Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence (IJCAI-18)

Algorithms for the Nearest Assignment Problem

Sara Rouhani, Tahrima Rahman, Vibhav Gogate
The University of Texas at Dallas
{sara.rouhani, tahrima.rahman, vibhav.gogate}@utdallas.edu

Abstract

We consider the following nearest assignment problem (NAP): given a Bayesian network $\mathcal{B}$ and probability value q, find a configuration ω of variables in $\mathcal{B}$ such that the difference between q and probability of ω is minimized. NAP is much harder than conventional inference problems such as finding the most probable explanation in that it is NP-hard even on independent Bayesian networks (IBNs), which are networks having no edges. We propose a two-way number partitioning encoding of NAP on IBNs and then leverage poly-time approximation algorithms from the number partitioning literature to develop algorithms with guarantees for solving NAP. We extend our basic algorithm from IBNs to arbitrary probabilistic graphical models by leveraging cutset-based conditioning, local search and (Rao-Blackwellised) sampling algorithms. We derive approximation and complexity guarantees for our new algorithms and show experimentally that they are quite accurate in practice.

1 Introduction

We focus on the following problem in probabilistic graphical models (PGMs): given a real number q such that $0 \leq q \leq 1$ and a PGM $\mathcal{B}$, find a configuration ω of variables in $\mathcal{B}$ such that $\left|\log\left(\frac{q}{P_{\mathcal{B}}(\omega)}\right)\right|$ is minimized where $P_{\mathcal{B}}(\omega)$ is the probability of ω according to $\mathcal{B}$. We call this problem the nearest assignment problem or NAP in short. NAP includes several well-known tasks such as finding the most likely configuration, also called the most probable explanation (MPE) or maximum-a-posteriori (MAP) estimate, as a special case (when $q = 1$) and is thus intractable or NP-hard in general.

A solution to NAP is desired in several real-world applications. For example, it is possible to construct a piece-wise approximation of the CDF of a PGM, possibly conditioned on evidence, by answering a few NAP queries (e.g., after finding the min and max assignments, we can recursively set q to midway between the probability of two known assignments). This approximation is useful in interactive applications [Kulesza *et al.*, 2015], in particular, for explaining the shape of the distribution to the user. CDFs can also be used to check whether two probability distributions are similar subject to a known renaming of their variables, a key sub-task in learning lifted probabilistic models such as Markov logic [Richardson and Domingos, 2006]. Other applications of NAP include (i) finding nearest neighbors when the distance distribution is compactly stored as a PGM for very large datasets; (ii) hardware test case generation [Fournier *et al.*, 1999; Dechter *et al.*, 2002]; and (iii) generating explanations that are close to a given explanation in terms of probability (cf. the DARPA Explainable AI program).

NAP is closely related to the order statistics problem [Smith *et al.*, 2017] and similar to the latter is NP-hard even on so-called *independent PGMs* – PGMs having only univariate potentials/CPTs. Therefore, in order to facilitate the development of non-trivial algorithms for NAP, we show how to encode NAP on independent PGMs as a two-way number partitioning problem. This encoding enables us to leverage well-researched algorithms from the number partitioning literature and develop a greedy algorithm with guarantees on approximation error for solving NAP on independent PGMs. We call it Algorithm INAP.

We then use the 0-cutset [Bidyuk and Dechter, 2004] concept, which is defined as a subset of variables such that when the subset is removed, the remaining PGM has no edges (and thus has zero treewidth) to yield an algorithm with guarantees on complexity and accuracy for arbitrary, high treewidth PGMs. Similar to cutset conditioning [Dechter, 1990], the key idea in this algorithm is to enumerate all configurations to the 0-cutset variables and then solve the sub-problem on each independent PGM using INAP. To make the technique practical, we propose generating configurations of the 0-cutset variables via random sampling and local search. This yields a stochastic anytime version of our 0-cutset algorithm, similar to previously proposed cutset sampling and local search algorithms [Bidyuk and Dechter, 2007; Kask and Dechter, 1996].

We empirically evaluate 0-cutset based sampling and local search algorithms on various benchmark networks from the 2014 UAI competition [Gogate, 2014], comparing their performance with conventional local search and random sampling algorithms (that do not exploit cutsets). Our experiments show that our new algorithms converge faster and are often orders of magnitude better in terms of approximation error than random sampling and local search, especially when q is closer to either the min or the max probability in the model.

2 Preliminaries and Notation

We focus on Bayesian networks [Pearl, 1988] having binary variables noting that results presented in this paper can be easily extended to Markov networks and multi-valued variables. Let $\mathbf{X} = \{X_1, \ldots, X_n\}$ denote a set of n Boolean random variables, namely each variable X_i takes values from the set $\{0, 1\}$. We denote the variable assignments $X_i = 1$ and $X_i = 0$ by x_i and $\overline{x_i}$ respectively.

