Exact Bernoulli Scan Statistics using Binary Decision Diagrams

Masakazu Ishihata[1] and **Takanori Maehara**[2]
[1]NTT Communication Science Laboratories
[2]RIKEN Center for Advanced Intelligence Project
masakazu.ishihata.ze@hco.ntt.co.jp, takanori.maehara@riken.jp

Abstract

In combinatorial statistics, we are interested in a statistical test of combinatorial correlation, i.e., existence a subset from an underlying combinatorial structure such that the observation is large on the subset. The combinatorial scan statistics has been proposed for such a statistical test; however, it is not commonly used in practice because of its high computational cost. In this study, we restrict our attention to the case that the number of data points is moderately small (e.g., 50), the outcome is binary, and the underlying combinatorial structure is represented by a zero-suppressed binary decision diagram (ZDD), and consider the problem of computing the p-value of the combinatorial scan statistics exactly. First, we prove that this problem is a #P-hard problem. Then, we propose a practical algorithm that solves the problem. Here, the algorithm constructs a binary decision diagram (BDD) for a set of realizations of the random variables by a dynamic programming on the ZDD, and computes the p-value by a dynamic programming on the BDD. We conducted experiments to evaluate the performance of the proposed algorithm using real-world datasets.

1 Introduction

Background and Motivation

Statistical test is one of the most important tools in decision-making under uncertainty. It evaluates how likely an observation occurs if a null hypothesis is true, and if it occurs very unlikely, we can reject the null hypothesis and accept the alternative hypothesis.

In this study, we consider a statistical test of "combinatorial correlation": Let V be a finite set, $\mathcal{F} \subseteq 2^V$ be a set family that represents a combinatorial structure, and $X_i \in \mathbb{R}$ be a random variable for each $i \in V$. Our purpose is to show whether $X = \{X_i \mid i \in V\}$ has a combinatorial correlation with respect to $\mathcal{F}$ with or not. Here, X has a combinatorial correlated with respect to $\mathcal{F}$ if there exists $S \in \mathcal{F}$ such that $\sum_{i \in S} X_i$ has a large value.

The combinatorial scan statistics [Lugosi, 2017] has been proposed for such a statistical test. Let $x_i \in \mathbb{R}$ be an observation of X_i and let $k = \max_{S \in \mathcal{F}} \sum_{i \in S} x_i$. As the null hypothesis, we assume that each X_i follows an independent distribution. Then, under the null hypothesis, the p-value, which is the probability that an observation would be greater than or equal to the actual observation k, is given by

$$P(K \geq k), \tag{1}$$

where

$$K = \max_{S \in \mathcal{F}} \sum_{i \in S} X_i. \tag{2}$$

The random variable K is referred as the *combinatorial scan statistics*[1]. If the p-value is sufficiently small, we can reject the null hypothesis and conclude that there is a combinatorial correlation.

The combinatorial scan statistics has several applications [Addario-Berry *et al.*, 2010]. However, it is not commonly used in practice because of its high computational cost (say, #P-hard; see also Theorem 2). [Arias-Castro *et al.*, 2008] studied a test with respect to the paths of some class of graphs; however, they only discussed the asymptotic performance of simple tests based on the average. [Arias-Castro *et al.*, 2011] studied a test with respect to the connected components in regular graphs; however, they did not discuss computational complexity. [Qian *et al.*, 2014] studied an efficient algorithm for computing the positive elevated mean scan statistic with respect to the connected components of a graph. Here, the positive elevated mean scan statistics does not require computing the probability; hence it avoids the #P-hard computation. They proposed a method based on a semidefinite programming relaxation and rounding, which scales up to $|V| \leq 300$ vertices; however, it has no theoretical guarantee about the approximation factor, which is not good for a statistical test. [Cadena *et al.*, 2017] studied an efficient algorithm for computing non-parametric scan statistics for the small connected components (say, at most 10) of a graph. Here, the non-parametric scan statistics also avoids the #P-hard computation. They employed a method based on the color coding technique [Alon *et al.*, 1995], which scales up to components of size $k \leq 10$; however, it is a randomized algorithm, which is not good for a statistical test, and its derandomization may be impractically expensive. It should be

[1][Lugosi, 2017] simply referred to this as the "scan statistics."

emphasized that all existing studies considered a particular type of combinatorial structures. Hence, these cannot be applied to more complicated (e.g., problem-specific-designed) combinatorial structures.

In this study, to overcome the computational difficulty, we restrict our attention to the following problem setting.

1. The number of regions, $|V|$, is small (e.g., ≤ 50).
2. The outcome is binary, i.e., $X_i \in \{0, 1\}$ for all $i \in V$.
3. The feasible domain $\mathcal{F}$ is generic, i.e., given as an input of the problem.

This simple setting has not been considered yet but includes several important applications such as the test of regional differences of a presidential election result in the United States.

Problem Formulation and Our Contribution

Given our problem settings 1 and 2, we focus on the *exact test* [Weerahandi, 2013] that computes the p-value of Eq. (1) *exactly* (cf: non-exact tests employ asymptotic theory or Monte-Carlo simulation to approximate the p-value). It is known that the exact test is preferred if there are only a small number of samples [MacKinnon, 2009]. Hence, it will fit our purpose.

