Concerning a Decision-Diagram-Based Solution to the Generalized Directed Rural Postman Problem

Renzo Roel P. Tan*, Jun Kawahara, Kazushi Ikeda, Agnes D. Garciano, and Kyle Stephen S. See

***Abstract*—Decision-diagram-based solutions for discrete optimization have been persistently studied. Among these is the use of the zero-suppressed binary decision diagram, a compact graph-based representation for a specified family of sets. Such a diagram may work out combinatorial problems by efficient enumeration. In brief, an extension to the frontier-based search approach for zero-suppressed binary decision diagram construction is proposed. The modification allows for the inclusion of a class-determined constraint in formulation. Variations of the generalized directed rural postman problem, proven to be nondeterministic polynomial-time hard, are solved on some rapid transit systems as illustration. Lastly, results are juxtaposed against standard integer programming in establishing the relative superiority of the new technique.**

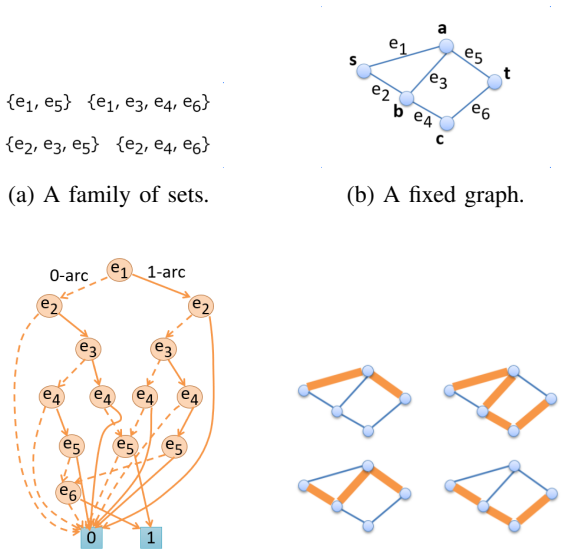
***Index Terms*—enumeration algorithm, frontier-based search, generalized directed rural postman, subgraph optimization, zero-suppressed binary decision diagram.**

I. Introduction

THE zero-suppressed binary decision diagram is a compressed data structure capable of storing families of sets [1]. Due to the recursive structure of representation, mathematical operations on families may be carried out comfortably [2]. The intersection and union, for instance, are swiftly calculated using the diagram. That being the case, feasible solutions to discrete optimization problems may be economically kept in a zero-suppressed binary decision diagram [3]. The number of feasible solutions, the optimal solutions, the mean and variance of solutions, and other statistics may be extracted without difficulty [2].

In combinatorial optimization over graphs, a zero-suppressed binary decision diagram may correspond to a collection of subgraphs [2]. The edge sets that make up each subgraph differentiate the elements in the collection. It is thus understood on this account that the nodes in the diagram are consistent with the edges and not the vertices of the graph. Should the items from the subsets in Figure 1a stand for the edges of the graph in Figure 1b, the elements of the zero-suppressed binary decision diagram in Figure 1c would be the paths shown in Figure 1d.

Regarding diagram construction, a fundamental approach is the frontier-based search [4]. The algorithm inputs the needed information to the nodes as they are being generated. As a consequence, requirements for the stored subgraphs are set in conformity with the given problem. The practicality of the zero-suppressed binary decision diagram has been growing. Domains such as architectural planning [5], disaster preparedness [6], grid power loss minimization [7], and route finding [8] present diverse applications.



(a) A family of sets. (b) A fixed graph. (c) The diagram. (d) Some paths.

Fig. 1: Representing a collection of subgraphs.

Concisely, the study augments the frontier-based search for zero-suppressed binary decision diagram construction. The resulting algorithm accommodates additional constraints during computation. By the advanced routine, the generalized directed rural postman problem and its reverse are simultaneously solved for several transit networks. To justify the effectiveness of the method, numerical assessment is done in conclusion to the analysis.

An outline of the paper is as follows. A summary of the zero-suppressed binary decision diagram, the frontier-based search, and the generalized directed rural postman is provided in the second section. The third section lays out the methodology for investigation. Experiment results and reports on computational efficiency are in the fourth section. The fifth section emphasizes the contribution of the research and suggests directions for subsequent work.

II. Preliminaries

A. Zero-Suppressed Binary Decision Diagram

The subsection mainly follows the progression in [8]. A formal definition for the zero-suppressed binary decision diagram is below [1].

*Author to whom correspondence should be addressed.
Manuscript submitted April 3, 2020.
R. R. P. Tan is with the Division of Information Science, Nara Institute of Science and Technology, Nara Prefecture, Japan and the School of Science and Engineering, Ateneo de Manila University, National Capital Region, Philippines e-mail: tan.renzo_roel_perez.tp7@is.naist.jp, rrtan@ateneo.edu.

J. Kawahara is with the Graduate School of Informatics, Kyoto University, Kyoto Prefecture, Japan and the Division of Information Science, Nara Institute of Science and Technology, Nara Prefecture, Japan.

K. Ikeda is with the Division of Information Science, Nara Institute of Science and Technology, Nara Prefecture, Japan.

A. D. Garciano is with the School of Science and Engineering, Ateneo de Manila University, National Capital Region, Philippines.

K. S. S. See is with the Augmented Intelligence Pros, Incorporated, National Capital Region, Philippines.

Definition 1 (Zero-Suppressed Binary Decision Diagram)**.** Let $U = \{x_1, x_2, \ldots, x_{|U|}\}$ be a finite universe with ordered elements. For $x_k \in U$, $x_i < x_j$ if and only if $i < j$. A zero-suppressed binary decision diagram is a labeled directed acyclic graph satisfying the properties listed infra.