A discrete Bayesian network $\mathcal{B}$ is a triple $\langle \mathbf{X}, \mathbf{P}, G \rangle$ where $\mathbf{P}$ is a set of conditional probability tables (CPTs) and G is a directed acyclic graph which has one node for each variable. Edges in G define scopes of CPTs. In particular, each CPT $P_i \in \mathbf{P} = \{P_1, \ldots, P_n\}$ has the form $P_i(X_i|pa(X_i))$ where $X_i \in \mathbf{X}$ and $pa(X_i)$ is the set of parents of X_i in G. Given a CPT P_i, we will use the notation $S(P_i)$ to denote the scope of P_i, namely $S(P_i) = \{X_i\} \cup pa(X_i)$. $\mathcal{B}$ represents the probability distribution: $P_{\mathcal{B}}(\omega) = \prod_{i=1}^{n} P_i(\omega_{S(P_i)})$ where ω is an assignment of values to all variables in $\mathbf{X}$ and $\omega_{S(P_i)}$ denotes the projection of ω on the scope of P_i. We will use $\overline{\omega}$ to denote the assignment in which all variable assignments in ω are flipped. For example, if $\omega = (x_1, \overline{x_2}, x_3)$ then $\overline{\omega} = (\overline{x_1}, x_2, \overline{x_3})$.

A *primal graph* or a moral graph of a Bayesian network $\mathcal{B}$ is an undirected graph $D \equiv (\mathbf{V}, \mathbf{E})$ which has one node $V_i \in \mathbf{V}$ for each variable X_i in $\mathcal{B}$ and an edge between any two variables that are in the scope of a CPT, namely $\mathbf{E} = \{(V_i, V_j) | \exists P_k \text{ such that } \{V_i, V_j\} \subseteq S(P_k)\}$.

An independent Bayesian Network (IBN) is a Bayesian network having no edges. Thus, in an IBN the scope of each CPT P_i equals $\{X_i\}$ and its primal graph is empty. A 0-cutset of an undirected graph is a subset of nodes $\mathbf{C}$ such that when all nodes in $\mathbf{C}$ are removed, the remaining graph is empty. This paper addresses the following inference task

Definition 1 (Nearest Assignment Problem (NAP)). *Given a Bayesian network $\mathcal{B}$ and a real number $0 \leq q \leq 1$, find an assignment ω of values to all variables of $\mathcal{B}$ such that $\left|\log\left(\frac{q}{P_{\mathcal{B}}(\omega)}\right)\right|$ is minimized.*

2.1 2-way Number Partitioning

Given a multi-set $\mathbf{S}$ of positive real numbers, the 2-way number partitioning problem is defined as finding a partition $\mathbf{S}_1$, $\mathbf{S}_2$ of $\mathbf{S}$, namely $\mathbf{S}_1 \cup \mathbf{S}_2 = \mathbf{S}$ and $\mathbf{S}_1 \cap \mathbf{S}_2 = \emptyset$, such that $|\sum_{\alpha \in \mathbf{S}_1} \alpha - \sum_{\beta \in \mathbf{S}_2} \beta|$ is minimized.

Popular approaches for solving number partitioning include the greedy method, Karmarkar-Karp heuristic [Karmarkar and Karp, 1983] and branch and bound algorithms [Korf, 1998]. In this paper, we use the Karmarkar-Karp heuristic because of its associated guarantees to yield an algorithm for solving NAP, noting that sophisticated algorithms from the number partitioning literature can be easily employed to further improve our proposed method.

3 NAP on IBNs

In this section, we develop an algorithm with guarantees for solving NAP over IBNs. As mentioned earlier, NAP is closely related to the order-statistics problem [Smith *et al.*, 2017]. In particular, Smith et al. showed that finding an assignment having the median rank in IBNs is NP-hard via a reduction from the 2-way number partitioning problem. Setting $q = P_{\mathcal{B}}(\omega_m)$ where ω_m is the assignment having the median rank in $\mathcal{B}$ yields the following proposition:

Proposition 1. *NAP on IBNs is NP-hard in general.*

A key use of this NP-hardness reduction is that it enables us to encode NAP on IBNs as a number partitioning problem and thus leverage algorithms developed in the literature on number partitioning [Vazirani, 2001; Korf, 1998; Karmarkar and Karp, 1983]. We describe a possible encoding next. Without loss of generality, for each variable X_i, let $P_i(x_i) \geq P_i(\overline{x_i})$ and $q^2 \geq \mu$ where $\mu = \prod_i P_i(x_i) P_i(\overline{x_i})$. We claim that an assignment ω that minimizes $|\log(q) - \log(P_{\mathcal{B}}(\omega))|$ can be recovered by solving the following 2-way number partitioning problem:

