}{\ln P(\mathcal{E})} = \frac{\sum_{e' \in \mathcal{E}'} - \ln\left[1 - \pi(e')\right]}{- \ln P(\mathcal{E})} \tag{4.20}$$

is a valid probability measure. In particular, it quantifies the fraction of (logarithmic) uncertainty in the subgraph $\mathcal{G}'$ induced by $\mathcal{E}'$ with $P(\mathcal{E})$ as the baseline quantifying the total (logarithmic) uncertainty in the graph $\mathcal{G}$, regardless of the structure. Thus, the path survival reliability P may then be interpreted as an exponentiated probability measure

$$P\left(\mathcal{E}'\right) = P\left(\mathcal{E}\right)^{\hat{P}\left(\mathcal{E}'\right)}, \qquad P\left(\mathcal{E}\right) < 1. \tag{4.21}$$

In summary, the proposed path survival reliabilities offer a distinctive advantage over the conventional K-terminal reliability by balancing network complexity with meaningful reliability assessment. While both measures adhere to the same reliability criteria, the K-terminal reliability inherently favors complex networks. In contrast, the proposed measures explicitly quantify the magnitude of reliability enhancements attributed to network complexity, providing a more nuanced perspective on network reliability.

4.4 Methodology

Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ be a graph with a set of source vertices $\mathcal{S} \subset \mathcal{V}$, edge failure probabilities $0 \leq \pi\left(e\right) \leq 1$, and edge capacities $\kappa\left(e\right) > 0$, for $e \in \mathcal{E}$. Consider the (constructed) ZDD $\mathcal{D}_{\langle v,s\rangle}$ representing the family $\mathcal{F}\left(\mathcal{G}, \langle v, s\rangle\right)$ of single-source paths from $v \in \mathcal{V} \setminus \mathcal{S}$ to $s \in \mathcal{S}$ rooted at node r. Denote by $\texttt{val}(r) \coloneqq \mathcal{M}^{(r)} : \mathbb{R}_+ \to \mathbb{R}$ a map assigned to each ZDD node, where[2]

$$\mathcal{M}^{(r)}(c) \coloneqq \sum_{\mathcal{P} \in \mathcal{F}(\mathcal{G}, \langle v,s\rangle)} \mathbb{1}_{\left\{\min\limits_{e \in \mathcal{P}} \kappa(e) = c\right\}}\left(\mathcal{P}\right) \cdot \prod_{e \in \mathcal{P}} \left[1 - \pi(e)\right]. \tag{4.22}$$

Specifically, given the minimum edge capacity c, $\mathcal{M}^{(r)}(c)$ returns the sum of the probability every edge in $\mathcal{P}$ survives across $\mathcal{P} \in \mathcal{F}\left(\mathcal{G}, \langle v, s\rangle\right)$ where $\min\limits_{e \in \mathcal{P}} \kappa(e) = c$.

Note that $\texttt{val}(\perp)$ is the empty map and $\texttt{val}(\top)$ is the map defined only at ∞ with value 1. Leveraging the recursive structure of the ZDD highlighted in Eqns. 2.1, 2.2, and 2.3, for a non-terminal node r with label $e_r \in \mathcal{E}$ and i-child r_i representing the family $\mathcal{F}_i$, for $i = 0, 1$, $\mathcal{M}^{(r)}(c)$ may be decomposed as

$$\begin{aligned}\mathcal{M}^{(r)}(c) = \mathcal{M}^{(r_0)}(c) + \left[1 - \pi(e_r)\right] \cdot \mathbb{1}_{\left\{\kappa(e_r) > c\right\}}\left(e_r\right) \cdot \mathcal{M}^{(r_1)}(c) + \\ \left[1 - \pi(e_r)\right] \cdot \mathbb{1}_{\left\{\kappa(e_r) = c\right\}}\left(e_r\right) \cdot \sum_{c' \geq c} \mathcal{M}^{(r_1)}(c').\end{aligned} \tag{4.23}$$

[2]For notational convenience, $\mathbb{1}_{\{f(x)=c\}}\left(x\right) \coloneqq \mathbb{1}_{\{x \mid f(x)=c\}}\left(x\right)$ for some function f.

Appendix B.1 presents the formal proof of the recursive formulation of $\mathcal{M}^{(r)}(c)$. Using $\texttt{val}(r) = \mathcal{M}^{(r)}$, the per-vertex path survival reliability can be calculated from

$$\sum_{c>0} c \cdot \mathcal{M}^{(r)}(c) = \sum_{c>0} c \cdot \sum_{\mathcal{P} \in \mathcal{F}(\mathcal{G}, \langle v,s \rangle)} \mathbb{1}_{\left\{ \min\limits_{e \in \mathcal{P}} \kappa(e) = c \right\}}(\mathcal{P}) \cdot \prod_{e \in \mathcal{P}} [1 - \pi(e)] \tag{4.24}$$

$$= \sum_{\mathcal{P} \in \mathcal{F}(\mathcal{G}, \langle v,s \rangle)} \sum_{c>0} c \cdot \mathbb{1}_{\left\{ \min\limits_{e \in \mathcal{P}} \kappa(e) = c \right\}}(\mathcal{P}) \cdot \prod_{e \in \mathcal{P}} [1 - \pi(e)] \tag{4.25}$$

$$= \sum_{\mathcal{P} \in \mathcal{F}(\mathcal{G}, \langle v,s \rangle)} \min_{e \in \mathcal{P}} \kappa(e) \cdot \prod_{e \in \mathcal{P}} [1 - \pi(e)] \,. \tag{4.26}$$

Algorithm 4.1 presents the pseudocode for calculating the path survival reliabilities. For all pairs of vertices comprising a non-source vertex $v \in \mathcal{V} \setminus \mathcal{S}$ and a source vertex $s \in \mathcal{S}$, the ZDD $\mathcal{D}_{\langle v,s \rangle}$ representing the family $\mathcal{F}(\mathcal{G}, \langle v,s \rangle)$ of single-source paths from v to s is first constructed. The degree constraint `dc` sets v and s as the endpoints of the single-source paths through the assignment of vertex degree 1 as seen in Lines 11 and 13. Line 5 guarantees that paths represented by the ZDD are single-source. Other non-source vertices are either inner vertices of a single-source path or are excluded in a single-source path with vertex degrees set to 0 or 2, as seen in Line 4. With `dc`, `ssp` in Line 14 corresponds to the root of $\mathcal{D}_{\langle v,s \rangle}$ constructed with ConstructPath using the FBS. Afterward, relevant family information is extracted from `ssp`. First, the map $\texttt{val}(r) = \mathcal{M}^{(r)}$ is extracted in Line 15 using Algorithm 4.2 following Eqn. 4.23, where PathSurvival(r, p) is cached to memory and computed only once for every combination of r, p for efficient evaluation. The map from the minimum edge capacity to the number of single-source paths with this minimum edge capacity is also extracted in Line 18. Line 21 then extracts the vertex-to-source shortest distance $\Delta^{(\mathcal{G})}(v)$, following standard dynamic programming techniques. Finally, in Lines 25 to 30, the density $\delta(\mathcal{G})$ is calculated along with the per-vertex and total path survival reliabilities.