1) The root is the only node with indegree 0.
2) There are exactly two nodes with outdegree 0 called the 0-terminal and 1-terminal.
3) Each nonterminal node has exactly two outgoing arcs labeled by 0 and 1 and these are called the 0-arc and 1-arc, respectively.
4) For $j \in \{0, 1\}$, the destination node of the j-arc of a node n is called the j-child of n.
5) Each nonterminal node is labeled by an element of U.
6) The label of a nonterminal node is strictly smaller than those of its children.

Building on the definition, a zero-suppressed binary decision diagram represents a family $\mathcal{F}$ of subsets from U. More specifically, a path P from node n to n' in a diagram corresponds to a subset $U' \subseteq U$ if and only if for all $x \in U'$, there exists a node n'' labeled with x whose 1-arc is in P [9]. A subset $U' \subseteq U$ is an element of $\mathcal{F}$ if and only if there is a path from the root node to the 1-terminal to which U' corresponds [9]. One proceeds to a helpful theorem [1].

Theorem 1. Let $\mathcal{D}$ be a zero-suppressed binary decision diagram corresponding to family $\mathcal{F}$ with root node e. The root is either terminal or nonterminal.

1) If e is the 0-terminal then $\mathcal{F} = \emptyset$, the empty family.
2) If e is the 1-terminal then $\mathcal{F} = \{\emptyset\}$, the family containing only the empty set.
3) If e is nonterminal then the root node of $\mathcal{D}$ has two children. Let e_0 be the 0-child and e_1 be the 1-child. Denote the family with diagram rooted at e_i by $\mathcal{F}_i$. The family $\mathcal{F}$ may be written as $\mathcal{F}_0 \cup \left(\bigcup_{x \in \mathcal{F}_1} x \cup \{e\}\right)$.

To reiterate, connected to e by the 0-arc are the sets in $\mathcal{F}$ that do not contain e. In symbols, $\mathcal{F}_0 = \{x \mid x \in \mathcal{F}, e \notin x\}$. The sets in $\mathcal{F}$ that do contain e are connected through the 1-arc, meaning $\mathcal{F}_1 = \{x \setminus \{e\} \mid x \in \mathcal{F}, e \in x\}$.

The existence and uniqueness of a reduced zero-suppressed binary decision diagram with the fewest nodes is guaranteed for a given $\mathcal{F}$ [1]. Reduction in linear time with respect to the number of nodes is possible through an algorithm found in [2].

Remark 1. A zero-suppressed binary decision diagram is defined to be reduced whenever:

- There are no distinct nodes that have the same label, 0-child, and 1-child; and
- There is no node whose 1-child is the 0-terminal.

As the diagram is inherently recursive, it is suited for family algebra [1]. Operations on two families such as the intersection and union are easily computed in time proportional to the product of the number of nodes in each of the concerned diagrams [2]. Explicitly, let $|\mathcal{D}|$ denote the number of nodes in a diagram $\mathcal{D}$. The time complexity of the intersection of two diagrams $\mathcal{D}_1$ and $\mathcal{D}_2$ is $\mathcal{O}(|\mathcal{D}_1| \cdot |\mathcal{D}_2|)$; it is the same with the union.

In the context of combinatorial optimization, a diagram may contain sets pertinent to a particular setup. For a diagram

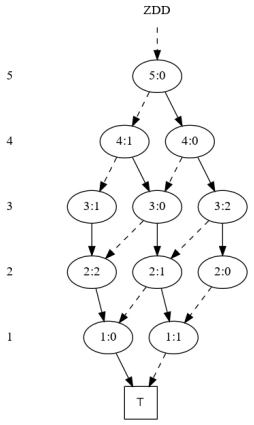


Fig. 2: The zero-suppressed binary decision diagram for the combination problem with $n = 5$ and $k = 3$. The 0-terminal and the arcs pointing to it are omitted.

$\mathcal{D}_1$, the solutions with maximum or minimum weight and the total number of solutions may be conveniently acquired with $\mathcal{O}(|\mathcal{D}_1|)$ complexity. The enumeration of feasible solutions takes time proportional to the number of solutions times the number of variables in $\mathcal{D}_1$.

Matters relevant to application and interpretation are tackled in the succeeding examples. Common problems in combinatorics are used as setting to explain the utility of the zero-suppressed binary decision diagram.

Example 1 (The Combination Problem)**.** Resolving ways in which k objects may be chosen from n distinct objects regardless of order is known as the combination problem. A single zero-suppressed binary decision diagram may represent the family of k-element subsets from a universe of cardinality n. Figure 2 holds all ways to select exactly 3 items from a set of 5 items.

In the diagram, all nodes are labeled by a variable number corresponding to an item and a node identification number for reference. A 1-arc is represented by a solid line and a 0-arc is represented by a dashed line. The former and latter correspond to the item being present or not, respectively. Any traversal from the root node to the 1-terminal with symbol $\top$ through a path formed by 1-arcs and 0-arcs is a 3-subset of the 5-element universe.

Example 2 (The Knapsack Problem)**.** A familiar combinatorial problem is the knapsack problem. Given a set of individually weighted and valued items, the goal is to identify which subset would have the maximum value while having a total weight not exceeding the prescribed weight. The decision-diagram-based solution approaches this formulation by first determining the subsets that satisfy the weight constraint. For $n = 5$ different items with assigned weight tuple $w = (12, 7, 11, 8, 9)$ and weight constraint $W = 26$, the rotated diagram is in Figure 3. Upon the incorporation of the value tuple $g = (24, 13, 23, 15, 16)$ into the problem, computing for the set with maximum total value and with weight not exceeding the limit is straightforward. The set formed by taking the second, third, and fourth items with total weight 26 has the maximum value of 51. The selection is consistent with the source text for the sample knapsack problem conditions [10].