Definition 2 (2-way Number Partitioning Encoding). *Given an independent Bayesian network $\mathcal{B} \equiv \langle \boldsymbol{X}, \boldsymbol{P}, G \rangle$, we construct a set $\boldsymbol{S} = \left\{\log\left(\frac{P_i(x_i)}{P_i(\overline{x_i})}\right) | P_i \in \boldsymbol{P}\right\} \cup \left\{\log\left(\frac{q^2}{\mu}\right)\right\}$. The 2-way number partitioning problem given $\boldsymbol{S}$ is to find a 2-way partition $\boldsymbol{S}_1$ and $\boldsymbol{S}_2$ of $\boldsymbol{S}$ such that $|\sum_{\alpha \in \boldsymbol{S}_1} \alpha - \sum_{\beta \in \boldsymbol{S}_2} \beta|$ is minimized.*

Theorem 1. *Finding an assignment ω that minimizes $\left|\log\left(\frac{q}{P_{\mathcal{B}}(\omega)}\right)\right|$ is equivalent to solving the 2-way number partitioning problem described in Definition 2.*

Proof. (Sketch)

$$\begin{aligned} & \arg\min_{\omega} \left|\log\left(\frac{q}{P_{\mathcal{B}}(\omega)}\right)\right| \\ = \; & \arg\min_{\omega} \left|\log\left(\frac{q^2}{P^2_{\mathcal{B}}(\omega)}\right)\right| \qquad (1) \\ = \; & \arg\min_{\omega} \left|\log\left(\frac{q^2}{P^2_{\mathcal{B}}(\omega)} \frac{P_{\mathcal{B}}(\overline{\omega})}{P_{\mathcal{B}}(\overline{\omega})}\right)\right| \qquad (2) \end{aligned}$$

Substituting $\mu = P_{\mathcal{B}}(\omega) P_{\mathcal{B}}(\overline{\omega})$ in Eq. (2), we get

$$= \arg\min_{\omega} \left|\log\left(\frac{q^2}{\mu} \frac{P_{\mathcal{B}}(\overline{\omega})}{P_{\mathcal{B}}(\omega)}\right)\right| \qquad (3)$$

We will simplify $\log\left(\frac{P_{\mathcal{B}}(\overline{\omega})}{P_{\mathcal{B}}(\omega)}\right)$ using the following notation. Let $\mathbf{X}_1$ and $\mathbf{X}_2$ denote the subsets of variables which are assigned to 1 and 0 respectively in ω.

$$\begin{aligned} & \log\left(\frac{P_{\mathcal{B}}(\overline{\omega})}{P_{\mathcal{B}}(\omega)}\right) \\ = \; & \log\left(\frac{\prod_{X_i \in \mathbf{X}_2} P_i(x_i) \prod_{X_i \in \mathbf{X}_1} P_i(\overline{x_i})}{\prod_{X_i \in \mathbf{X}_1} P_i(x_i) \prod_{X_i \in \mathbf{X}_2} P_i(\overline{x_i})}\right) \qquad (4) \\ = \; & \sum_{X_i \in \mathbf{X}_2} \log\left(\frac{P_i(x_i)}{P_i(\overline{x_i})}\right) - \sum_{X_i \in \mathbf{X}_1} \log\left(\frac{P_i(x_i)}{P_i(\overline{x_i})}\right) \qquad (5) \end{aligned}$$

Without loss of generality, let

$$\mathbf{S}_1 = \log\left(\frac{q^2}{\mu}\right) \cup \left\{\log\left(\frac{P_i(x_i)}{P_i(\overline{x_i})}\right) | X_i \in \mathbf{X}_2\right\} \qquad (6)$$

Algorithm 1 INAP: Greedy Algorithm for NAP over IBNs

Input: An Independent Bayesian network $\mathcal{B} \equiv \langle \mathbf{X}, \mathbf{P}, G \rangle$ and a probability q
Output: Nearest assignment ω to q
Begin:

1: $\mathbf{S} = \emptyset$; $\mu = 1$
2: **for** each variable X_i in $\mathbf{X}$ **do**
3: $\quad \mu = \mu \times P_i(x_i)P(\overline{x_i})$
4: $\quad \mathbf{S} = \mathbf{S} \bigcup \left\{ \log\left(\frac{P_i(x_i)}{P_i(\overline{x_i})}\right) \right\}$
5: $\mathbf{S} = \mathbf{S} \cup \left\{ \log\left(\frac{q^2}{\mu}\right) \right\}$
6: Find a partition $\mathbf{S}_1$ and $\mathbf{S}_2$ of $\mathbf{S}$ using either the greedy method or Karmarkar-Karp heuristic [Karmarkar and Karp, 1983]
7: Construct ω from $\mathbf{S}_1$ and $\mathbf{S}_2$ using Corollary 1
8: **return** ω

End.

$$\mathbf{S}_2 = \left\{ \log\left(\frac{P_i(x_i)}{P_i(\overline{x_i})}\right) | X_i \in \mathbf{X}_1 \right\} \tag{7}$$

Combining Equations (3), (5), (6) and (7), we get:

$$\arg\min_{\omega} \left| \log\left(\frac{q}{P_{\mathcal{B}}(\omega)}\right) \right| = \arg\min_{\mathbf{S}_1, \mathbf{S}_2} \left| \sum_{\alpha \in \mathbf{S}_1} \alpha - \sum_{\beta \in \mathbf{S}_2} \beta \right|$$

□

Given a solution to the number partitioning problem described in Definition 2, we can use Theorem 1 to construct the nearest assignment as follows.