Given our problem settings 1 and 3, we represent $\mathcal{F}$ as a *binary decision diagram (BDD)* [Bryant, 1992] or *zero-suppressed binary decision diagram (ZDD)* [Minato, 1993]. BDD/ZDD is a data structure that can represent a set family in a compact form. It has been applied in several problems such as paths, clusters, spanning trees, and so on. See [Minato, 2013] for a recent survey.

To summarize, our problem is the following.

Problem 1 (Exact Bernoulli Scan Statistics)**.** We are given a finite set V, a ZDD (or BDD) $\mathcal{D}$ that represents a feasible domain $\mathcal{F} \subseteq 2^V$, and probability $p_i \in [0, 1]$ for each $i \in V$. Let $X_i \in \{0, 1\}$ be random variable such that $X_i = 1$ with probability p_i and $X_i = 0$ otherwise. The task is to compute the p-value of Eq. (1) exactly for a given k.

In this study, we first show that Problem 1 is #P-hard (Section 3). Then, we propose a practical algorithm to solve Problem 1 (Section 4).

Our main technical contribution is an algorithm that constructs a BDD for the following set family:

$$\mathcal{W}_k = \{W \subseteq V : \max_{S \in \mathcal{F}} |S \cap W| \geq k\}. \tag{3}$$

We refer this the *weight family* given a feasible domain $\mathcal{F}$ and k because each $W \in \mathcal{W}_k$ corresponds to a "binary weight vector" $w = 1_W \in \{0, 1\}^V$ such that the optimal value of the linear maximization problem $\max_{S \in \mathcal{F}} \sum_{i \in S} w_i$ is at least k. We also refer its BDD the *weight BDD*. In the application to the combinatorial scan statistics, $\mathcal{W}_k$ corresponds to the set of realization of the random variable X such that the combinatorial scan statistics K of Eq. (2) is at least k. For a general $\mathcal{F}$, computing the weight BDD is a non-trivial task. On the other hand, if $\mathcal{F}$ is represented by a ZDD, the weight BDD is obtained by a dynamic programming on the ZDD (Section 4.1).

Once we have the weight BDD, the probability

$$P(\mathcal{W}_k) = \sum_{W \in \mathcal{W}_k} \prod_{i \in W} p_i \prod_{j \in V \setminus W} (1 - p_j) \tag{4}$$

is efficiently obtained by a dynamic programming. Since this value equals to the p-value of Eq. (1), we can solve Problem 1 efficiently in the time proportional to the size of the weight BDD (Section 4.2).

As a side-product of having the weight BDD, we can also compute the *post-selection inference* [Taylor and Tibshirani, 2015] (Section 4.3).

We demonstrate the effectiveness of our algorithm by conducting experiments on real-world datasets (Section 5). We employ a family of connected subsets of a 48 contiguous US states and a 47 Japanese prefectures as a feasible domain $\mathcal{F}$, and compute the scan statistics K of Eq. (2) and its p-value of Eq. (1) to test the localities of the changes of population, income, and GDP by state/prefecture. We also test the locality of the results of the 2016 US presidential election.

name	meaning	complexity
union	$\mathcal{S}_1 \cup \mathcal{S}_2$	$\lvert\mathcal{D}_1\rvert\lvert\mathcal{D}_2\rvert$
intersection	$\mathcal{S}_1 \cap \mathcal{S}_2$	$\lvert\mathcal{D}_1\rvert\lvert\mathcal{D}_2\rvert$
complement	$2^V \setminus \mathcal{S}$	$\lvert\mathcal{D}\rvert$
select(i)	$\mathcal{S}.\text{select}(i) := \{S \in \mathcal{S} : i \in S\}$	$\lvert\mathcal{D}\rvert$
add(i)	$\mathcal{S}.\text{add}(i) := \{S \cup \{i\} : S \in \mathcal{S}\}$	$\lvert\mathcal{D}\rvert$

Table 1: Some operations supported by BDDs/ZDDs. $\mathcal{S}$, $\mathcal{S}_1$, and $\mathcal{S}_\in$ are set families represented by BDDs/ZDDs $\mathcal{D}$, $\mathcal{D}_1$, and $\mathcal{D}_2$, respectively.

2 Preliminaries — BDD and ZDD

Let V be a finite set. A *BDD* [Bryant, 1992] is a directed acyclic graph $\mathcal{D} = (N(\mathcal{D}), A(\mathcal{D}))$, where $N(\mathcal{D})$ is the set of nodes containing a single root ρ and two terminals $\top$ and $\bot$, and $A(\mathcal{D})$ is the set of arcs. A path from the root to the $\top$ terminal corresponds to a subset of V in the following manner. Each non-terminal node $\alpha \in N(\mathcal{D}) \setminus \{\top, \bot\}$ is associated with an element $i \in V$ of the underlying finite set (represented by $\alpha.\text{label} = i$), and has exactly two out-going edges, called 1-arc and 0-arc. A node pointed by 1-arc and 0-arc of α are called 1-child and 0-child (denoted by $\alpha.1$ and $\alpha.0$), respectively. A path from the root to $\top$ represents a subset of V: the subset contains $i \in V$ (resp. does not contain i) if-and-only-if the path contains the 1-arc (resp. 0-arc) of α such that $\alpha.\text{label} = i$.

A BDD is *ordered* if V has a total order, and the element associated with α is smaller than that of the descendants of α. In the following, we only consider the ordered BDDs.

To make the diagram compact and canonical (i.e., uniquely determined), we make the following two assumptions called the *BDD reduction rule*.