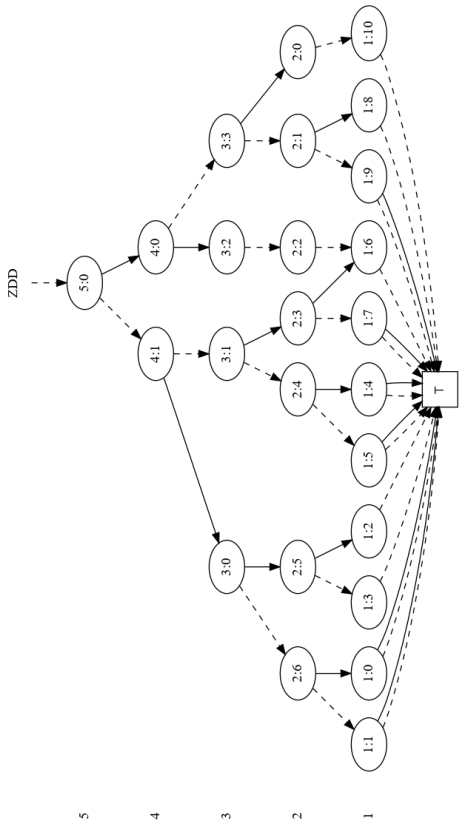


Fig. 3: The zero-suppressed binary decision diagram for the specified knapsack problem.

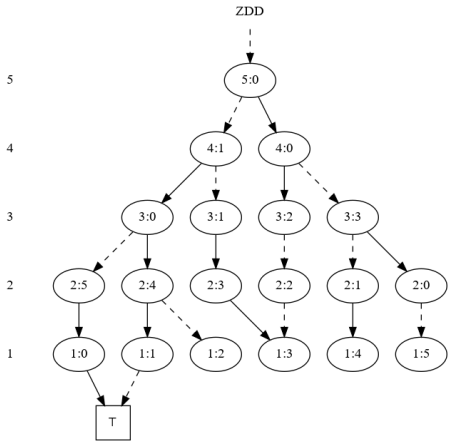


Fig. 4: The zero-suppressed binary decision diagram for the intersection of the combination and knapsack diagrams.

Furthermore, there would be no complications should one frame a version of the knapsack problem and impose the restriction that precisely 3 items may be carried. Noticeably, the formulation simply puts together the constraints for the combination and knapsack problems. Solving the problem is tantamount to taking the intersection of the two diagrams previously produced. Figure 4 shows the zero-suppressed binary decision diagram. It is discovered that only two 3-item subsets have total weight less than or equal to 26; there are only two paths from the root node to the 1-terminal. Among the two, one proves to be the optimal solution mentioned in the previous paragraph.

B. Frontier-Based Search

An algorithm for constructing a zero-suppressed binary decision diagram representing a defined set of subgraphs from a given graph is the frontier-based search. Through the aforementioned process, diagrams for subgraphs that are paths, trees, matchings, and others may be generated. The construction of the zero-suppressed binary decision diagram in a graph-theoretic setting is condensed consistent with that of [8]. Appropriately, one pays special attention to path representation. Refer to [4] for a complete discussion.

Consider $G = (V, E)$, an undirected edge-weighted graph with vertex set V and edge set $E = \{e_1, e_2, \ldots, e_m\}$. The set of edges E is a collection of 2-element subsets of V, each of which corresponds to a unique edge in G. Equivalently, an edge $e \in E$ is written as $\{v, w\}$, where $v, w \in V$. The graph G is also assumed to be simple and connected.

Further, a subgraph of G is defined as (V', E') with $V' = \bigcup_{e \in E'} e$ for $E' \subseteq E$. The union $\bigcup_{j=1}^{i} e_j$ is the set of vertices to which at least one of $e_1, e_2, \ldots, e_i$ is incident. This means that a subgraph has no vertex with degree 0; in general, an edge set determines the subgraph. The set E with ordered elements $e_1 < e_2 < \cdots < e_m$ is the universe of the zero-suppressed binary decision diagram.

As the algorithm begins, the root node is labeled as e_1. The breadth-first construction advances and creates nodes e_{i+1} after all nodes e_i have been generated for $i = 1, 2, \ldots, m-1$. The arcs of the nonterminal nodes e_i are ensured to point at nodes labeled e_{i+1}, the 0-terminal, or the 1-terminal. Concurrently, an array $n.\texttt{deg}$ mapping a particular subset of V to the set of natural numbers is stored into every node n. If $n.\texttt{deg}$ is equal to $n'.\texttt{deg}$ for two nodes n and n' then the two nodes are merged. Node sharing is the term coined for such a circumstance [4]. A subgraph with vertex degrees specified by $n.\texttt{deg}$ corresponds to a path from the root node to n.

Let e_i be the edge of largest index among all edges incident to a vertex v. After e_i is processed, $\texttt{deg}[v]$ is no longer referred to since the degree of v is independent of edges $e_{i+1}, e_{i+2}, \ldots, e_m$. The node v is then tagged as fixed [4]. The ith frontier is defined as $F_i = \left(\bigcup_{j=1}^{i} e_j\right) \cap \left(\bigcup_{j=i+1}^{m} e_j\right)$ for $i = 1, 2, \ldots, m-1$ and $F_0 = F_m = \emptyset$ [4]. For node n with label e_i, $n.\texttt{deg}[v]$ is stored into the node if and only if $v \in F_{i-1}$. Should there be violations in the degree conditions, pruning using the array $\texttt{deg}$ is done [4]. Through node sharing and pruning, the zero-suppressed binary decision diagram representing subgraphs of a prescribed type may be constructed; notwithstanding, more information than what is in $n.\texttt{deg}$ may be required for other kinds of subgraphs.