In summary, the proposed calculation procedure constructs the ZDDs representing single-source paths to efficiently extract relevant family information needed to compute the per-vertex and total path survival reliabilities. Its efficiency lies in leveraging the recursive structure of the diagram, as a consequence of Eqn. 4.23, as well as the use of memoization techniques. In particular, denote the

Algorithm 4.1 PathSurvivalZDD

```
1: Given a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with a set of source vertices $\mathcal{S} \subset \mathcal{V}$, edge failure probabilities $0 \leq \pi(e) \leq 1$, and edge capacities $\kappa(e) > 0$, for $e \in \mathcal{E}$
2:
3: Let dc, sumProb, numPath, minDist be Maps whose keys are vertices $v \in \mathcal{V}$
4: $\texttt{dc}[v] \leftarrow \{0, 2\}$  for $v \in \mathcal{V} \setminus \mathcal{S}$
5: $\texttt{dc}[s] \leftarrow \{0\}$  for $s \in \mathcal{S}$
6: $\texttt{sumProb}[v] \leftarrow 0$  for $v \in \mathcal{V}$
7: $\texttt{numPath}[v] \leftarrow 0$  for $v \in \mathcal{V}$
8: $\texttt{minDist}[v] \leftarrow \infty$  for $v \in \mathcal{V}$
9:
10: for $v \in \mathcal{V} \setminus \mathcal{S}$ do
11:     $\texttt{dc}[v] \leftarrow \{1\}$
12:     for $s \in \mathcal{S}$ do
13:         $\texttt{dc}[s] \leftarrow \{1\}$
14:         $\texttt{ssp} \leftarrow$ ConstructPath(dc).root
15:         $\texttt{MapProb} \leftarrow$ PathSurvival(ssp, True)
16:         for $c \in$ MapProb.keys do
17:             $\texttt{sumProb}[v] \leftarrow \texttt{sumProb}[v] + c \times \texttt{MapProb}[c]$
18:         $\texttt{MapPath} \leftarrow$ PathSurvival(ssp, False)
19:         for $c \in$ MapPath.keys do
20:             $\texttt{numPath}[v] \leftarrow \texttt{numPath}[v] + c \times \texttt{MapPath}[c]$
21:         $\texttt{minDist}[v] \leftarrow \min(\texttt{minDist}[v], \text{ShortestDist}(\texttt{ssp}))$
22:         $\texttt{dc}[s] \leftarrow \{0\}$
23:     $\texttt{dc}[v] \leftarrow \{0, 2\}$
24:
25: $\texttt{weight} \leftarrow 0, \texttt{sum} \leftarrow 0, \texttt{density} \leftarrow \dfrac{2 \times |\mathcal{E}|}{|\mathcal{V}| \times (|\mathcal{V}| - 1)}$
26: for $v \in \mathcal{V} \setminus \mathcal{S}$ do
27:     print $\left(v, \dfrac{\texttt{sumProb}[v]}{\texttt{numPath}[v]}\right)$        ▷ Per-Vertex Path Survival Reliability
28:     $\texttt{weight} \leftarrow \texttt{weight} + \texttt{minDist}[v]$
29:     $\texttt{sum} \leftarrow \texttt{sum} + \texttt{minDist}[v] \times \text{Pow}\left(\dfrac{\texttt{sumProb}[v]}{\texttt{numPath}[v]}, \texttt{density}\right)$
30: print $\left(\dfrac{\texttt{sum}}{\texttt{weight}}\right)$        ▷ Total Path Survival Reliability
```

Algorithm 4.2 PathSurvival(r, p)

```
1: Assume node $r$ has label $e_r \in \mathcal{E}$
2: if $r = \bot$ then
3:     return Map($\{\}$)
4: else if $r = \top$ then
5:     return Map($\{\infty \mapsto 1\}$)
6: else
7:     Map0 $\leftarrow$ PathSurvival $(r.\text{child}(0), p)$
8:     Map1 $\leftarrow$ PathSurvival $(r.\text{child}(1), p)$
9:     Map1 $\leftarrow$ Map1.ceil $(\kappa(e_r))$
10:    if $p$ is True then
11:        Map1 $\leftarrow$ Map1.scale $(1 - \pi(e_r))$
12:    return Map0 $+$ Map1
```

Notes:

a. `Map`.ceil(c) returns the map such that the value of key c is the sum of the values of keys greater than or equal to c; the keys greater than c are then deleted from the map.

b. `Map`.scale(p) returns the map such that the value of every key is multiplied by p.

c. `Map0` $+$ `Map1` returns the map such that the value of every key is the sum of their values, defaulting to 0 if the key does not exist, across `Map0` and `Map1`.

ZDD with the largest number of nodes across all constructed diagrams by $\mathcal{D}_\Omega$[3]. The time complexity of the calculation procedure is $\mathcal{O}\left(|\mathcal{V} \setminus \mathcal{S}| \cdot |\mathcal{S}| \cdot |\kappa| \cdot Z\left(\mathcal{D}_\Omega\right)\right)$, where $|\kappa|$ is the number of unique minimum edge capacities. Since $Z\left(\mathcal{D}_\Omega\right)$ is generally significantly less than the number of single-source paths, the proposed algorithm offers order-of-magnitude increase in computational efficiency relative to a naive brute-force approach enumerating all single-source paths.

4.5 Results

To demonstrate the utility of the path survival reliabilities and the efficiency of the proposed calculation procedure, consider three real-world networks—the Hanoi, Bursa, and Kobe water distribution networks [14, 24, 25] with $12 \leq |\mathcal{V}| \leq 32$

[3]In practice, constructing the ZDD representing a family of subgraphs using the FBS exhibits moderate time and space complexity [27, 40, 41, 42, 43, 44].

and $15 \leq |\mathcal{E}| \leq 34$ as shown in Fig. 4.2. The C++ programming language, with version 9.4.0 of g++, is chosen for implementation. Fundamental diagram manipulation is carried out through the `TdZdd` library [23]. The complete code is available in the `PSZDD` repository[4]. The reported results use a machine with the Ubuntu 22.04 Long Term Support operating system, the AMD EPYC$^{\text{TM}}$ 7402P processor, and a memory of 256 GB.