1. No two nodes have the isomorphic descendants.
2. No node has the 1-arc and 0-arc that point to the same node.

An important property of BDD is that it admits efficient manipulations. We summarize the operations used in this study in Table 1.

ZDD [Minato, 1993] is a variant of BDD that employs the following assumptions called the *ZDD reduction rule* instead of the BDD reduction rule. Here, the first rule is the same as the BDD reduction rule, and the second rule is different.

1. No two nodes have the isomorphic descendants.
2. No node has the 1-arc that points $\perp$.

For each application, we have to choose BDD or ZDD as a representation of a feasible domain. BDD is more suitable for a set family with many irrespective elements. For example, BDD is suitable for a family of the edge subsets of a graph that makes two specified vertices connected because a subset consists of a path between the vertices and other vertices that are irrespective to the connectivity [Imai *et al.*, 1999; Maehara *et al.*, 2017]. On the other hand, ZDD is more suitable for a sparse set family (i.e., each subset contains a small number of elements). For example, ZDD is suitable for a family of the edge subsets of a graph that forms a path between two specified vertices [Knuth, 2009].

3 #P-Hardness of the Problem

We prove that Problem 1 is #P-hard.

Theorem 2. Problem 1 is #P-hard and has no FPRAS unless NP $=$ RP even if $k = 2$.

Proof. We reduce the #independent set problem[2] to Problem 1. Let $G = (V(G), E(G))$ be an instance of the #independent set problem (i.e., an undirected graph). We define $\mathcal{F} = \{\{u, v\} : (u, v) \in E(G)\}$, $p(u) = 1/2$ for all $u \in V(G)$ and $k = 2$. Note that the size of ZDD of $\mathcal{F}$ is bounded by $O(\sum_{S\in\mathcal{F}} |S|)$ because each path of the ZDD corresponds to a single set. In this case, this value is $O(|E(G)|)$, which is linear in the size of the input.

The combinatorial scan statistics K in Eq. (2) satisfies $K \leq 1$ if and only if for all $(u, v) \in E(G)$, $X_u + X_v \leq 1$ holds. In other words, $\{u \in V(G) : X_u = 1\}$ forms an independent set of G. Therefore, $2^{|V(G)|}P(K \leq 1) = 2^{|V(G)|}(1 - P(K \geq 2))$ is the number of independent sets of G. $\square$

This result implies that Problem 1 is hard for large instances. On the other hand, if the size of the instance is small (e.g., ≤ 50), there remains a hope to obtain a solution by a practically fast algorithm.

4 Exact Algorithm

In this section, we propose an algorithm to solve Problem 1. We assume that the feasible domain $\mathcal{F}$ is represented by a ZDD $\mathcal{D}$ because ZDD is more suitable than BDD to represent a sparse set family, such as paths and connected components, which are typical examples of underlying combinatorial structures of the combinatorial correlation tests. The algorithm can be easily adapted to the case that $\mathcal{F}$ is represented by a BDD.

Our goal is to construct a compact representation of the set family $\mathcal{W}_k$ in Eq. (3). We observe that this set family contains many irrespective elements because if W is in $\mathcal{W}_k$, any superset of W is also in $\mathcal{W}_k$. Therefore, we employ BDD to represent $\mathcal{W}_k$.

Our algorithm consists of two steps: (1) construct a BDD of $\mathcal{W}_k$, and (2) compute the p-value of the combinatorial scan statistics using the BDD of $\mathcal{W}_k$. The second part uses a well-known dynamic programming technique. Thus, the novelty of this study is in the first part.

4.1 Weight BDD Construction

The algorithm is a dynamic programming on the ZDD $\mathcal{D}$ that manipulates the realized weight family $\mathcal{W}_k$ stored by BDDs.

Recall that each $\alpha \in N(\mathcal{D})$ induces a ZDD rooted by α, which represents a set family $\mathcal{F}_\alpha \subseteq 2^{\{\alpha.\text{label},\ldots,|V|\}}$. Note that $\mathcal{F}_\alpha \neq \emptyset$ for all $\alpha \neq \perp$ because of the reduction rule. Also, by the definition of ZDD, we have the following relations

$$\mathcal{F}_\alpha = \mathcal{F}_{\alpha.1}.\text{add}(\alpha.\text{label}) \cup \mathcal{F}_{\alpha.0}, \tag{5}$$

$$\mathcal{F}_{\alpha.1} = \{S \setminus \{\alpha.\text{label}\} \in \mathcal{F}_\alpha : \alpha.\text{label} \in S\}, \tag{6}$$

$$\mathcal{F}_{\alpha.0} = \{S \in \mathcal{F}_\alpha : \alpha.\text{label} \notin S\}, \tag{7}$$

where the union in Eq. (5) is a disjoint union (i.e., the first term and the second term are disjoint).