C. Generalized Directed Rural Postman

The definition of a path in graph theory is first delved into. It is essential in problem formulation.

Definition 2 (Path)**.** Given a graph, a path is a sequence of edges $y_1, y_2, \ldots, y_p$ where $y_i = \{v_{i-1}, v_i\}$ for $i = 1, 2, \ldots, p$ with vertex $v_i \neq v_j$ if $i \neq j$.

The question of finding the shortest journey in a metro network that uses each line at least once may be formulated as a known problem in operations research, the generalized directed rural postman [11]. The problem, along with the

Chinese postman [12] and the rural postman [13] problems, falls under the field of arc routing, which focuses on arc or edge properties rather than node features [14]. A definition derived from [15] follows.

Problem 1 (Generalized Directed Rural Postman)**.** Given a set of weights W, a set of colors C, a graph $G = (V, E)$, an edge-weighting function $w : E \to W$, and a coloring function $c : E \to C$, find path E' such that $\sum_{e \in E'} w(e)$ is minimum and $\bigcup_{e \in E'} \{c(e)\} = C$.

The generalized directed rural postman problem is nondeterministic polynomial-time hard [14]. Accordingly, another nondeterministic polynomial-time hard problem is the reverse [16]. By seeking the maximum instead of the minimum weight, the problem is distinctively named the crazy generalized directed rural postman. Both the original and the reverse problems are hired by the study to further demonstrate the efficiency and effectiveness of the technique.

Problem 2 (Crazy Generalized Directed Rural Postman)**.** Given a set of weights W, a set of colors C, a graph $G = (V, E)$, an edge-weighting function $w : E \to W$, and a coloring function $c : E \to C$, find path E' such that $\sum_{e \in E'} w(e)$ is maximum and $\bigcup_{e \in E'} \{c(e)\} = C$.

As an aside, an exact algorithm for solving the original problem is introduced in [15]. Given in [17] are improvements through a branch-and-cut solution. A more recent study is [11], providing a clever workaround for finding the optimal solution in reasonable time.

The constraints for the generalized directed rural postman problem and the crazy generalized directed rural postman problem may be relaxed in cases where no solutions are found. In lieu of paths, one may search for trails.

Definition 3 (Trail)**.** Given a graph, a trail is a sequence of edges $y_1, y_2, \ldots, y_p$ where $y_i = \{v_{i-1}, v_i\}$ for $i = 1, 2, \ldots, p$ with edge $y_i \neq y_j$ if $i \neq j$.

In a weighted graph, a minimal path is a path of minimum weight; a path of maximum weight is called a maximal path. Consistently, a trail of minimum weight is a minimal trail and a trail of maximum weight is a maximal trail.

III. Methodology

The proposed method was implemented in the C++ programming language aided by TdZdd[1], an existing library for diagram manipulation documented in [18] and [19]. The ZDDLines[2] repository, utilized in [8], served as basis for the program. In addition, the entirety of the code is committed to the GDRPDD[3] project.

Version 9.3.0 of g++ is used as compiler. Machine specifications include the Ubuntu 18.04.4 Long Term Support (Bionic Beaver) operating system, the Intel® Core™ i7-8565U processor at 1.80GHz, the NVIDIA® GeForce® MX250 graphics card, and a memory of 16GB.

[1]https://github.com/kunisura/TdZdd
[2]https://github.com/renzopereztan/ZDDLines
[3]https://github.com/renzopereztan/GDRPDD

A. Data Preprocessing

A sample grid and chosen metro networks of increasing size are encoded. The first four rows of the text data for the sample network is shown as example.

```
1 2 100 1
2 3 100 1
3 4 100 1
4 5 100 1
```

Each row represents an edge and its properties. The first two numbers are the start and end vertices that define the edge. The weight of the edge, typically measured in units of distance or time, is given by the third number. The fourth number specifies the category of which the edge is part; in the case of a metro, this would be the line by which the edge is labeled or to which the two linked stations belong. Coordinate data for all station positions are also gathered for the purpose of visualization.

B. Algorithm Implementation

For each vertex pair (s, t) in the network, the following steps are done. First, the diagram for all paths from s to t is created. The task is accomplished through designating a degree constraint using the frontier-based search. For vertices s and t, a vertex degree of 1 is set. The remaining vertices are forced to be of degree 2 or 0 depending on the involvement in s-t path generation. This guarantees subgraphs included in the diagram to be paths. For trails, the limitation is eased to accept even degrees greater than 2.

The requirement of having to use each metro line at least once is then addressed. The `Lines` class summarizing the approach in generating the diagram that satisfies the said restriction is seen in Appendix A. Nodes in a level in the zero-suppressed binary decision diagram created by the `Lines` class collectively correspond to an edge in the graph. Stored in every node is a bitmask indicating the line usage of the subgraph represented by a traversal from the root to the node. As the diagram is constructed from top to bottom, the states of the nodes are updated depending on the line information coming from the designated edge.

For a problem with color set $C = \{c_1, c_2, \ldots, c_{|C|}\}$, edge set $E = \{e_1, e_2, \ldots, e_m\}$, and coloring function $c : E \to C$, the bitmask is set to be of $|C|$ bits. Each bit ties in with the elements of C in order, implying that the kth bit indicates whether the color c_k is covered or not. The root node has a bitmask state with all bits being 0. For a node e_i, the 0-child retains state and the state of the 1-child is revised, flipping the bit designating $c(e_i)$ to 1 if it is 0. This happens routinely as one descends. A path from the root node ends in the 1-terminal if the resulting mask contains no bits with value 0, signifying all lines being utilized.