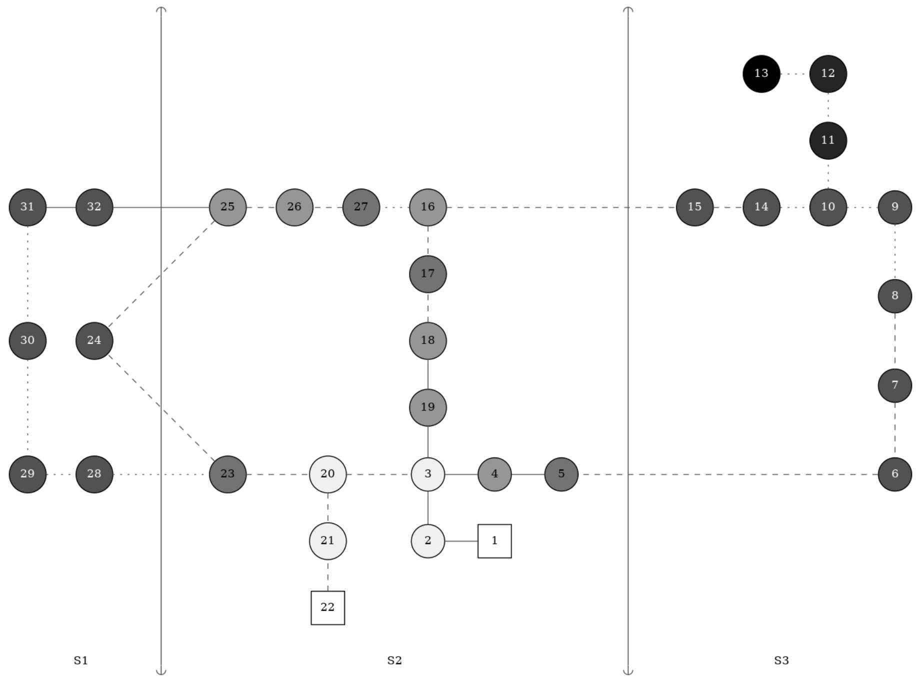
Table 4.1 presents the summary of the calculation procedure for the water distribution networks. Across the three networks, the total time taken by Algorithm 4.1 for both the ZDD construction via FBS and the path survival reliabilities calculation via Algorithm 4.2 took less than a second with minimal memory requirements, underscoring its efficiency. Tables 4.2 to 4.4 then present the vertex-to-source shortest distances, the per-vertex path survival reliabilities, and the adjusted measures for every vertex across the three networks. For completeness, the source vertices are also included and indicated by a superscript † with their per-vertex path survival reliabilities set to 0 for convenience. Moreover, the total path survival reliabilities for the three networks are presented in Table 4.5. It is worth noting that due to the lack of available data, the reported results assume unit distance and capacity for every edge. Nevertheless, physical distances and capacities may be used in actual analyses.

Table 4.1: Summary of Path Survival Reliabilities Calculation for Real-World Networks

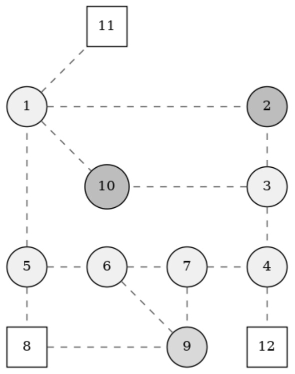
$\mathcal{G}$	$\lvert\mathcal{E}\rvert$	$\lvert\mathcal{V}\rvert$	$\lvert\mathcal{S}\rvert$	Calculation Time	Memory Used
Hanoi	34	32	2	< 0.01 s	4 MB
Bursa	15	12	3	< 0.01 s	4 MB
Kobe	20	15	2	< 0.01 s	4 MB

The utility of the proposed reliability measures is first demonstrated on the Hanoi water distribution network. The network has 32 vertices and 34 edges as seen in Fig. 4.2a. Source vertices 1 and 22 are indicated by squares, while non-source vertices are indicated by circles. The per-vertex path survival reliabilities are also visually presented in the figure, with darker colors indicating a low per-vertex path survival reliability and lighter colors indicating a high per-vertex path

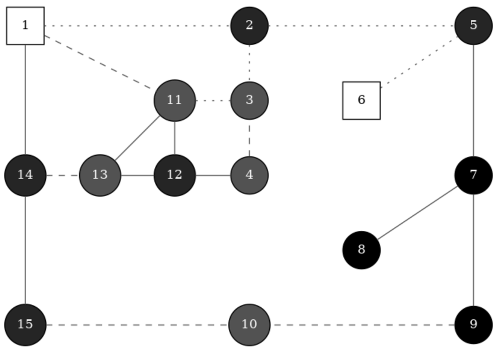
[4]Accessible at `https://github.com/briangodwinlim/PSZDD`.



(a) Hanoi Water Distribution Network



(b) Bursa Water Distribution Network



(c) Kobe Water Distribution Network

Figure 4.2: Per-Vertex Path Survival Reliabilities for Real-World Networks

Table 4.2: Per-Vertex Path Survival Reliabilities for the Hanoi Water Distribution Network

v	$\Delta^{(\mathcal{G})}(v)$	$\Pi^{(\mathcal{G})}(v)$	$\left[\Pi^{(\mathcal{G})}(v)\right]^{\delta(\mathcal{G})}$
$1^{\dagger}$	0	0.000000	0.000000
2	1	0.267818	0.913649
3	2	0.267818	0.913649
4	3	0.153041	0.879265
5	4	0.135560	0.871985
6	5	0.115691	0.862563
7	6	0.097775	0.852671
8	7	0.086071	0.845252
9	8	0.080647	0.841489
10	9	0.081510	0.842103
11	10	0.061214	0.825735
12	11	0.045666	0.809314
13	12	0.033884	0.792928
14	8	0.089747	0.847679
15	7	0.104415	0.856521
16	6	0.157871	0.881140
17	5	0.135344	0.871890
18	4	0.148885	0.877607
19	3	0.163238	0.883162
20	2	0.266390	0.913314
21	1	0.259691	0.911721
$22^{\dagger}$	0	0.000000	0.000000
23	3	0.142651	0.875038
24	4	0.121790	0.865606
25	5	0.167797	0.884831
26	6	0.153099	0.879288
27	7	0.149859	0.878000
28	4	0.115760	0.862598
29	5	0.106663	0.857772
30	6	0.107309	0.858127
31	7	0.117276	0.863368
32	6	0.117276	0.863368

Table 4.3: Per-Vertex Path Survival Reliabilities for the Bursa Water Distribution Network

v	$\Delta^{(\mathcal{G})}(v)$	$\Pi^{(\mathcal{G})}(v)$	$\left[\Pi^{(\mathcal{G})}(v)\right]^{\delta(\mathcal{G})}$
1	1	0.445975	0.832335
2	2	0.392467	0.808505
3	2	0.433994	0.827199
4	1	0.468841	0.841847
5	1	0.442642	0.830916
6	2	0.436605	0.828327
7	2	0.452628	0.835140
$8^\dagger$	0	0.000000	0.000000
9	1	0.422965	0.822374
10	2	0.392486	0.808514
$11^\dagger$	0	0.000000	0.000000
$12^\dagger$	0	0.000000	0.000000

Table 4.4: Per-Vertex Path Survival Reliabilities for the Kobe Water Distribution Network

v	$\Delta^{(\mathcal{G})}(v)$	$\Pi^{(\mathcal{G})}(v)$	$\left[\Pi^{(\mathcal{G})}(v)\right]^{\delta(\mathcal{G})}$
$1^\dagger$	0	0.000000	0.000000
2	1	0.117280	0.664828
3	2	0.135007	0.682894
4	3	0.131926	0.679898
5	1	0.125903	0.673873
$6^\dagger$	0	0.000000	0.000000
7	2	0.108765	0.655351
8	3	0.106263	0.652453
9	3	0.111469	0.658424
10	3	0.139553	0.687216
11	1	0.140014	0.687648
12	2	0.125719	0.673686
13	2	0.132914	0.680866
14	1	0.116606	0.664098
15	2	0.114933	0.662273

Table 4.5: Total Path Survival Reliabilities for Real-World Networks

$\mathcal{G}$	$\lvert\mathcal{E}\rvert$	$\lvert\mathcal{V}\rvert$	$\lvert\mathcal{S}\rvert$	$\delta(\mathcal{G})$	$\Psi(\mathcal{G})$	$\Psi_{\neg\Delta}(\mathcal{G})$	$\Psi_{\neg\Delta,\neg\delta}(\mathcal{G})$	$\Psi_{\neg\delta}(\mathcal{G})$
Hanoi	34	32	2	0.068548	0.853682	0.866054	0.135059	0.109637
Bursa	15	12	3	0.227273	0.824489	0.826128	0.432067	0.428342
Kobe	20	15	2	0.190476	0.670560	0.671039	0.123566	0.123158

survival reliability. Moreover, edges with low failure probabilities are indicated by solid lines, edges with moderate failure probabilities are indicated by dashed lines, and edges with high failure probabilities are indicated by dotted lines.