The algorithm maintains a BDD of the set family

$$\mathcal{W}[\alpha][l] := \{W \subseteq V : \max_{S\in\mathcal{F}_\alpha} |S \cap W| \geq l\} \tag{8}$$

for each $\alpha \in N(\mathcal{D})$ and $\ell \in \{0, \ldots, k\}$; namely, $\mathcal{W}[\rho][k] = \mathcal{W}_k$ where ρ is the root of ZDD $\mathcal{D}$, i.e., $\mathcal{F}_\rho = \mathcal{F}$. For the initial condition, we have

$$\mathcal{W}[\alpha][0] = 2^V, \quad \alpha \in N(\mathcal{D}) \setminus \{\perp\} \tag{9}$$

$$\mathcal{W}[\top][\ell] = \emptyset, \quad \ell \in \{1, \ldots, k\}, \tag{10}$$

$$\mathcal{W}[\perp][\ell] = \emptyset, \quad \ell \in \{0, \ldots, k\}. \tag{11}$$

Here, Eq. (9) follows because $\mathcal{F}_\alpha \neq \emptyset$ since the ZDD is reduced. Eq. (10) follows by the definition. Eq. (11) follows because $\mathcal{F}_\perp = \emptyset$. For the induction part, we have

$$\begin{aligned}\mathcal{W}[\alpha][l] = \mathcal{W}[\alpha.0][l] \cup \mathcal{W}[\alpha.1][l] \\ \cup\, \mathcal{W}[\alpha.1][l-1].\text{select}(\alpha.\text{label}).\end{aligned} \tag{12}$$

These are computed in the reverse topological order of $\mathcal{D}$. To verify this equation, we use the relations Eq. (5), Eq. (6), and Eq. (7). $W \in \mathcal{W}[\alpha][l]$ if and only if there exists $S \in \mathcal{F}_\alpha$ such that $|S \cap W| \geq l$. By Eq. (5), there are two cases: (1) $S = S' \cup \{\alpha.\text{label}\}$ for some $S' \in \mathcal{F}_{\alpha.1}$, or (2) $S \in \mathcal{F}_{\alpha.0}$. In case (2), W is in $\mathcal{W}[\alpha.0][l]$, which is captured by the first term of the right-hand side of Eq. (12). In case (1), there are two cases: (1-1) $|S' \cap W| \geq l$ or (1-2) not. In case (1-1), W is in $\mathcal{W}[\alpha.1][l]$, which is captured by the second term of the right-hand side of Eq. (12). In case (1-2), W satisfies that $|S' \cap W| = l - 1$ and $\alpha.\text{label} \in W$, which is captured by the third term of the right-hand side of Eq. (12). Consequently, the weight family $\mathcal{W}_k$ can be constructed by recursive evaluations of Eq. (12).

Our algorithm construct the weight BDD $\mathcal{B} = (N(\mathcal{B}, A(\mathcal{B})))$ for the weight family $\mathcal{W}_k$ by dynamic programming on ZDD $\mathcal{D}$. Let $\mathcal{B}[\alpha][l] \in N(\mathcal{B})$ be a BDD node such that its corresponding set family is $\mathcal{W}[\alpha][l]$; $\mathcal{B}[\rho][k]$ represents $\mathcal{W}_k$. Since all logical operations required in Eq. (12)

[2] The *#independent set problem* asks the number of independent set of a given graph $G = (V(G), E(G))$, where a subset $I \subseteq V$ is an independent set if $|I \cap \{u, v\}| \leq 1$ for all $(u, v) \in E(G)$. This problem is known to be #P-hard and has no FPRAS unless NP $=$ RP [Jerrum, 2003].

Algorithm 1 Construct the weight BDD $\mathcal{B}$

```
    Input: ZDD $\mathcal{D} = (N(\mathcal{D}), A(\mathcal{D}))$ rooted by $\rho$
1: $\mathcal{B}[\alpha][0] \leftarrow \top$ for all $\alpha \in N(\mathcal{D}) \setminus \{\bot\}$
2: $\mathcal{B}[\top][l] \leftarrow \bot$ for all $l \in \{1, \dots, k\}$
3: $\mathcal{B}[\bot][l] \leftarrow \bot$ for all $l \in \{0, \dots, k\}$
4: for $\alpha \in N(\mathcal{D}) \setminus \{\top, \bot\}$ in the reverse topological order of $\mathcal{D}$ do
5:     $\mathcal{B}[\alpha][l] \leftarrow \mathcal{B}[\alpha.0][l] \cup \mathcal{B}[\alpha.1][l] \cup \mathcal{B}[\alpha.1][l-1]$.select($\alpha$.label) for all $l \in \{1, \dots, k\}$
6: end for
7: return the BDD induced by $\mathcal{B}[\rho][k]$ as $\mathcal{B}$
```

Algorithm 2 Compute the probability $P(\mathcal{W}_k)$ on $\mathcal{B}$

```
    Input: BDD $\mathcal{B} = (N(\mathcal{B}), A(\mathcal{B}))$ rooted by $\sigma$
1: $P[\top] \leftarrow 1, P[\bot] \leftarrow 0$
2: for $\beta \in N(\mathcal{B}) \setminus \{\top, \bot\}$ in the reverse topological order of $\mathcal{B}$ do
3:     $P[\beta] = (1 - p_{\beta.\text{label}})P[\beta.0] + p_{\beta.\text{label}}P[\beta.1]$
4: end for
5: return $P[\sigma]$
```

are supported by BDD operations as shown in Table 1, $\mathcal{B}$ can be constructed by recursive evaluations of Eq. (12) by BDD operations. For any $\mathcal{B}[\alpha][l]$ and $\mathcal{B}[\alpha'][l']$, let $\mathcal{B}[\alpha][l] \cup \mathcal{B}[\alpha'][l']$ return a BDD node representing $\mathcal{W}[\alpha][l] \cup \mathcal{W}[\alpha'][l']$ and also let $\mathcal{B}[\alpha][l]$.select(α.label) return a BDD node representing $\mathcal{W}[\alpha][l]$.select(α.label). Then, the overall algorithm for constructing the weight BDD is shown in Algorithm 1.