The intersection of the s-t path zero-suppressed binary decision diagram and the constraint diagram is taken afterward. The final step is to obtain the overall optimal paths for the graph. Attached as Appendices B and C are the `MinDist` and `MaxDist` classes based on [2] utilized to produce the paths with minimum and maximum weight.

IV. Experiments

The results are discussed in two parts. The outcome of the implementation based on the zero-suppressed binary decision

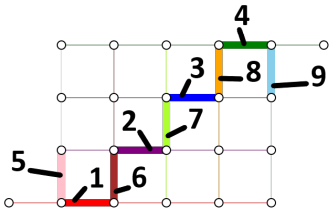


Fig. 5: A solution to the original problem for the sample grid.

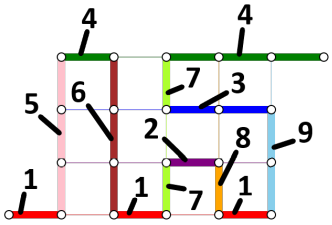


Fig. 6: A solution to the reverse problem for the sample grid.

diagram is in the first subsection. This is further segmented into the sample grid setting and the metro context. The second subsection replicates a standing method in literature for comparison. An integer linear program for the generalized directed rural postman problem is examined.

A. *Decision-Diagram-Based Approach*

The simple graph consists of 22 vertices and 33 edges in a mesh-like pattern. All edge weights are set to be 100 units. The 9 lines are determined by the vertical and horizontal lines that cut through the grid.

The generalized directed rural postman problem and its reverse are solved in 56.96 seconds. The recorded time includes the enumeration of all paths in the network that satisfy the category-based constraint and the extraction of the paths that have the minimum and maximum cumulative weights. Note that through the approach, even the solutions with the second smallest weight, the third largest weight, and so on, are known and may easily be recalled.

There are 2 unique solutions of total weight 900 units found for the minimum. Figure 5 shows one of them. A total of 19 different paths exist for the maximum. A solution with the maximum weight of 2100 units is seen in Figure 6.

1) The Hong Kong Metro: Composed of 98 vertices and 108 edges is the Hong Kong Mass Transit Railway, referred to in the paper as the Hong Kong Metro[4]. The network has 10 lines identified in Appendix E. For the purposes of the study, an edge is weighted one step. For every additional edge, the cost of the journey would increase by one step.

An out-turn of no solution was reached in 1010.67 seconds. No path that uses each line at least once exists for the metro. In response, relaxation is done. Trails of minimum and maximum weight that pass through all lines are sought. This problem-solving detour yields added complexity, hence the running time of 1172.33 seconds.

There are two minimal solutions. A trail with the minimum 35 steps is in Figure 7 and Table I. Two solutions of 56 steps

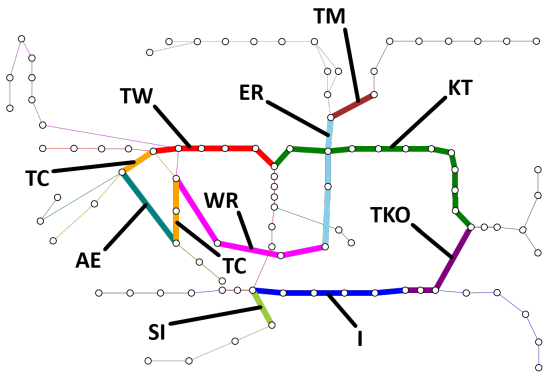


Fig. 7: A solution to the relaxed original problem for the Hong Kong Metro.

TABLE I: The Shortest Journey for the Hong Kong Metro

Move	Initial Station	Terminal Station	Line
1	Ocean Park	Admiralty	SI
2	Admiralty	North Point	I
3	North Point	Yau Tong	TKO
4	Yau Tong	Prince Edward	KT
5	Prince Edward	Lai King	TW
6	Lai King	Tsing Yi	TC
7	Tsing Yi	Kowloon	AE
8	Kowloon	Nam Cheong	TC
9	Nam Cheong	Hung Hom	WR
10	Hung Hom	Tai Wai	ER
11	Tai Wai	Che Kung Temple	TM

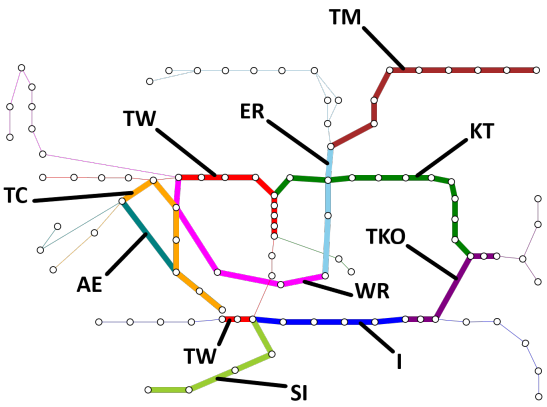


Fig. 8: A solution to the relaxed reverse problem for the Hong Kong Metro.

are maximal. Notice in Figure 8 that the algorithm forces itself to extend the trail in producing the maximum. Pairs of edges that are incident to the same vertices but are on different lines are likely to be taken as part of the trail.