From the figure, the network may be divided into three sections—S1 on the left, S2 at the center, and S3 on the right. Notably, S2 contains both source vertices of the network. Thus, this section may be regarded as the core of the network. The non-source vertices contained in S2 exhibit a high per-vertex measure, attributed to their relatively shorter distance from the source vertices. Meanwhile, the non-source vertices contained in S1 and S3 exhibit a low per-vertex measure, attributed to their relatively farther distance from the source vertices. Consequently, these two sections may be regarded as remote areas of the network. Furthermore, the per-vertex measure is expected to be lower for remote vertices since more probabilities make up the measure product; the per-vertex measure is expected to be higher for core vertices since fewer probabilities make up the measure product. Additionally, the network is sparse with a density of only 0.0685. Given this, the per-vertex path survival reliabilities are observed to be small in magnitude. Nevertheless, the total path survival reliability is maintained at 0.8537 after adjustment, indicating moderate network reliability. This suggests that points of demand within the core region are adequately serviced, while the reach in remote regions may be improved. Towards proactive risk mitigation, the following observations are noted:

- The network makes a clear distinction between the core and remote areas where there is inequitable access to services;
- The demand points located in remote regions are prone to being inaccessible after damage and would need immediate reinforcement;
- The sparse network structure combined with the presence of numerous non-resilient edges proves to be a major hindrance in its ability to service remote

areas; and

- The network as a whole is moderately reliable in ensuring operation upon sustaining damage.

Similar investigations are conducted for the Bursa and Kobe water distribution networks. Fig. 4.2b presents the result for the Bursa network, formatted in the same manner. The per-vertex path survival reliabilities are relatively high for all vertices since the source vertices are well-distributed across the network. Furthermore, edges in the network are all moderately resilient to damage. Highly resilient edges, however, are not present in the network. In sum, the Bursa network has a total path survival reliability of 0.8245, pointing to moderate network reliability. In addition, the result for the Kobe network is presented in Fig. 4.2c. At first glance, the per-vertex path survival reliabilities are relatively low for all vertices. This is due to the sole edge connecting source vertex 6 to the rest of the network having a high probability of failure. As a result, vertex 6 is highly unlikely to be able to service the network in the event of damage. Consequently, the Kobe network has a total path survival reliability of 0.6706, indicating low network reliability.

4.5.1 Ablation Results

Ablation experiments concerning the graph density $\delta\left(\mathcal{G}\right)$ and the vertex-to-source shortest distance $\Delta^{(\mathcal{G})}\left(v\right)$ as regulating elements for network comparison are also performed to highlight the utility of the total path survival reliability and to complement the theoretical discussions in Section 4.3.2. These additional ablation results are also presented in Table 4.5.

First, consider $\Psi_{\neg\Delta}(\mathcal{G})$, which calculates the total path survival reliability using a simple average by removing the influence of $\Delta^{(\mathcal{G})}\left(v\right)$. The new reliability measure for the larger Hanoi network increased to 0.8661 by 1.45% compared to $\Psi(\mathcal{G})$. This result may be attributed to remote vertices with lower per-vertex measures given substantially lower weights in the aggregated measure, as seen in Table 4.2. Meanwhile, the new reliability measure for the smaller Bursa and Kobe networks increased to 0.8261 and 0.6710 by 0.20% and 0.07%, respectively, reflecting the relatively smaller change in aggregation weights, as seen in Tables 4.3 and 4.4. Nevertheless, the influence of $\Delta^{(\mathcal{G})}\left(v\right)$ becomes more prominent in larger networks where remote vertices are given higher weights to better penalize difficulty in servicing demand points, allowing the total path survival reliability to provide holistic insights into network reliability.

Next, consider $\Psi_{\neg\Delta,\neg\delta}(\mathcal{G})$, which calculates the total path survival reliability using a simple average without adjustment based on $\delta(\mathcal{G})$. Across the Hanoi, Bursa, and Kobe networks, the new reliability measure significantly decreased to 0.1351, 0.4321, and 0.1236, respectively, compared to $\Psi_{\neg\Delta}(\mathcal{G})$. This may be attributed to $\Psi_{\neg\Delta,\neg\delta}(\mathcal{G})$ failing to consider the broader network structure, which is crucial for network comparison. For instance, using $\Psi_{\neg\Delta,\neg\delta}(\mathcal{G})$ suggests that the Hanoi and Kobe networks are comparable in their ability to ensure operation upon sustaining damage. This insight, however, fails to consider the difference in network structures where the larger Hanoi network is expected to have lower per-vertex path survival reliabilities, whereas the smaller Bursa and Kobe networks are expected to have higher per-vertex path survival reliabilities. Accounting for network structure differences through $\delta(\mathcal{G})$ in $\Psi_{\neg\Delta}(\mathcal{G})$ provides more meaningful and nuanced comparisons on network reliability, allowing it to better highlight the high likelihood that source vertex 6 will fail to service the Kobe network upon sustaining damage.

Finally, consider $\Psi_{\neg\delta}(\mathcal{G})$, which calculates the total path survival reliability without adjustment based on $\delta(\mathcal{G})$. Similar to the previous comparison between $\Psi(\mathcal{G})$ and $\Psi_{\neg\Delta}(\mathcal{G})$, the new reliability measure $\Psi_{\neg\delta}(\mathcal{G})$ for the larger Hanoi network decreased to 0.1096 by 18.82% compared to $\Psi_{\neg\Delta,\neg\delta}(\mathcal{G})$ while the new reliability measure for the smaller Bursa and Kobe networks decreased to 0.4283 and 0.1232 by 0.86% and 0.33%, respectively. These results thus highlight the role of $\delta(\mathcal{G})$ and $\Delta^{(\mathcal{G})}(v)$ in accounting for the structural and topological properties of different networks, providing a more nuanced perspective on their ability to service demand points after sustaining damage, while also enabling meaningful reliability comparisons.

Overall, initial results on the three real-world water distribution networks demonstrate that the path survival reliabilities are well-grounded in civil engineering principles. The quantities obtained accurately reflect the network structure and edge failure probabilities. Moreover, practical insights on mitigation and reinforcement strategies may also be easily inferred.

4.6 Summary

Overall, novel moment-based reliability measures for stochastic graphs with designated source vertices and edges characterized by known failure probabilities and capacities are put forward. The proposed path survival reliabilities quantify the likelihood of single-source paths to remain functional after sustaining damage and assess network reliability on two levels—per-vertex and total. Additionally, a theoretical discussion is presented to highlight the distinctive features and advantages of these measures over conventional methods. Moreover, an efficient calculation procedure, leveraging the ZDD for the compact representation of all single-source paths, is also presented to complement the proposal. The path survival reliabilities are then worked out across three real-world water distribution networks, demonstrating their utility in providing valuable insights into network reliability. With the new reliability measures, an alternative and complementary perspective on network reliability, accounting for the nuances and complexities of lifeline networks, is made available. This may aid policymakers and the development sector in identifying potential risks of existing networks and designing effective reinforcement measures.