Corollary 1. *Without loss of generality, let* $1 \leq k \leq n$ *be an integer such that* $\mathbf{S}_1 = \left\{ \log\left(\frac{P_i(x_i)}{P_i(\overline{x_i})}\right) | i = 1, \ldots, k \right\} \cup \left\{ \log\left(\frac{q^2}{\mu}\right) \right\}$ *and* $\mathbf{S}_2 = \left\{ \log\left(\frac{P_j(x_j)}{P_j(\overline{x_j})}\right) | j = k+1, \ldots, n \right\}$. *Then, the assignment* ω *that minimizes* $\left| \log\left(\frac{q}{P_{\mathcal{B}}(\omega)}\right) \right|$ *is* $(\overline{x_1}, \ldots, \overline{x_k}, x_{k+1}, \ldots, x_n)$.

3.1 Poly-time Approximation Algorithm for Solving NAP over IBNs

Theorem 1, Corollary 1 and the Karmarkar-Karp heuristic yields Algorithm INAP (see Algorithm 1). The algorithm first constructs the number partitioning problem using Definition 2 (lines 1-5). Then, it uses number partitioning algorithms to construct the two subsets $\mathbf{S}_1$ and $\mathbf{S}_2$ (line 6). Finally, it uses Corollary 1 to compute the nearest assignment from $\mathbf{S}_1$ and $\mathbf{S}_2$ (line 7). Note that Algorithm INAP assumes that for each variable X_i, $P_i(x_i) \geq P_i(\overline{x_i})$ and $q^2 \geq \mu$. The two assumptions are required because the Karmarkar-Karp heuristic has guarantees on the approximation error only when all numbers are positive. When the first assumption is violated we use $\log\left(\frac{P_i(\overline{x_i})}{P_i(x_i)}\right)$ instead of $\log\left(\frac{P_i(x_i)}{P_i(\overline{x_i})}\right)$ and then flip the assignment to the corresponding variable (namely replace x_i by $\overline{x_i}$ and vice versa) while constructing ω from $\mathbf{S}_1$ and $\mathbf{S}_2$. When the second assumption is violated, we use $\log\left(\frac{\mu}{q^2}\right)$ instead of $\log\left(\frac{q^2}{\mu}\right)$ and return $\overline{\omega}$ instead of ω.

Since the Karmarkar-Karp algorithm outputs a $7/6$ approximation [Matteo and Silvano, 1987], we can prove that Algorithm INAP has the following guarantees:

Theorem 2. *Let* ω^* *be the (optimal) nearest assignment to q,* ω *be the assignment returned by Algorithm* INAP *and* $T = \sum_{\alpha \in \mathbf{S}} \alpha$. *Then,*

$$\left| \log\left(\frac{P_{\mathcal{B}}(\omega^*)}{P_{\mathcal{B}}(\omega)}\right) \right| \leq \frac{1}{12}T$$

Proof. (Sketch) Without loss of generality, let $q = P_{\mathcal{B}}(\omega^*)$ and $\sum_{\alpha \in \mathbf{S}_1} \alpha > \sum_{\beta \in \mathbf{S}_2} \beta$. Since the greedy algorithm and Karmarkar-Karp method yield a $7/6$ approximation, we have the following guarantees on the sets $\mathbf{S}_1$ and $\mathbf{S}_2$ computed in Step 6 of Algorithm INAP:

$$\sum_{\alpha \in \mathbf{S}_1} \alpha \leq \frac{7}{6}\frac{T}{2} = \frac{7}{12}T \tag{8}$$

$$\sum_{\beta \in \mathbf{S}_2} \beta \geq \frac{5}{6}\frac{T}{2} = \frac{5}{12}T \tag{9}$$

From Eqs. (8) and (9), we get:

$$\sum_{\alpha \in \mathbf{S}_1} \alpha - \sum_{\beta \in \mathbf{S}_2} \beta \leq \frac{7}{12}T - \frac{5}{12}T = \frac{T}{6} \tag{10}$$

We consider two cases.

- Case 1: $\log\left(\frac{q^2}{\mu}\right)$ is in set $\mathbf{S}_1$
- Case 2: $\log\left(\frac{q^2}{\mu}\right)$ is in set $\mathbf{S}_2$.

Let $\mathbf{X}_1$ and $\mathbf{X}_2$ be subsets of variables such that $\forall X_i \in \mathbf{X}_1$, $\log\left(\frac{P_i(x_i)}{P_i(\overline{x_i})}\right)$ is in $\mathbf{S}_1$ and $\forall X_j \in \mathbf{X}_2$, $\log\left(\frac{P_j(x_j)}{P_j(\overline{x_j})}\right)$ is in $\mathbf{S}_2$.