2) The Osaka Metro: The Osaka Metro[5] is chosen for the second experiment. Previous studies [8] have used the same network to find solutions to a constrained version of the simpler *s-t* path problem. The metro has 107 vertices and 125 edges, a larger network compared to the Hong Kong

[4]https://en.wikipedia.org/wiki/MTR

[5]https://en.wikipedia.org/wiki/Osaka_Metro

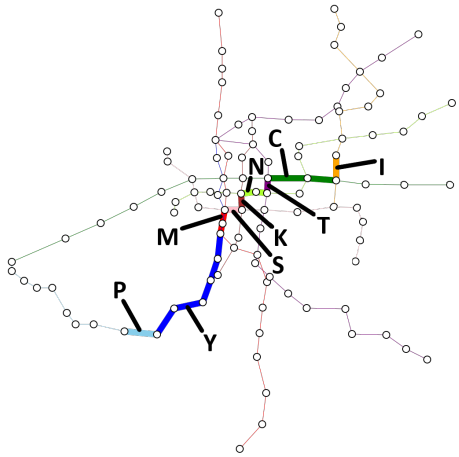


Fig. 9: The solution to the original problem for the Osaka Metro.

TABLE II: The Shortest Journey for the Osaka Metro

Move	Initial Station	Terminal Station	Line
1	Hirabayashi	Suminoekoen	P
2	Suminoekoen	Daikokucho	Y
3	Daikokucho	Namba	M
4	Namba	Nippombashi	S
5	Nippombashi	Nagahoribashi	K
6	Nagahoribashi	Tanimachi 6-Chome	N
7	Tanimachi 6-Chome	Tanimachi 4-Chome	T
8	Tanimachi 4-Chome	Midoribashi	C
9	Midoribashi	Shigino	I

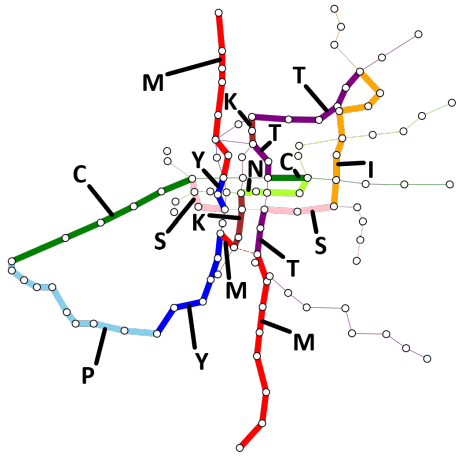


Fig. 10: A solution to the reverse problem for the Osaka Metro.

system. The edge weights are the distances between stations given in kilometers. The 9 lines are in Appendix F.

The time taken for the simultaneous solution of the two problems is 938.89 seconds. Interestingly, there is only one solution for the minimum. The path presented in Figure 9 and Table II with distance 16.2 kilometers is the optimal solution. For the crazy generalized directed rural postman problem, a maximum distance of 77.2 kilometers is registered. There are 30 ways to complete the journey as in Figure 10.

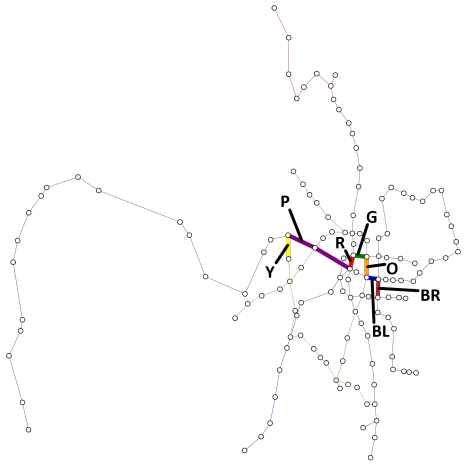


Fig. 11: A solution to the original problem for the Taipei Metro.

TABLE III: The Shortest Journey for the Taipei Metro

Move	Initial Station	Terminal Station	Line
1	Xing Fu	New Taipei Industrial Park	Y
2	New Taipei Industrial Park	Taipei Main	P
3	Taipei Main	Zhongshan	R
4	Zhongshan	Songjiang Nanjing	G
5	Songjiang Nanjing	Zhongxiao Xinsheng	O
6	Zhongxiao Xinsheng	Zhongxiao Fuxing	BL
7	Zhongxiao Fuxing	Daan	BR

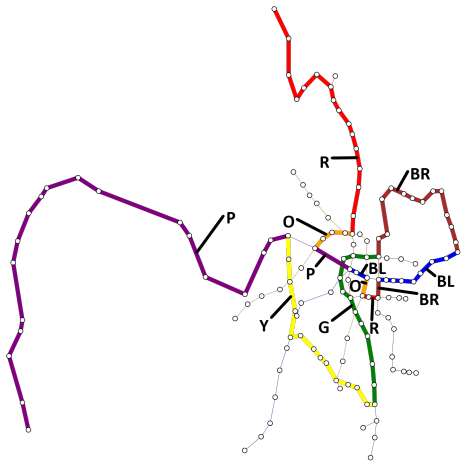


Fig. 12: A solution to the reverse problem for the Taipei Metro.

3) The Taipei Metro: The Taipei Metro[6], also known as the Taipei Mass Rapid Transit, comprises 136 vertices and 149 edges. One includes the Airport Line and considers 7 lines total listed in Appendix G for the network. The edges are simply weighted in units of steps.

The solution on the metro results in a computation time of 2176.70 seconds. The minimum of 8 steps and maximum of 90 steps each share two solutions. For the original problem, a solution is seen in Figure 11 and Table III; a solution for the reverse problem is in Figure 12.

[6]https://en.wikipedia.org/wiki/Taipei_Metro

B. Mathematical Programming

One pays particular attention to the generalized directed rural postman problem. Solving the crazy variant entails a similar strategy. The problem is posed as an integer program with linear constraints. A formulation motivated by [11] and [20] is employed.