5. Conclusion

In summary, this dissertation explores two novel applications of the recursive property of the ZDD in the context of discrete optimization. First, a method-of-moments-based framework for bridging statistical moments and the ZDD is introduced, motivated by combinatorial applications. Within the framework, an efficient recursive algorithm for extracting moments information, leveraging the inherent structure of the ZDD, is developed. Its utility is then demonstrated across three classical combinatorial problems, achieving orders-of-magnitude improvements in computational efficiency compared to conventional approaches. The proposed algorithm thus contributes to ZDD literature by providing a practical means to extract distributional properties from arbitrary families represented by a ZDD.

Moreover, novel moment-based reliability measures for stochastic graphs with designated source vertices are also introduced. A comprehensive theoretical discussion underscores the technical features and advantages of the path survival reliabilities over standard metrics. The proposal is further accompanied by an efficient calculation procedure utilizing the recursive structure of the ZDD. When applied to three real-world water distribution networks, these measures yield well-grounded results, offering valuable insights into network mitigation and reinforcement strategies. These measures thus provide a meaningful contribution to reliability engineering by providing an alternative and complementary perspective on network reliability for lifeline networks.

Future studies may explore integrating the Fast Fourier Transform into the moment evaluation algorithm to achieve linearithmic complexity with respect to the moment k. Additionally, hierarchically incorporating additional network attributes into the path survival reliabilities may further enhance their ability to capture the complexities of lifeline utility networks. Lastly, a redundancy-centric reformulation of the path survival reliabilities, potentially based on flow capacities, may also be explored.

Acknowledgements

These three years have been an incredible journey filled with challenges, growth, and unforgettable moments. I could not have made it through without the support, guidance, and kindness of many.

To the Government of Japan and MEXT, for the generous financial support that made this pursuit possible.

To *Ikeda-sensei* and the members of the Mathematical Informatics Laboratory, for their invaluable guidance and encouragement throughout my research.

To my family, for their unwavering support and steadfast belief in me, which have been my greatest source of strength throughout this journey.

To my friends, for the camaraderie and cherished memories that made this journey all the more meaningful.

To the *Gohonzon*, for the wisdom, strength, and countless blessings that have guided my path.

No words can fully capture the depth of my gratitude, but from the bottom of my heart 本当にありがとうございます.

References

[1] Michael O. Ball. Computational complexity of network reliability analysis: An overview. *IEEE Transactions on Reliability*, 35(3):230–239, 1986.

[2] Javiera Barrera, Héctor Cancela, and Eduardo Moreno. Topological optimization of reliable networks under dependent failures. *Operations Research Letters*, 43(2):132–136, 2015.

[3] Rami Beidas and Jason H. Anderson. Mapping enumeration for multi-context CGRAs using zero-suppressed binary decision diagrams. In *2024 IEEE 32nd Annual International Symposium on Field-Programmable Custom Computing Machines (FCCM)*, pages 151–161, 2024.

[4] Randal E. Bryant. Graph-based algorithms for boolean function manipulation. *IEEE Transactions on Computers*, C-35(8):677–691, 1986.

[5] Jacques Carlier, Yu Li, and Jean-luc Lutton. Reliability evaluation of large telecommunication networks. *Discrete Applied Mathematics*, 76(1):61–80, 1997.

[6] Jesús M. Ceresuela, Daniel Chemisana, and Nacho López. Household photovoltaic systems optimization methodology based on graph theory reliability. *Energy Reports*, 8:11334–11342, 2022.

[7] Sanjay K. Chaturvedi. *Network Reliability: Measures and Evaluation*. John Wiley & Sons, 2016.

[8] Charles J. Colbourn. Analysis and synthesis problems for network resilience. *Mathematical and Computer Modelling*, 17(11):43–48, 1993.

[9] Jason L. Cook and Jose E. Ramirez-Marquez. Reliability of capacitated mobile ad hoc networks. *Proceedings of the Institution of Mechanical Engineers, Part O: Journal of Risk and Reliability*, 221(4):307–318, 2007.

[10] Alex Davila-Frias, Nita Yodo, Trung Le, and Om Prakash Yadav. A deep neural network and Bayesian method based framework for all-terminal

network reliability estimation considering degradation. *Reliability Engineering & System Safety*, 229:108881, 2023.

[11] Hiromi Emoto, Yuni Iwamasa, and Shin-ichi Minato. On the sizes of BDDs and ZDDs representing matroids, 2024. arXiv:2404.14670.

[12] Trefor Evans and Derek Smith. Optimally reliable graphs for both edge and vertex failures. *Networks*, 16(2):199–204, 1986.

[13] Kazuma Fuchimoto, Shin-Ichi Minato, and Maomi Ueno. Automated parallel test forms assembly using zero-suppressed binary decision diagrams. *IEEE Access*, 11:112804–112813, 2023.

[14] Okitsugu Fujiwara and Do Ba Khang. A two-phase decomposition method for optimal design of looped water distribution networks. *Water Resources Research*, 26(4):539–549, 1990.

[15] Vaibhav Gaur, Om Prakash Yadav, Gunjan Soni, and Ajay Pal Singh Rathore. A literature review on network reliability analysis and its engineering applications. *Proceedings of the Institution of Mechanical Engineers, Part O: Journal of Risk and Reliability*, 235(2):167–181, 2021.

[16] Ilya Gertsbakh, Yoseph Shpungin, and Radislav Vaisman. *Ternary Networks: Reliability and Monte Carlo*. SpringerBriefs in Electrical and Computer Engineering. Springer, 2014.

[17] Mohammad Ghasemzadeh, Christoph Meinel, and Sara Khanji. K-terminal network reliability evaluation using binary decision diagram. In *2008 3rd International Conference on Information and Communication Technologies: From Theory to Applications*, pages 1–5, 2008.

[18] Yu Gu, Xiao Fu, Zhiyuan Liu, Xiangdong Xu, and Anthony Chen. Performance of transportation network under perturbations: Reliability, vulnerability, and resilience. *Transportation Research Part E: Logistics and Transportation Review*, 133:101809, 2020.

[19] Gary Hardy, Corinne Lucet, and Nikolaos Limnios. K-terminal network reliability measures with binary decision diagrams. *IEEE Transactions on Reliability*, 56(3):506–515, 2007.

[20] Takeru Inoue, Hiroaki Iwashita, Jun Kawahara, and Shin-ichi Minato. Graphillion: Software library for very large sets of labeled graphs. *International Journal on Software Tools for Technology Transfer*, 18:57–66, 2016.

[21] Shinya Ishihara and Shin-ichi Minato. Manipulation of regular expressions under length constraints using zero-suppressed-BDDs. In *Proceedings of ASP-DAC'95/CHDL'95/VLSI'95 with EDA Technofair*, pages 391–396, 1995.

[22] Hiroaki Iwashita, Jun Kawahara, and Shin-ichi Minato. ZDD-based computation of the number of paths in a graph, 2012. Hokkaido University Division of Computer Science TCS Technical Report TCS-TR-A-12-60.