By Theorem 2, unless P = NP, the size of the weight BDD is not bounded by a polynomial in the size of the input (otherwise, by Algorithm 2 described below, we can solve Problem 1 in polynomial time). Therefore, Algorithm 1 is not a polynomial time algorithm. However, in practice, it works well on small size instances; see Section 5.2.

4.2 p-Value of Combinatorial Scan Statistics

Next, we show an algorithm to compute the p-value Eq. (1) exactly. As mentioned in Section 1, our goal is to compute $P(\mathcal{W}_k)$ defined in Eq. (4). This is performed by a standard dynamic programming technique [Ishihata *et al.*, 2010].

Given the weight BDD $\mathcal{B} = (N(\mathcal{B}), A(\mathcal{B}))$, the algorithm maintains a value $P[\beta]$ that represents the probability of the set family represented by a BDD rooted by $\beta \in N(\mathcal{B})$. For the initial condition, we have

$$P[\top] = 1, \quad P[\bot] = 0. \tag{13}$$

For the induction part, we have

$$P[\beta] = (1 - p_{\beta.\text{label}})P[\beta.0] + p_{\beta.\text{label}}P[\beta.1]. \tag{14}$$

As same as Algorithm 1, we compute the values in the reverse topological order. Then, at the root node $\sigma \in N(\mathcal{B})$, we have

$$P(K \geq k) = P(\mathcal{W}_k) = P[\sigma]. \tag{15}$$

Algorithm 2 shows the complete process. The runtime of the algorithm is proportional to the size of the BDD $|\mathcal{B}|$.

4.3 Post-Selection Inference

Here, we show that Algorithm 1 can also be used for the post-selection inference.

Suppose that we have observations x_i for each $i \in V$. Then, we compute a *combinatorial hotspot* $H \in \operatorname{argmax}_{S \in \mathcal{F}} \sum_{i \in S} x_i$. Note that, in our case, H is obtained in linear time by a similar algorithm to Algorithm 2. The post-selection inference requires to compute the conditional probability

$$P\left(\sum_{i \in H} X_i \geq k \mid H \in \operatorname*{argmax}_{S \in \mathcal{F}} \sum_{i \in S} X_i\right). \tag{16}$$

To compute this quantity, we construct BDDs of set family

$$\mathcal{W}_l(H) = \{W \subseteq V : |H \cap W| \geq l\} \tag{17}$$

for each $l \in \{0, \dots, k_{\max}\}$, where $k_{\max} = \max_{S \in \mathcal{F}} |S|$. This set family represents the set of realizations of X such that $\sum_{i \in H} X_i \geq l$. Note that this set family is $\mathcal{W}_k$ in the weight family for the singleton set family $\mathcal{F} = \{H\}$. Therefore, we can use Algorithm 1 to construct the BDDs.

Once the BDDs are obtained, the set of realization of X such that $H \in \operatorname{argmax}_{S \in \mathcal{F}} \sum_{i \in S} X_i$ is obtained by

$$\mathcal{W}_{\text{OPT}}(H) := \bigcup_{l=0}^{k_{\max}} \left((2^V \setminus \mathcal{W}_{l-1}) \cap \mathcal{W}_l(H) \right). \tag{18}$$

Using these families, the probability Eq. (16) is computed by

$$\frac{P(\mathcal{W}_k(H) \cap \mathcal{W}_{\text{OPT}}(H))}{P(\mathcal{W}_{\text{OPT}}(H))}. \tag{19}$$

5 Experiments

We conducted experiments to evaluate the performance of the algorithm and to demonstrate applications to a real-world problem. All code was implemented in C/C++ (gcc 7.3.0 with the -O3 option) using SAPPOROBDD library[3] and TdZdd library[4]. All experiments were conducted on 64-bit Ubuntu 18.04.2 LTS with an Intel Core i7-7700K 3.6 GHz CPU and 16 GB RAM.

5.1 Experimental Setting

We apply our algorithm to test the locality of real-world observations: the population, income, and GDP changes of US and Japan, and the result of the 2016 US presidential election.

To test the locality, we employed a family of connected subsets of states/prefectures as a feasible domain $\mathcal{F}$. Let G^{US} (resp. G^{JP}) be an undirected graph representing 48 contiguous states in North America (resp. 47 prefecture in Japan), and let $\mathcal{F}_\ell^{\text{US}}$ (resp. $\mathcal{F}_\ell^{\text{JP}}$) be a family of subsets of connected states (resp. prefectures) on G^{US} (resp. G^{JP}) of size ℓ. We constructed ZDDs $\mathcal{D}_\ell^{\text{US}}$ and $\mathcal{D}_\ell^{\text{JP}}$ of $\mathcal{F}_\ell^{\text{US}}$ and $\mathcal{F}_\ell^{\text{JP}}$, respectively, by the frontier-based construction method [Kawahara *et al.*, 2017]. We omit the details of the construction time of the ZDDs since those are less than a second.