For Case 1, we can rewrite Eq. (10) as

$$\sum_{X_i \in \mathbf{X}_1} \log\left(\frac{P_i(x_i)}{P_i(\overline{x_i})}\right) + \log\left(\frac{q^2}{\mu}\right) - \sum_{X_j \in \mathbf{X}_2} \log\left(\frac{P_j(x_j)}{P_j(\overline{x_j})}\right) \leq \frac{T}{6} \tag{11}$$

Rearranging Eq. (11) and substituting $q = P_{\mathcal{B}}(\omega^*)$, $P_{\mathcal{B}}(\omega) = \prod_{X_i \in \mathbf{X}_1} P_i(\overline{x_i}) \prod_{X_j \in \mathbf{X}_2} P_j(x_j)$, and $\mu = \prod_{X_i \in \mathbf{X}} P_i(x_i)P_i(\overline{x_i})$, we get

$$\log\left(\frac{P_{\mathcal{B}}(\omega^*)}{P_{\mathcal{B}}(\omega)}\right) \leq \frac{1}{12}T \tag{12}$$

For Case 2, performing similar analysis as above, we get:

$$\log\left(\frac{P_{\mathcal{B}}(\omega)}{P_{\mathcal{B}}(\omega^*)}\right) \leq \frac{1}{12}T \tag{13}$$

Combining Eqs. (12) and (13), we get the following result:

$$\left| \log\left(\frac{P_{\mathcal{B}}(\omega^*)}{P_{\mathcal{B}}(\omega)}\right) \right| \leq \frac{1}{12}T$$

□

From Theorem 2, we can see that Algorithm INAP is accurate when T is small. In turn, T is small when the following two conditions are satisfied: (1) the difference between q and $\sqrt{\mu}$ is small, namely q is close to the probability of the median assignment in the network, and (2) the log odds, namely $\log\left(\frac{P_i(x_i)}{P_i(\overline{x_i})}\right)$ of all variables are small.

4 NAP on Arbitrary Bayesian networks

Algorithm 2 0CNAP 0-cutset based algorithm for solving NAP on Arbitrary Bayesian networks

Input: Bayesian network $\mathcal{B} \equiv \langle \mathbf{X}, \mathbf{P}, G \rangle$ and probability q
Output: Nearest assignment ω to q
Begin:

1: ω_{best} = a random assignment
2: **C** = A 0-cutset of $\mathcal{B}$
3: **for** each assignment $\omega_{\mathbf{C}}$ to **C** **do**
4: $\quad \mathbf{Y} = \mathbf{X} \setminus \mathbf{C}$
5: $\quad \mathbf{U} = \{P_i(\omega_{S(P_i)}) | S(P_i) \not\subset \mathbf{C}\}$
6: $\quad \mathcal{B}_c \equiv \langle \mathbf{Y}, \mathbf{U}, G_c \equiv (\mathbf{Y}, \emptyset) \rangle$
7: $\quad q_c = \frac{q}{\prod_{i:S(P_i)\subseteq \mathbf{C}} P_i(\omega_{S(P_i)})}$
8: $\quad \omega_{\mathbf{Y}} = \text{INAP}(\mathcal{B}_c, q_c)$
9: $\quad \omega = (\omega_{\mathbf{Y}}, \omega_{\mathbf{C}})$
10: $\quad$ **if** $|\log(\frac{P_{\mathcal{B}}(\omega)}{q})| < |\log(\frac{P_{\mathcal{B}}(\omega_{best})}{q})|$ **then**
11: $\quad\quad \omega_{best} = \omega$
12: **return** ω_{best}

End.

In this section, we extend Algorithm INAP from independent Bayesian networks to arbitrary Bayesian networks, yielding an algorithm with guarantees on approximation error as well as time and (linear) space complexity.

The key idea in our proposed method is the following: (1) find a 0-cutset, namely remove a subset of variables from the primal graph of the Bayesian network until no edges remain, and (2) for each possible assignment to the 0-cutset variables, solve the sub-problem defined over an independent Bayesian network using Algorithm INAP and (3) return the assignment whose probability is closest to q over all sub-problems. Our proposed method is described in Algorithm 0CNAP (see Algorithm 2). The algorithm takes as input a Bayesian network $\mathcal{B}$ and a probability q and returns the nearest assignment to q in $\mathcal{B}$ (line 12). It begins by finding a zero cutset (line 2). Then, for all configurations $\omega_{\mathbf{C}}$ to the 0-cutset variables, it executes the following steps. First, it creates an independent Bayesian network $\mathcal{B}_c$ given $\omega_{\mathbf{C}}$ (lines 4-6). The variables of $\mathcal{B}_c$ are all variables in $\mathcal{B}$ minus the variables included in the 0-cutset. Note that once the 0-cutset variables are assigned a value, each CPT will be in one the two following states: (1) all variables of the CPT are assigned a value (fully instantiated CPTs); and (2) exactly one variable of the CPT is uninstantiated. The latter CPTs form the CPTs of $\mathcal{B}_c$ (line 5). Second, it computes q_c, the remaining value of q that is relevant to the sub-problem over $\mathcal{B}_c$. q_c is obtained by dividing q by the product of CPTs that are fully instantiated (line 7). Third, it runs Algorithm INAP on the sub-problem defined on $\mathcal{B}_c$ and q_c (line 8) and constructs a full assignment over all variables of $\mathcal{B}$ by concatenating the assignment returned by Algorithm INAP and $\omega_{\mathbf{C}}$ (line 9). Finally, it updates the best assignment if the full assignment defined in line 9 is better than the best assignment so far (lines 10-11).