[23] Hiroaki Iwashita and Shin-ichi Minato. Efficient top-down ZDD construction techniques using recursive specifications, 2013. Hokkaido University Division of Computer Science TCS Technical Report TCS-TR-A-13-69.

[24] Mohammad B. Javanbarg, Junji Kiyono, and Mohammad R. Jalili Ghazizadeh. Reliability analysis of lifeline networks using binary decision diagram. In *Proceedings of the 4th International Conference on Modern Research in Civil Engineering, and Architectural and Urban Development*, 2016.

[25] Mohammad B. Javanbarg, Charles Scawthorn, Junji Kiyono, and Yusuke Ono. Reliability analysis of infrastructure and lifeline networks using OBDD. In *Safety, Reliability and Risk of Structures, Infrastructures and Engineering Systems*, pages 3463–3470, 2010.

[26] Jun Kawahara, Takeru Inoue, Hiroaki Iwashita, and Shin-ichi Minato. Frontier-based search for enumerating all constrained subgraphs with compressed representation. *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, E100.A(9):1773–1784, 2017.

[27] Jun Kawahara, Koki Sonoda, Takeru Inoue, and Shoji Kasahara. Efficient construction of binary decision diagrams for network reliability with imperfect vertices. *Reliability Engineering & System Safety*, 188:142–154, 2019.

[28] Donald E. Knuth. *The Art of Computer Programming: Combinatorial Algorithms*, volume 4A. Addison-Wesley, 2009.

[29] Victor Kondratiev, Ilya Otpuschennikov, and Alexander Semenov. Using decision diagrams of special kind for compactification of conflict data bases generated by CDCL SAT solvers. In *2020 43rd International Convention on Information, Communication and Electronic Technology (MIPRO)*, pages 1046–1051, 2020.

[30] Shunlong Li, Jie Wang, and Shaoyang He. Connectivity probability evaluation of a large-scale highway bridge network using network decomposition. *Reliability Engineering & System Safety*, 236:109191, 2023.

[31] Yan-Fu Li and Chuanzhou Jia. An overview of the reliability metrics for power grids and telecommunication networks. *Frontiers of Engineering Management*, 8:531–544, 2021.

[32] Xuelian Long, David Tipper, and Teresa Gomes. Measuring the survivability of networks to geographic correlated failures. *Optical Switching and Networking*, 14:117–133, 2014.

[33] Denis A. Migov and Vladimir Shakhov. Reliability of ad hoc networks with imperfect nodes. In *Multiple Access Communications*, pages 49–58, 2014.

[34] Shin-ichi Minato. Zero-suppressed BDDs for set manipulation in combinatorial problems. In *Proceedings of the 30th International Design Automation Conference*, DAC '93, page 272–277, 1993.

[35] Shin-ichi Minato. Implicit manipulation of polynomials using zero-suppressed BDDs. In *Proceedings the European Design and Test Conference. ED&TC 1995*, pages 449–454, 1995.

[36] Shin-ichi Minato. Zero-suppressed BDDs and their applications. *International Journal on Software Tools for Technology Transfer*, 3:156–170, 2001.

[37] Shin-ichi Minato, Jun Kawahara, Mutsunori Banbara, Takashi Horiyama, Ichigaku Takigawa, and Yutaro Yamaguchi. Fast enumeration of all cost-bounded solutions for combinatorial problems using ZDDs. *Discrete Applied Mathematics*, 360:467–486, 2025.

[38] Yuchang Mo, Min Liang, Liudong Xing, Jinping Liao, and Xuli Liu. Network simplification and K-terminal reliability evaluation of sensor-cloud systems. *IEEE Access*, 8:177206–177218, 2020.

[39] Hebert Pérez-Rosés. Sixty years of network reliability. *Mathematics in Computer Science*, 12:275–293, 2018.

[40] Hirofumi Suzuki, Masakazu Ishihata, and Shin-ichi Minato. Designing survivable networks with zero-suppressed binary decision diagrams. In *WALCOM: Algorithms and Computation*, pages 273–285, 2020.

[41] Atsushi Takizawa, Yasufumi Takechi, Akio Ohta, Naoki Katoh, Takeru Inoue, Takashi Horiyama, Jun Kawahara, and Shin-ichi Minato. Enumeration of region partitioning for evacuation planning based on ZDD. In *11th International Symposium on Operations Research and its Applications in Engineering, Technology and Management 2013 (ISORA 2013)*, pages 1–8, 2013.

[42] Renzo Roel P. Tan, Jun Kawahara, Kazushi Ikeda, Agnes D. Garciano, and Kyle Stephen S. See. Concerning a decision-diagram-based solution to the generalized directed rural postman problem. *IAENG International Journal of Computer Science*, 47(2):302–309, 2020.

[43] Renzo Roel P. Tan, Kyle Stephen S. See, Jun Kawahara, Kazushi Ikeda, Richard M. de Jesus, Lessandro Estelito O. Garciano, and Agnes D. Garciano. The relative isolation probability of a vertex in a multiple-source edge-weighted graph. *Engineering Letters*, 30(1):117–130, 2022.

[44] Renzo Roel P. Tan, Florian Sikora, Kazushi Ikeda, and Kyle Stephen S. See. Arc routing based on the zero-suppressed binary decision diagram. In *Transactions on Engineering Technologies*, pages 105–120, 2021.

[45] Shihu Xiang and Jun Yang. K-terminal reliability of ad hoc networks considering the impacts of node failures and interference. *IEEE Transactions on Reliability*, 69(2):725–739, 2020.

[46] Qi Ye, Bin Wu, and Bai Wang. Distance distribution and average shortest path length estimation in real-world networks. In *Advanced Data Mining and Applications*, pages 322–333, 2010.

[47] Ryo Yoshinaka, Jun Kawahara, Shuhei Denzumi, Hiroki Arimura, and Shin-ichi Minato. Counterexamples to the long-standing conjecture on the complexity of BDD binary operations. *Information Processing Letters*, 112(16):636–640, 2012.

Appendices

A. Statistical Moments

A.1 Proof of Recursive Formulation for Eqn. 3.5

Leveraging the recursive structure of the ZDD, $\texttt{val}^{(k)}(r)$ in Eqn. 3.5 may be written recursively as

$$\texttt{val}^{(k)}(r) = \sum_{S\in\mathcal{F}} \sum_{\substack{\sum_{x\in S} p(x)=k \\ p(x)\geq 0,\ \forall x\in S}} \frac{k!}{\prod_{x\in S} p(x)!} \prod_{x\in S} [v(x)]^{p(x)} \tag{A.1}$$

$$= \sum_{S\in\mathcal{F}_0} \sum_{\substack{\sum_{x\in S} p(x)=k \\ p(x)\geq 0,\ \forall x\in S}} \frac{k!}{\prod_{x\in S} p(x)!} \prod_{x\in S} [v(x)]^{p(x)} + \sum_{S\in\mathcal{F}\setminus\mathcal{F}_0} \sum_{\substack{\sum_{x\in S} p(x)=k \\ p(x)\geq 0,\ \forall x\in S}} \frac{k!}{\prod_{x\in S} p(x)!} \prod_{x\in S} [v(x)]^{p(x)} \tag{A.2}$$