We obtained the estimated amounts of population, income, and GDP by state/prefecture from American FactFinder [5] and

[3]https://github.com/takemaru/graphillion/tree/master/src/SAPPOROBDD

[4]https://github.com/kunisura/TdZdd

[5]https://factfinder.census.gov/faces/nav/jsf/pages/index.xhtml

Index	US	Japan
Population	2012–2016	1976–2016
Income	2010–2017	2002–2014
GDP	1998–2017	2002–2014
# observations	32	67

Table 2: The number of the obtained observations of population, income, and GDP change of US and Japan.

e-Stat [6], official portal sites of US and Japanese governmental statistics, and computed the amount of change of each index. Table 2 shows information about the observations. We also obtained the result of the 2016 US presidential election from Wikipedia [7]

5.2 Growth of the Weight BDD

Let $\mathcal{W}_{\ell,k}^{\text{US}}$ and $\mathcal{W}_{\ell,k}^{\text{JP}}$ be the weight families of $\mathcal{F}_{\ell}^{\text{US}}$ and $\mathcal{F}_{\ell}^{\text{JP}}$, respectively. Given ZDDs $\mathcal{D}_{\ell}^{\text{US}}$ and $\mathcal{D}_{\ell}^{\text{JP}}$, we constructed a *shared* weight BDD of $\{\mathcal{W}_{\ell,k}^{\text{US}} \mid k \in \{0,\ldots,48\}\}$ for $\ell \in \{2,\ldots,48\}$ and one of $\{\mathcal{W}_{\ell,k}^{\text{JP}} \mid k \in \{0,\ldots,47\}\}$ for $\ell \in \{2,\ldots,47\}$. Here, a shared BDD is a compact representation of multiple BDDs that shares their isomorphic subgraphs[8]. We denote the above shared weight BDDs by $\mathcal{B}_{\ell}^{\text{US}}$ and $\mathcal{B}_{\ell}^{\text{JP}}$, respectively.

Figure 1 show that the relationship between the ZDD sizes $|\mathcal{D}_{\ell}^{\text{US}}|$ and $|\mathcal{D}_{\ell}^{\text{US}}|$, the corresponding shared weight BDD sizes $|\mathcal{B}_{\ell}^{\text{US}}|$ and $|\mathcal{B}_{\ell}^{\text{JP}}|$, and their construction times. The size of the shared weight BDDs is 10^5 times larger than that of BDDs. The construction time of the shared weight BDD is proportional to its size. The longest construction time of the shard weight BDD of Japan is less than 10 seconds, and one of US is less than 13 minutes. Once we constructed $\mathcal{B}_{\ell}^{\text{US}}$ and $\mathcal{B}_{\ell}^{\text{JP}}$, we can compute the p-value of Eq. (1) in a few seconds.

Using the ZDD $\mathcal{D}_{\ell}^{\text{US}}$ and its shared weight BDD $\mathcal{B}_{\ell}^{\text{US}}$, we can evaluate $|\mathcal{F}_{\ell}^{\text{US}}|$ and $|\mathcal{W}_{\ell,k}^{\text{US}}|$ in time proportional to the sizes of the ZDD and BDD. For instance, $|\mathcal{F}_{20}^{\text{US}}| =$ 14,607,877,196 whereas $|N(\mathcal{D}_{\ell}^{\text{US}})| =$ 8,966, and $|\mathcal{W}_{20,20}^{\text{US}}| =$ 59,466,160,560,888 whereas $|N(\mathcal{B}_{\ell}^{\text{US}})| =$ 38,901,120. This means that ZDDs/BDDs represent the set families very compactly, and without using them, it will be intractable to compute the p-values.

5.3 p-Values of Artificial Settings

We computed p-values of the following setting: For each $\ell \in \{2,\ldots,|V|-1\}$, we observed X such that $|X| = \ell$ and its scan statistics K was also ℓ, that is, all states (or prefectures) with value 1 were connected. Then, we computed the p-value of the above observation X, where we set p_i as the empirical probability $\ell/|V|$ for each $i \in V$. The p-value is the probability to observe 1s that form a connected subarea of size ℓ.

[6]https://www.e-stat.go.jp/en

[7]https://en.wikipedia.org/wiki/2016UnitedStatespresidentialelection

[8]Many BDD libraries support shared BDDs and we can construct shared weight BDDs by running our algorithm using a such library.

p-value	US dataset		Japan dataset	
$[0.0, 0.1)$	0	(0.00%)	9	(0.79%)
$[0.1, 0.2)$	1	(0.21%)	19	(1.68%)
$[0.2, 0.3)$	0	(0.00%)	17	(1.50%)
$[0.3, 0.4)$	7	(1.44%)	38	(3.35%)
$[0.4, 0.5)$	15	(3.09%)	53	(4.67%)
$[0.5, 0.6)$	149	(30.66%)	469	(41.36%)
$[0.6, 0.7)$	87	(17.90%)	121	(10.67%)
$[0.7, 0.8)$	10	(2.06%)	59	(5.20%)
$[0.8, 0.9)$	22	(4.53%)	102	(8.99%)
$[0.9, 1.0)$	195	(40.12%)	247	(21.78%)
Total	486		1134	

Table 3: The histograms of p-values of US and Japan datasets.

Figure 2 shows the relationship between ℓ and the p-value of US and Japan, respectively. The p-values of US with $\ell \in \{4,\ldots,20\}$ and those of Japan with $\ell \in \{3,\ldots,28\}$ are less than 0.05. This results infer that the probability of obtaining a connected subarea on Japan is smaller than one on US; it is natural because the width of G^{JP} is narrower than that of G^{US}.

5.4 Population, Income, and GDP Changes in US and Japan

We computed the p-value of 36 and 67 observations shown in Table 2 given a feasible domain $\mathcal{F}_{\ell}^{\text{US}}$ and $\mathcal{F}_{\ell}^{\text{JP}}$ with changing $\ell \in \{5, 10, 15, 20, 25, 30, 35, 40, 45\}$.