Program 1 (Linear Program for the Generalized Directed Rural Postman Problem)**.** Minimize

$$\sum_{(u,v,l)\in A} x_{u,v,l}$$

subject to

$$\begin{aligned}
&\textstyle\sum_{(u,v,l)\in A} x_{u,v,l},\\
&\textstyle\sum_{(u,v,l)\in A} x_{u,v,l} = \sum_{(v,w,l)\in A} x_{v,w,l} \ \forall v \in V \setminus \{s,t\},\\
&\textstyle\sum_{(s,v,l)\in A} x_{s,v,l} = \sum_{(u,t,l)\in A} x_{u,t,l} = 1,\\
&\textstyle\sum_{(u,v,l)\in A} x_{u,v,l} \geq 1 \ \forall l \in C,\\
&|V| \cdot x_{u,v,l} \geq f_{u,v,l} \ \forall (u,v,l) \in A,\\
&\textstyle\sum_{(u,v,l)\in A} f_{u,v,l} - \sum_{(v,w,l)\in A} f_{v,w,l} \geq y_0 \ \forall v \in V \setminus \{s\},\\
&\textstyle\sum_{(u,v,l)\in A} x_{u,v,l} - \sum_{(v,w,l)\in A} x_{v,w,l} \leq y_v \ \forall v \in V,\\
&x_{u,v,l} \in \{0,1\} \ \forall (u,v,l) \in A,\\
&f_{u,v,l} \in N \ \forall (u,v,l) \in A, \text{ and}\\
&y_v \in N \ \forall v \in V.
\end{aligned}$$

For each edge connecting vertices u and v on line l, a binary variable $x_{u,v,l}$ is assigned. Instinctively, the sum of the variables is to be minimized. A virtual source s and a virtual target t, both of which are connected to every vertex in the station, are added to overcome the crux of having to solve the program for each possible vertex pair. One may consult [11] and [21] for a thorough explanation of the constraints and some possible refinements.

Upon completing the integer programming experiment, the correctness of the solutions attained by the decision-diagram-based algorithm is confirmed. From the optimal solutions to the number of solutions, results from both methods match. A complete iteration for a network, however, took several hours. It is important to note that this is primarily because of the need for solving the model multiple times to get the number of solutions that give minimum.

V. Conclusion

The paper has put forth a novel solution to the known generalized directed rural postman problem and the unconventional crazy generalized directed rural postman problem. The two nondeterministic polynomial-time hard problems were simultaneously solved by the decision-diagram-based procedure in minutes while a traditional integer program analogue consumed several hours on each problem. Moreover, the capacity for enumerating feasible solutions and the flexibility in adjustment further the relative superiority of the zero-suppressed binary decision diagram.

With regard to future work, the parallelization of the algorithm is recommended. The distribution of the vertex pairs may return a smaller computation time. To close, new applications for the zero-suppressed binary decision diagram and the frontier-based search are to be explored.

Appendix A
Code for the Line Constraint

```
class Lines: public tdzdd::DdSpec<Lines, int, 2> {
public:
        Lines(){}
        int getRoot(int& state)
        const{
          state = 0;
          return n;}
        int getChild(int& state, int level, int value)
        const{
              if(value == 1) state |= (1<<l[n-level]);
              level--;

              if(level == 0){
                      if(state == ((1<<L)-1)<<1){
                              return -1;
                      }else{
                              return 0;
                      }
              }
          return level;
        }
};
```

Appendix B
Code for Extracting the Minimum

```
class MinDist: public tdzdd::DdEval<MinDist,DistData> {
public:
        MinDist(){}
        void evalTerminal(DistData& data, bool one)
        const {
                data.val = one ? 0 : 100000;}
        void evalNode(DistData& data, int level,
        tdzdd::DdValues<DistData,2> const& values)
        const {
                const DistData& data0 = values.get(0);
                const DistData& data1 = values.get(1);
                if(data0.val <= data1.val + d[n-level]){
                        data.val = data0.val;
                        data.mask = data0.mask;
                }else{
                        data.val = data1.val + d[n-level];
                        data.mask = data1.mask;
                        data.mask[level-1] = 1;
                }
        }
};
```

Appendix C
Code for Extracting the Maximum

```
class MaxDist: public tdzdd::DdEval<MaxDist,DistData> {
public:
        MaxDist(){}
        void evalTerminal(DistData& data, bool one)
        const {
                data.val = one ? 0 : INT_MIN;}
        void evalNode(DistData& data, int level,
        tdzdd::DdValues<DistData,2> const& values)
        const {
                const DistData& data0 = values.get(0);
                const DistData& data1 = values.get(1);
                if(data0.val >= data1.val + d[n-level]){
                        data.val = data0.val;
                        data.mask = data0.mask;
                }else{
                        data.val = data1.val + d[n-level];
                        data.mask = data1.mask;
                        data.mask[level-1] = 1;
                }
        }
};
```

Appendix D
Some Experiment Statistics

Graph	$\|V\|$	$\|E\|$	$\|C\|$	Time (s)
Sample	22	33	9	56.96
Hong Kong	98	108	10	1172.33†
Osaka	107	125	9	938.89
Taipei	136	149	7	2176.70

†Computation time for the relaxed problem.

APPENDIX E
LINE REFERENCE FOR THE HONG KONG METRO

Line	Name
AE	Airport Express Line
ER	East Rail Line
I	Island Line
KT	Kwun Tong Line
TM	Tuen Ma Line
SI	South Island Line
TKO	Tseung Kwan O Line Line
TW	Tsuen Wan Line
TC	Tung Chung Line
WR	West Rail Line

APPENDIX F
LINE REFERENCE FOR THE OSAKA METRO

Line	Name
M	Midosuji Line
T	Tanimachi Line
Y	Yotsubashi Line
C	Chuo Line
S	Sennichimae Line
K	Sakaisuji Line
N	Nagahori Tsurumi-Ryokuchi Line
I	Imazatosuji Line
P	Nanko Port Town Line

APPENDIX G
LINE REFERENCE FOR THE TAIPEI METRO

Line	Name
BR	Wenhu Line
R	Tamsui-Xinyi Line
G	Songshan-Xindian Line
O	Zhonghe-Xinlu Line
BL	Bannan Line
Y	Circular Line
P	Airport Line

ACKNOWLEDGMENT

Gratitude is expressed to Florian Sikora for providing valuable insight leading to the completion of the paper. F. Sikora is with the Paris Dauphine University, Paris Sciences et Lettres University, Island of France Region, France.