$$\begin{aligned} = &\sum_{S\in\mathcal{F}_0} \sum_{\substack{\sum_{x\in S} p(x)=k \\ p(x)\geq 0,\ \forall x\in S}} \frac{k!}{\prod_{x\in S} p(x)!} \prod_{x\in S} [v(x)]^{p(x)} \\ &+ \sum_{S\in\mathcal{F}\setminus\mathcal{F}_0} \sum_{\substack{\sum_{x\in S\setminus\{x_r\}} p(x)=k-p(x_r) \\ p(x)\geq 0,\ \forall x\in S}} \frac{k!}{[k-p(x_r)]!\cdot p(x_r)!} \frac{[k-p(x_r)]!}{\prod_{x\in S\setminus\{x_r\}} p(x)!} [v(x_r)]^{p(x_r)} \prod_{x\in S\setminus\{x_r\}} [v(x)]^{p(x)} \end{aligned} \tag{A.3}$$

$$\begin{aligned} = &\sum_{S\in\mathcal{F}_0} \sum_{\substack{\sum_{x\in S} p(x)=k \\ p(x)\geq 0,\ \forall x\in S}} \frac{k!}{\prod_{x\in S} p(x)!} \prod_{x\in S} [v(x)]^{p(x)} \\ &+ \sum_{S\in\mathcal{F}\setminus\mathcal{F}_0} \sum_{p(x_r)=0}^{k} \binom{k}{p(x_r)} [v(x_r)]^{p(x_r)} \sum_{\substack{\sum_{x\in S\setminus\{x_r\}} p(x)=k-p(x_r) \\ p(x)\geq 0,\ \forall x\in S\setminus\{x_r\}}} \frac{[k-p(x_r)]!}{\prod_{x\in S\setminus\{x_r\}} p(x)!} \prod_{x\in S\setminus\{x_r\}} [v(x)]^{p(x)} \end{aligned} \tag{A.4}$$

$$= \sum_{S \in \mathcal{F}_0} \sum_{\substack{\sum_{x \in S} p(x) = k \\ p(x) \geq 0, \ \forall x \in S}} \frac{k!}{\prod_{x \in S} p(x)!} \prod_{x \in S} [v(x)]^{p(x)} + \sum_{p=0}^{k} \binom{k}{p} [v(x_r)]^p \sum_{S \in \mathcal{F}_1} \sum_{\substack{\sum_{x \in S} p(x) = k-p \\ p(x) \geq 0, \ \forall x \in S}} \frac{[k-p]!}{\prod_{x \in S} p(x)!} \prod_{x \in S} [v(x)]^{p(x)} \tag{A.5}$$

$$= \texttt{val}^{(k)}(r_0) + \sum_{p=0}^{k} \binom{k}{p} [v(x_r)]^p \, \texttt{val}^{(k-p)}(r_1). \tag{A.6}$$

A.2 Dynamic Programming for the Knapsack Problem

The Python code to obtain the kth moment of the total rewards across all valid subsets in the knapsack problem is provided in the function `dp` below. The `@memoize` decorator handles the memoization procedure.

```python
@memoize
def dp(idx, capacity, value, moment):
    if idx == len(items):
        return value ** moment

    values = dp(idx + 1, capacity, value, moment)           # Exclude current item

    if capacity >= items[idx]["w"]:                          # Include current item
        values += dp(idx + 1, capacity - items[idx]["w"],
                                    value + items[idx]["v"], moment)

    return values
```

Listing A.1: Python Dynamic Programming Code for the Knapsack Problem

A.3 Central Moments of a Family

In certain cases, the kth central moment of a non-empty family $\mathcal{F}$ defined as

$$M^{(k)}(\mathcal{F}) \coloneqq \frac{1}{|\mathcal{F}|} \sum_{S \in \mathcal{F}} \left[V(S) - \mu\right]^k, \tag{A.7}$$

where μ is the mean (or some constant) may be required. One may use the current framework and redefine

$$\begin{aligned} \texttt{val}^{(k)}(r) &\coloneqq \sum_{S \in \mathcal{F}} \left[\sum_{x \in S} v(x) - \mu\right]^k & \text{(A.8)} \\ &= \sum_{S \in \mathcal{F}} \left[\sum_{x \in S \cup \{\varnothing\}} v(x)\right]^k, & \text{(A.9)} \end{aligned}$$

where $\varnothing$ is a placeholder element such that $v(\varnothing) = -\mu$. It still follows that $\texttt{val}^{(0)}(r) = |\mathcal{F}|$ and

$$M^{(k)}(\mathcal{F}) = \frac{\texttt{val}^{(k)}(r)}{\texttt{val}^{(0)}(r)}. \tag{A.10}$$

Moreover, $\texttt{val}^{(k)}(\bot) = 0$ and $\texttt{val}^{(k)}(\top) = (-\mu)^k$ for all $k \geq 0$. It may be shown, following a similar reasoning in Appendix A.1, that

$$\texttt{val}^{(k)}(r) = \texttt{val}^{(k)}(r_0) + \sum_{p=0}^{k} \binom{k}{p} [v(x_r)]^p \, \texttt{val}^{(k-p)}(r_1). \tag{A.11}$$

A.4 Normalized Moments of a Family

It may also be useful to define the kth normalized moment of a non-empty family $\mathcal{F}$ as

$$M^{(k)}(\mathcal{F}) \coloneqq \frac{1}{|\mathcal{F}|} \sum_{S \in \mathcal{F}} \left[\frac{V(S) - \mu}{\sigma}\right]^k, \tag{A.12}$$

where σ is the standard deviation (or some constant). One may then redefine

$$\texttt{val}^{(k)}(r) \coloneqq \sum_{S \in \mathcal{F}} \left[\frac{1}{\sigma} \cdot \left(\sum_{x \in S} v(x) - \mu\right)\right]^k \tag{A.13}$$

$$= \sum_{S \in \mathcal{F}} \left[\sum_{x \in S \cup \{\varnothing\}} \frac{v(x)}{\sigma} \right]^k . \tag{A.14}$$

Similarly, it follows that $\mathtt{val}^{(0)}(r) = |\mathcal{F}|$ and

$$M^{(k)}(\mathcal{F}) = \frac{\mathtt{val}^{(k)}(r)}{\mathtt{val}^{(0)}(r)}. \tag{A.15}$$

Furthermore, $\mathtt{val}^{(k)}(\bot) = 0$ and $\mathtt{val}^{(k)}(\top) = \left(-\frac{\mu}{\sigma}\right)^k$ for all $k \geq 0$. Finally, following Appendix A.1, it may be shown that

$$\mathtt{val}^{(k)}(r) = \mathtt{val}^{(k)}(r_0) + \sum_{p=0}^{k} \binom{k}{p} \left[\frac{v(x_r)}{\sigma}\right]^p \mathtt{val}^{(k-p)}(r_1). \tag{A.16}$$