From each "amount of change" observation, we generated two types of "binary" observations: the first one is that a positive value (i.e., increasing) is 1 and a negative value (i.e., decreasing) is 0, and the second one is just opposite. To test a null hypothesis "the binary observation is generated from an i.i.d. distribution", we compute the p-value given the observation and a particular ℓ: the size of connected subareas. In this experiment, we set p_i of observation X to the empirical probability $|X|/|V|$ that is the ratio of 1s in the observation X. Consequently, we computed p-values of $32 \times 2 \times 9 = 572$ hypotheses of US, and $67 \times 2 \times 9 = 1,206$ hypotheses of Japan.

Table 3 shows the histograms of p-values, where we omitted the results of trivial observations that consist only of 1s or 0s. 95% of the computed p-values of US and 88% of those of Japan are greater than 0.5, that is, most of observations have no locality with high probability.

Figure 3 and 4 are the observations with the minimum p-value of US and Japan, respectively. Figure 3 indicates that red states had negative population change in 2016, and its p-value with $\ell = 5$ was 0.107. Figure 4 indicates that red prefectures had positive population change in 1999, and its p-value with $\ell = 15$ was 0.030. If we would test the above two hypotheses with significance level 0.05, the first one could not be rejected but the second one could be. Consequently, all 572 hypotheses of US could not be rejected because we exactly compute the p-value.

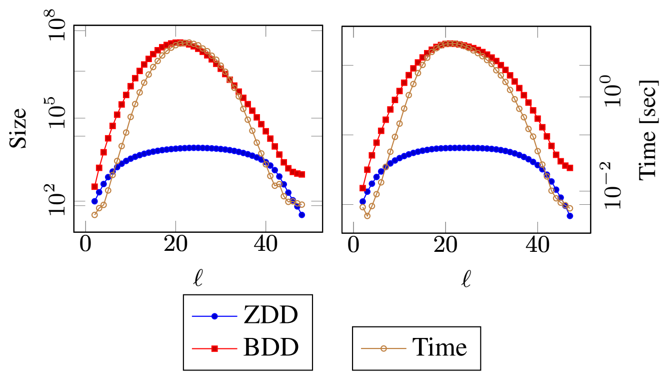

Figure 1: Sizes of the ZDD and the weight BDD with the BDD construction time for the 48 states in the United States (left) and 47 prefectures in Japan (right).

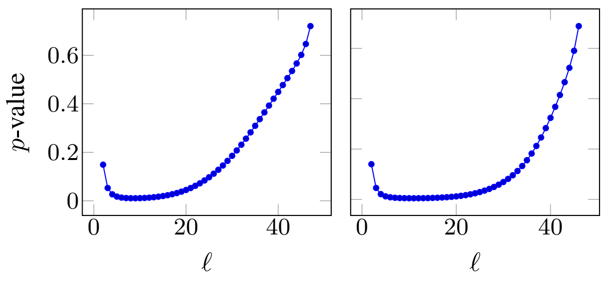

Figure 2: The relationship between ℓ and the p-value on the United States (left) and Japan (right). When $\ell = 9$, the p-value achieved the smallest value 0.0106 in the United States and when $\ell = 11$, the p-value achieved the smallest value 0.00452 in Japan.

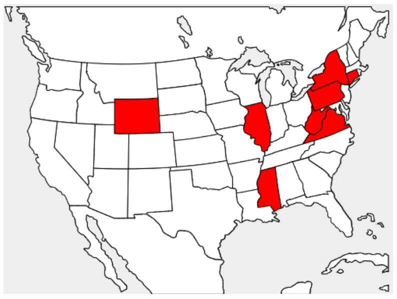

Figure 3: The observation with the minimum p-value in US dataset. Red states indicate the populations of those states decreased in 2016. The p-value of this observation with $\ell = 5$ was 0.107. It is not significant if we set the significant level at 0.05.

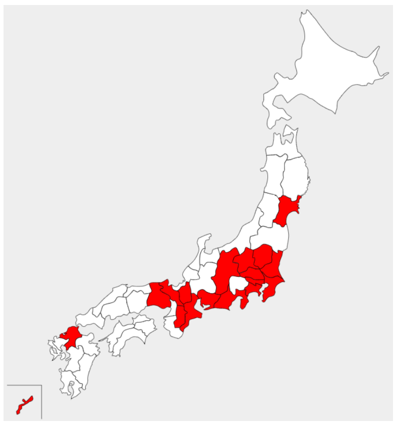

Figure 4: The observation with the minimum p-value in Japanese dataset. Red prefectures indicate the populations of those prefecture increased in 1999. The p-value of this observation with $\ell = 15$ was 0.030. It is significant if we set the significant level at 0.05.

5.5 2016 US Presidential Election

We computed the p-value of the results of the 2016 US presidential election. Figure 5 shows the result of the election: the red state indicates that Donald J. Trump won and the blue state indicates Hillary Clinton won. The number of red states is 29 and one of blue states is 19. Table 4 shows the p-values of red states and blue states with changing $\ell \in \{5, 10, 15, 20, 25, 30\}$. All computed p-values are greater than 0.3, that is, it is difficult to conclude the results of the election have locality.