4.1 Theoretical Guarantees

We can use Theorem 2 to yield the following guarantees on Algorithm 0CNAP (see Theorem 3). The key idea is that only bounds for the independent sub-problems are relevant and thus 0-cutset allows us to derive stronger guarantees. More formally, in Algorithm 0CNAP, let $\mathbf{J} = \{1, \ldots, 2^{|\mathbf{C}|}\}$ index the iterations of the for loop of Algorithm 0CNAP. Also, let $\omega^{(k)} = (\omega_{\mathbf{C}}^{(k)}, \omega_{\mathbf{Y}}^{(k)})$ denote the assignment generated in the k-th iteration (line 9) and let $T^{(k)}$ denote the sum of elements of set **S** constructed by Algorithm INAP in the k-iteration (line 8). We can prove that

Theorem 3. *Let ω^* be the (optimal) nearest assignment to q, and ω be the assignment returned by Algorithm* 0CNAP. *Then,*

$$\left|\log\left(\frac{P_{\mathcal{B}}(\omega^*)}{P_{\mathcal{B}}(\omega)}\right)\right| \leq \frac{1}{12}\max_{k\in\mathbf{J}} T^{(k)}$$

Proof. (Sketch) Let i be the index such that $\omega_{\mathbf{C}}^{(i)} = \omega_{\mathbf{C}}^*$. Using Theorem 2, we have:

$$\left|\log\left(\frac{P_{\mathcal{B}}(\omega_{\mathbf{Y}}^*|\omega_{\mathbf{C}}^*)}{P_{\mathcal{B}}(\omega_{\mathbf{Y}}^{(i)}|\omega_{\mathbf{C}}^{(i)})}\right)\right| \leq \frac{1}{12}T^{(i)} \tag{14}$$

Left hand side of Eq. (14) can be simplified as follows:

$$\left|\log\left(\frac{P_{\mathcal{B}}(\omega_{\mathbf{Y}}^*|\omega_{\mathbf{C}}^*)}{P_{\mathcal{B}}(\omega_{\mathbf{Y}}^{(i)}|\omega_{\mathbf{C}}^{(i)})}\right)\right| \tag{15}$$

$$= \left|\log\left(\frac{P_{\mathcal{B}}(\omega_{\mathbf{Y}}^*|\omega_{\mathbf{C}}^*)P_{\mathcal{B}}(\omega_{\mathbf{C}}^*)}{P_{\mathcal{B}}(\omega_{\mathbf{Y}}^{(i)}|\omega_{\mathbf{C}}^{(i)})P_{\mathcal{B}}(\omega_{\mathbf{C}}^*)}\right)\right| \tag{16}$$

$$= \left|\log\left(\frac{P_{\mathcal{B}}(\omega^*)}{P_{\mathcal{B}}(\omega^{(i)})}\right)\right| \text{ (since } \omega_{\mathbf{C}}^{(i)} = \omega_{\mathbf{C}}^*) \tag{17}$$

Substituting Eq. (17) in Eq. (14), we get:

$$\left|\log\left(\frac{P_{\mathcal{B}}(\omega^*)}{P_{\mathcal{B}}(\omega^{(i)})}\right)\right| \leq \frac{1}{12}T^{(i)} \tag{18}$$

However, since ω is the best assignment, we have

$$\left|\log\left(\frac{P_{\mathcal{B}}(\omega^*)}{P_{\mathcal{B}}(\omega)}\right)\right| \leq \left|\log\left(\frac{P_{\mathcal{B}}(\omega^*)}{P_{\mathcal{B}}(\omega^{(i)})}\right)\right| \tag{19}$$

Combining Eqs. (18) and (19), we get:

$$\left|\log\left(\frac{P_{\mathcal{B}}(\omega^*)}{P_{\mathcal{B}}(\omega)}\right)\right| \leq \frac{1}{12}T^{(i)} \leq \frac{1}{12}\max_{k\in\mathbf{J}} T^{(k)} \tag{20}$$

□

Notice that since the error $\max_{k\in\mathbf{J}} T^{(k)}$ of Algorithm 0CNAP is bounded by the sum over log odds of a subset of variables plus a constant term, it is superior to INAP, whose error is bounded by the sum over log odds of *all* variables.

Moreover Theorem 3 suggests that in order to improve the accuracy (bounds), it is beneficial to select 0-cutset variables in such a way that the remaining variables have small log odds. In other words, Theorem 3 helps provide heuristic guidance for selecting good 0-cutsets.