B. Path Survival Reliabilities

B.1 Proof of Recursive Formulation for Eqn. 4.22

Leveraging the recursive structure of the ZDD, the map `val`$(r) = \mathcal{M}^{(r)}$ in Eqn. 4.22 may be written recursively as

$$\begin{aligned}
\mathcal{M}^{(r)}(c) &= \sum_{\mathcal{E}' \in \mathcal{F}} \mathbb{1}_{\left\{\min\limits_{e \in \mathcal{E}'} \kappa(e) = c\right\}}(\mathcal{E}') \cdot \prod_{e \in \mathcal{E}'} [1 - \pi(e)] && \text{(B.1)} \\
&= \sum_{\mathcal{E}' \in \mathcal{F}_0} \mathbb{1}_{\left\{\min\limits_{e \in \mathcal{E}'} \kappa(e) = c\right\}}(\mathcal{E}') \cdot \prod_{e \in \mathcal{E}'} [1 - \pi(e)] + \sum_{\mathcal{E}' \in \mathcal{F} \setminus \mathcal{F}_0} \mathbb{1}_{\left\{\min\limits_{e \in \mathcal{E}'} \kappa(e) = c\right\}}(\mathcal{E}') \cdot \prod_{e \in \mathcal{E}'} [1 - \pi(e)] && \text{(B.2)} \\
&= \mathcal{M}^{(r_0)}(c) + \sum_{\mathcal{E}' \in \mathcal{F} \setminus \mathcal{F}_0} \mathbb{1}_{\{\kappa(e_r) > c\}}(e_r) \cdot \mathbb{1}_{\left\{\min\limits_{e \in \mathcal{E}' \setminus \{e_r\}} \kappa(e) = c\right\}}(\mathcal{E}') \cdot [1 - \pi(e_r)] \cdot \prod_{e \in \mathcal{E}' \setminus \{e_r\}} [1 - \pi(e)] \\
&\qquad + \sum_{\mathcal{E}' \in \mathcal{F} \setminus \mathcal{F}_0} \mathbb{1}_{\{\kappa(e_r) = c\}}(e_r) \cdot \mathbb{1}_{\left\{\min\limits_{e \in \mathcal{E}' \setminus \{e_r\}} \kappa(e) \geq c\right\}}(\mathcal{E}') \cdot [1 - \pi(e_r)] \cdot \prod_{e \in \mathcal{E}' \setminus \{e_r\}} [1 - \pi(e)] && \text{(B.3)} \\
&= \mathcal{M}^{(r_0)}(c) + \mathbb{1}_{\{\kappa(e_r) > c\}}(e_r) \cdot [1 - \pi(e_r)] \cdot \sum_{\mathcal{E}' \in \mathcal{F}_1} \mathbb{1}_{\left\{\min\limits_{e \in \mathcal{E}'} \kappa(e) = c\right\}}(\mathcal{E}') \cdot \prod_{e \in \mathcal{E}'} [1 - \pi(e)] \\
&\qquad + \mathbb{1}_{\{\kappa(e_r) = c\}}(e_r) \cdot [1 - \pi(e_r)] \cdot \sum_{\mathcal{E}' \in \mathcal{F}_1} \mathbb{1}_{\left\{\min\limits_{e \in \mathcal{E}'} \kappa(e) \geq c\right\}}(\mathcal{E}') \cdot \prod_{e \in \mathcal{E}'} [1 - \pi(e)] && \text{(B.4)} \\
&= \mathcal{M}^{(r_0)}(c) + [1 - \pi(e_r)] \cdot \left\{ \mathbb{1}_{\{\kappa(e_r) > c\}}(e_r) \cdot \mathcal{M}^{(r_1)}(c) + \mathbb{1}_{\{\kappa(e_r) = c\}}(e_r) \cdot \sum_{c' \geq c} \mathcal{M}^{(r_1)}(c') \right\}. && \text{(B.5)}
\end{aligned}$$

Publication List

Journal Articles

[1] Brian Lim, Miguel Saavedra, Renzo Tan, Kazushi Ikeda, and William Yu. Geo-distributed multi-cloud data centre storage tiering and selection with zero-suppressed binary decision diagrams. *International Journal of Cloud Computing*, 14(2):163–182, 2025.

[2] Brian Godwin Lim, Dominic Dayta, Benedict Ryan Tiu, Renzo Roel Tan, Len Patrick Dominic Garces, and Kazushi Ikeda. Dynamic factor analysis of price movements in the Philippine Stock Exchange. *Financial Innovation*, 2026. In Press.

[3] Brian Godwin Lim, Galvin Brice Sy Lim, Renzo Roel Tan, and Kazushi Ikeda. Contextualized messages boost graph representations. *Transactions on Machine Learning Research*, 2025.

[4] Brian Godwin Lim, Renzo Roel Tan, Richard de Jesus, Lessandro Estelito Garciano, Agnes Garciano, and Kazushi Ikeda. Path survival reliabilities as measures of reliability for lifeline utility networks. *Journal of Combinatorial Optimization*, 49(57), 2025.

[5] Brian Godwin Lim, Renzo Roel Tan, Jun Kawahara, Shin-Ichi Minato, and Kazushi Ikeda. A recursive framework for evaluating moments using zero-suppressed binary decision diagrams. *IEEE Access*, 12:91886–91895, 2024.

[6] Zihao Yu, Mark Christian S. G. Guinto, Brian Godwin S. Lim, Renzo Roel P. Tan, Junichiro Yoshimoto, Kazushi Ikeda, Yasumi Ohta, and Jun Ohta. Engineering a data processing pipeline for an ultra-lightweight lensless fluorescence imaging device with neuronal-cluster resolution. *Artificial Life and Robotics*, 28:483–495, 2023.

Conference Proceedings

[1] Brian Godwin Lim, Jiahong Liu, Hans Jarett Ong, Jan Adrian Chan, Renzo Roel Tan, Irwin King, and Kazushi Ikeda. FinSIR: Financial SIR-GCN for market-aware satock recommendation. In *2025 International Joint Conference on Neural Networks (IJCNN)*, 2025. In Press.

[2] Brian Godwin Lim, Hans Jarett Ong, Renzo Roel Tan, and Kazushi Ikeda. Dynamic principal component analysis for the construction of high-frequency economic indicators. In *Proceedings of the 4th International Conference on Advances in Computational Science and Engineering*, pages 645–663, 2024.

[3] Hans Jarett J. Ong, Brian Godwin S. Lim, Benedict Ryan C. Tiu, Renzo Roel P. Tan, and Kazushi Ikeda. A compression-based dependence measure for causal discovery by additive noise models. In *Neural Information Processing*, pages 61–75, 2025.

[4] Renzo Roel Perez Tan, Aldrich Ellis Catapang Asuncion, Brian Godwin Sy Lim, Mate Soos, and Kazushi Ikeda. The pancake graph of order 10 is 4-colorable. In *Proceedings of the 2023 6th International Conference on Mathematics and Statistics*, pages 1–6, 2023.

Submitted Works

[1] Galvin Brice S. Lim, Brian Godwin S. Lim, Argel A. Bandala, John Anthony C. Jose, Timothy Scott C. Chu, and Edwin Sybingco. AGTCNet: A graph-temporal approach for principled motor imagery EEG classification. *IEEE Access*, 2025.

[2] Hans Jarett J. Ong, Brian Godwin S. Lim, Renzo Roel Tan, and Kazushi Ikeda. Redefining the shortest path problem formulation of the linear non-gaussian acyclic model: Pairwise likelihood ratios, prior knowledge, and path enumeration. *IEICE Transactions on Information and Systems*, 2025.

[3] Zihao Yu, Brian Godwin S. Lim, Renzo Roel P. Tan, Hirotaka Kaji, and Kazushi Ikeda. A graph-based heuristic for bus route planning using taxi origin-destination data. *IEEE Access*, 2025.