Assistance in matters touching on implementation was received from Miguel Zenon Nicanor L. Saavedra. M. Z. N. L. Saavedra is with the School of Science and Engineering, Ateneo de Manila University, National Capital Region, Philippines and the Trainocate Group, Quintegral Limited, National Capial Region, Philippines.

The Japan Society for the Promotion of Science supports the research through the Grants-in-Aid for Scientific Research Program (KAKENHI 18K19821 and 18K04610).

REFERENCES

[1] S. Minato, "Zero-suppressed BDDs for set manipulation in combinatorial problems," *Proceedings of the 30th International Design Automation Conference*, Dallas, United States of America, 1993, pp. 272-277.

[2] D. Knuth, *The Art of Computer Programming Vol. 4 Fasc. 1*. Addison-Wesley, 2009.

[3] H. Iwashita, Y. Nakazawa, J. Kawahara, T. Uno, and S. Minato, "ZDD-based computation of the number of paths in a graph," *Hokkaido University Division of Computer Science TCS Technical Report TCS-TR-A-12-60*, 2012.

[4] J. Kawahara, T. Inoue, H. Iwashita, and S. Minato, "Frontier-based search for enumerating all constrained subgraphs with compressed representation," *IEICE Transactions on Fundamentals of Electronics, Communications, and Computer Sciences Vol. E100-A No. 9*, 2017, pp. 1773-1784.

[5] A. Takizawa, Y. Miyata, and N. Katoh, "Enumeration of floor plans based on a zero-suppressed binary decision diagram," *International Journal of Architectural Computing Iss. 1 Vol. 13*, 2015, pp. 25-44.

[6] A. Takizawa, Y. Takechi, A. Ohta, N. Katoh, T. Inoue, T. Horiyama, J. Kawahara, and S. Minato, "Enumeration of region partitioning for evacuation planning based on ZDD," *Proceedings of the 11th International Symposium on Operations Research and Its Applications in Engineering, Technology, and Management*, Huangshan, China, 2013, pp. 1-8.

[7] T. Inoue, K. Takano, T. Watanabe, J. Kawahara, R. Yoshinaka, A. Kishimoto, K. Tsuda, S. Minato, and Y. Hayashi, "Distribution loss minimization with guaranteed error bound," *IEEE Transactions on Smart Grid Vol. 5 No. 1*, 2014, pp. 102-111.

[8] R. Tan, J. Kawahara, A. Garciano, and I. Sin, "A zero-suppressed binary decision diagram approach for constrained path enumeration," *Lecture Notes in Engineering and Computer Science: Proceedings of the World Congress on Engineering 2019*, 3-5 July 2019, London, United Kingdom, pp. 132-136.

[9] S. Minato, "Zero-suppressed BDDs and their applications," *International Journal on Software Tools for Technology Transfer Vol. 3 No. 2*, 2001, pp. 156-170.

[10] D. Kreher and D. Stinson, *Combinatorial Algorithms: Generation, Enumeration, and Search*. Chemical Rubber Company Press, 2009.

[11] F. Sikora, "The shortest way to visit all metro lines in a city," *arXiv Electronic Preprint arXiv:1709.05948*, 2018.

[12] J. Edmonds and E. Johnson, "Matching, euler tours, and the chinese postman," *Mathematical Programming Vol. 5 No. 1*, 1973, pp. 88-124.

[13] J. Lenstra and A. R. Kan, "On general routing problems," *Networks Vol. 6 No. 3*, 1976, pp. 273-280.

[14] A. Corberan and G. Laporte, *Arc Routing: Problems, Methods, and Applications*. Society for Industrial and Applied Mathematics, 2015.

[15] M. Drexl, "On the generalized directed rural postman problem," *Journal of the Operations Research Society Vol. 65 No. 8*, 2014, pp. 1143-1154.

[16] M. Garey and D. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman and Company, 1979.

[17] T. Avila, A. Corberan, I. Plana, and J. Sanchis, "A new branch-and-cut algorithm for the generalized directed rural postman problem," *Transportation Science Vol. 50 No. 2*, 2016, pp. 750-761.

[18] H. Iwashita and S. Minato, "Efficient top-down ZDD construction techniques using recursive specifications," *Hokkaido University Division of Computer Science TCS Technical Report TCS-TR-A-13-69*, 2013.

[19] T. Toda, T. Saitoh, H. Iwashita, J. Kawahara, and S. Minato, "ZDDs and enumeration problems: State-of-the-art techniques and programming tool," *Computer Software Vol. 34 No. 3*, 2017, pp. 97-120.

[20] R. Miyashiro, T. Kasai, and T. Matsui, "Strictly solving the longest one-way ticket problem," *Proceedings of the 2000 Fall National Conference of the Operations Research Society of Japan*, Tokyo, Japan, 2000, pp. 24-25.

[21] P. Cerny, T. Henzinger, L. Kovacs, A. Radhakrishna, and J. Zwirchmayr, "Segment abstraction for worst-case execution time analysis," *Proceedings of Programming Languages and Systems: The 24th European Symposium of Programming*, 2015, pp. 105-131.