4.2 Practical Considerations

A key issue with Algorithm OCNAP is that it has exponential computational complexity. Formally,

Proposition 2. *The time complexity of Algorithm* OCNAP *is* $O(2^c r \log(r))$ *where* c *is the size of the* 0*-cutset* $\boldsymbol{C}$ *and* r *is the number of variables in the set* $\boldsymbol{X} \setminus \boldsymbol{C}$.

To make the algorithm practical, we propose to either sample or perform local search over the assignments to the 0-cutset variables. Our proposed method belongs to the class of cutset sampling algorithms [Bidyuk and Dechter, 2007] which achieve variance reduction using the Rao-Blackwell Theorem. The key idea is that combining methods that have guarantees with sampling improves the quality of estimates output by the latter.

5 Experiments

In this section, we describe results of our extensive empirical investigation on a large number of benchmark problems from the UAI 2014 probabilistic inference competition. Our hypothesis is that given a time bound, cutset based local search and random sampling algorithms are more accurate than their conventional counterparts (that do not exploit cutsets).

Algorithms. We implemented four algorithms: (a) **Random Sampling (RS):** We generate assignments ω uniformly at random and output the one having the smallest $\left|\log\left(\frac{q}{P_{\mathcal{B}}(\omega)}\right)\right|$. (b) **0-cutset Random Sampling (CRS)**: Given a cutset $\mathbf{C}$, we generate assignments $\omega_{\mathbf{C}}$ to the cutset variables uniformly at random and then use the Karmarkar-Karp heuristic to compute the assignment $\omega_{\mathbf{Y}}$ to the remaining variables given $\omega_{\mathbf{C}}$. As before, we choose the assignment $\omega = (\omega_{\mathbf{C}}, \omega_{\mathbf{Y}})$ having the smallest $\left|\log\left(\frac{q}{P_{\mathcal{B}}(\omega)}\right)\right|$. (c) **Local search (LS)**: Starting with a random assignment ω, we evaluate all neighbors of ω (set of assignments in which exactly one variable in ω is flipped) and choose the neighbor ω' having the smallest $\left|\log\left(\frac{q}{P_{\mathcal{B}}(\omega')}\right)\right|$. We randomly restart the search if the local minima is reached or 10000 moves have been made, which ever is earlier. (d) **Cutset Local search (CLS)**: We perform local search over the cutset variables only, namely we evaluate only those assignments in which one of the cutset variables is flipped and choose the neighbor $\omega'_{\mathbf{C}}$ having the smallest $\left|\log\left(\frac{q}{P_{\mathcal{B}}(\omega'_{\mathbf{C}}, \omega'_{\mathbf{Y}})}\right)\right|$ where $\omega'_{\mathbf{Y}}$ is computed using the Karmarkar-Karp heuristic given $\omega'_{\mathbf{C}}$. As before, we randomly restart the search if the local minima is reached or 1000 moves have been made which ever is earlier. It is easy to see that each move of cutset local search (cutset random sampling) has higher computationally complexity than local search (random sampling). However, we expect the cutset algorithms to be more accurate because they explore the search space more efficiently.

Benchmark Networks. We compared our algorithms on the following four types of benchmark Bayesian and Markov networks: (1) Noisy OR Bayesian networks (BN20), (2) Ising Models, (3) Relational Markov Networks, and (4) Image Segmentation. These networks were used in the UAI 2014 and 2016 inference competitions (see [Gogate, 2014]).

Methodology. We randomly generated 20 q values for each network as follows. We generated 10 million random samples, sorted them according to their probability (or weight), and chose every 500 thousandth sample. We evaluated the performance using the following error measure. Let ω be the assignment output by the sampling algorithm. Then, for Bayesian networks, we use the quantity $Error = \left|\log\left(\frac{q}{P_{\mathcal{B}}(\omega)}\right)\right|$ to measure performance. For Markov networks or log-linear models, we use $Error = \left|\log\left(\frac{q}{\exp(\sum_i f_i(\omega)\theta_i)}\right)\right|$ where f_i is a feature, $\theta_i \in \mathbb{R}$ is its weight and $f_i(\omega)$ is 1 if f_i is true in ω and 0 otherwise. We constructed the 0-cutset as follows. We recursively delete a variable having the highest degree until the graph is empty, breaking ties using average log odds (see Theorem 3).