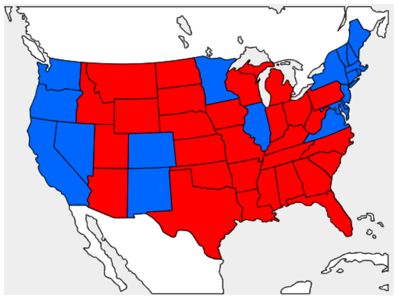

Figure 5: The results of the 2016 US presidential election. Red states carried by the Republicans and blue states did by the Democrats. The number of red states is 29 and one of blue states is 19.

	Blue States		Read States	
ℓ	K	p-value	K	p-value
5	5	0.879	5	0.999
10	10	0.382	10	0.954
15	13	0.682	15	0.806
20	16	0.691	20	0.604
25	19	0.528	25	0.335
30	19	0.555	29	0.346

Table 4: The combinatorial scan statistics K and its p-value of red and blue states with changing ℓ. There is no p-value significantly small.

6 Conclusion

We studied a problem of computing the p-value of the combinatorial scan statistics exactly. We restrict our attention to the case that the number of data points is moderately small, outcome is binary, and a feasible domain is represented by a ZDD. We first showed that the problem is #P-hard. Then, we proposed a practical algorithm that constructs a BDD for a set of the weight vectors by a dynamic programming on the ZDD, then computes the p-value by a dynamic programming on the BDD. We conducted experiments to evaluate the performance of the algorithm. The algorithm computes the p-values in a minute when the feasible domain is the connected components of the graphs of the states in North America and the prefectures in Japan.

References

[Addario-Berry *et al.*, 2010] Louigi Addario-Berry, Nicolas Broutin, Luc Devroye, Gábor Lugosi, et al. On combinatorial testing problems. *The Annals of Statistics*, 38(5):3063–3092, 2010.

[Alon *et al.*, 1995] Noga Alon, Raphael Yuster, and Uri Zwick. Color-coding. *J. ACM*, 42(4):844–856, 1995.

[Arias-Castro *et al.*, 2008] Ery Arias-Castro, Emmanuel J Candès, Hannes Helgason, Ofer Zeitouni, et al. Searching for a trail of evidence in a maze. *The Annals of Statistics*, 36(4):1726–1757, 2008.

[Arias-Castro *et al.*, 2011] Ery Arias-Castro, Emmanuel J Candes, Arnaud Durand, et al. Detection of an anomalous cluster in a network. *The Annals of Statistics*, 39(1):278–304, 2011.

[Bryant, 1992] Randal E Bryant. Symbolic boolean manipulation with ordered binary-decision diagrams. *ACM Computing Surveys (CSUR)*, 24(3):293–318, 1992.

[Cadena *et al.*, 2017] Jose Cadena, Feng Chen, and Anil Vullikanti. Near-optimal and practical algorithms for graph scan statistics. In *Proceedings of the 2017 SIAM International Conference on Data Mining*, pages 624–632. SIAM, 2017.

[Imai *et al.*, 1999] Hiroshi Imai, Kyoko Sekine, and Keiko Imai. Computational investigations of all-terminal network reliability via bdds. *IEICE transactions on fundamentals of electronics, communications and computer sciences*, 82(5):714–721, 1999.

[Ishihata *et al.*, 2010] Masakazu Ishihata, Yoshitaka Kameya, Taisuke Sato, and Shin-ichi Minato. An em algorithm on bdds with order encoding for logic-based probabilistic models. In *Proceedings of 2nd Asian Conference on Machine Learning*, pages 161–176, 2010.

[Jerrum, 2003] Mark Jerrum. *Counting, sampling and integrating: algorithms and complexity*. Springer Science & Business Media, 2003.

[Kawahara *et al.*, 2017] 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*, 100(9):1773–1784, 2017.

[Knuth, 2009] Donald E Knuth. The art of computer programming: Bitwise tricks & techniques. *Binary Decision Diagrams*, 4, 2009.

[Lugosi, 2017] Gábor Lugosi. Lectures on combinatorial statistics, 2017.

[MacKinnon, 2009] James G MacKinnon. Bootstrap hypothesis testing. *Handbook of Computational Econometrics*, 183:213, 2009.

[Maehara *et al.*, 2017] Takanori Maehara, Hirofumi Suzuki, and Masakazu Ishihata. Exact computation of influence spread by binary decision diagrams. In *Proceedings of the 26th International Conference on World Wide Web*, pages 947–956. International World Wide Web Conferences Steering Committee, 2017.

[Minato, 1993] Shin-ichi Minato. Zero-suppressed bdds for set manipulation in combinatorial problems. In *30th ACM/IEEE Design Automation Conference*, pages 272–277. IEEE, 1993.

[Minato, 2013] Shin-ichi Minato. Techniques of bdd/zdd: brief history and recent activity. *IEICE TRANSACTIONS on Information and Systems*, 96(7):1419–1429, 2013.

[Qian *et al.*, 2014] Jing Qian, Venkatesh Saligrama, and Yuting Chen. Connected sub-graph detection. In *Artificial Intelligence and Statistics*, pages 796–804, 2014.

[Taylor and Tibshirani, 2015] Jonathan Taylor and Robert J Tibshirani. Statistical learning and selective inference. *Proceedings of the National Academy of Sciences*, 112(25):7629–7634, 2015.

[Weerahandi, 2013] Samaradasa Weerahandi. *Exact statistical methods for data analysis*. Springer Science & Business Media, 2013.