5.1 Results

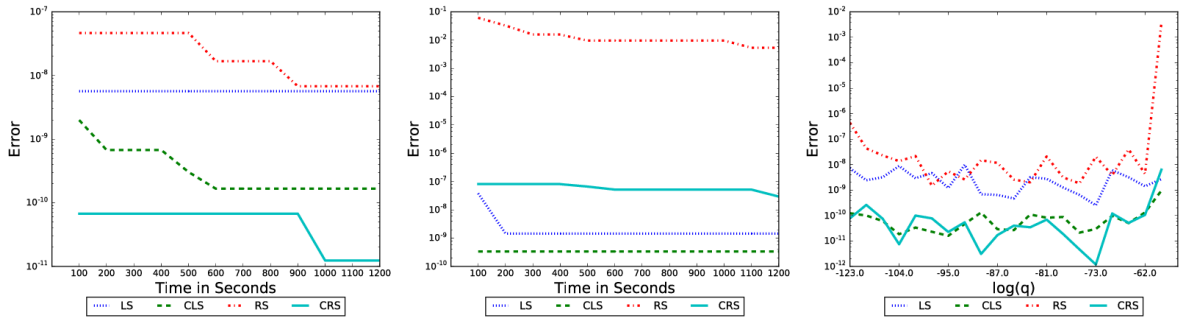
We compare the impact of changing q as well as increasing time on the performance of various algorithms. We ran each algorithm for 20 minutes. Figures 1-4 show the results. For each benchmark type, due to lack of space, we report results for a randomly chosen network. For each chosen network, we report three plots showing the impact of varying the time bound for a given q and of varying q for a given time bound. The first two plots show the $Error$ as a function of time for two q values, one near the median and one close to the MAP assignment. The third plot shows $Error$ as a function q after 1200 seconds, the time bound. In general, we found that cutset based algorithms are superior to their conventional counterparts in resource-limited settings.

BN20 networks

BN20 networks [Henrion *et al.*, 1992] are two level Bayesian networks in which parent interactions are modeled using the noisy-or interaction model. These networks are used for medical diagnosis in which the top level (parents) has Boolean variables which model the diseases and the bottom level has variables modeling the symptoms. Exact inference using conventional algorithms such as variable elimination and cutset conditioning is known to be intractable in these networks. Figure 1 shows the results for BN20 networks. We find that CRS and CLS are the best performing algorithms and dominate both LS and RS. For values of q close to the probability of the MAP assignment, LS is slightly better than CRS. However, in such cases, CLS is better than LS by an order of magnitude (notice the log-scale on the Y-axis). Finally, we find that unlike other algorithms, RS tends to have larger error at the extremes than the median values (see Fig. 1(c)).

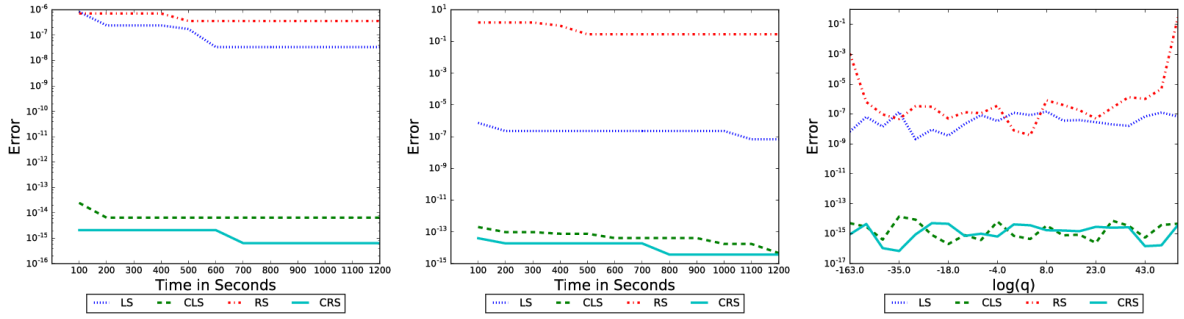
Ising Models

Fig. 2 shows performance of various algorithms on an Ising model having 400 variables. We see that CLS and CRS are the best performing schemes and dominate LS and RS by several orders of magnitude. LS outperforms random sampling.



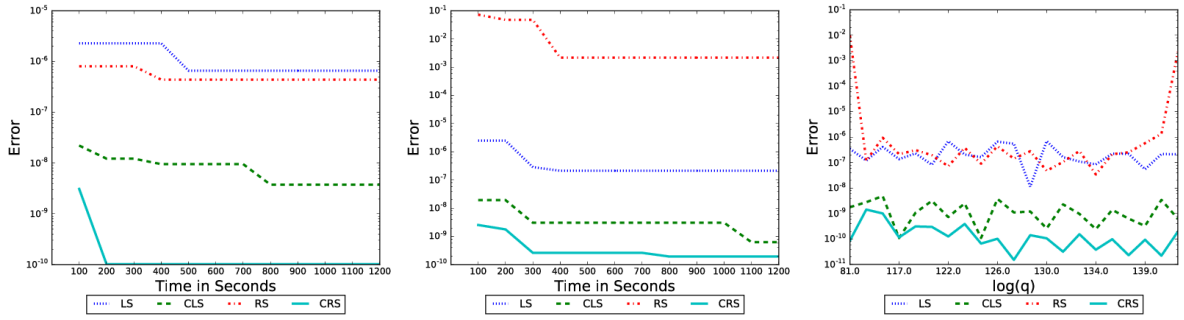
(a) $\log(q)$ =-90.18 (b) $\log(q)$ =-27.94 (c) Time = 1200 seconds

Figure 1: Results on bn2o-30-25-250-3b Bayesian network. The network has 55 variables.



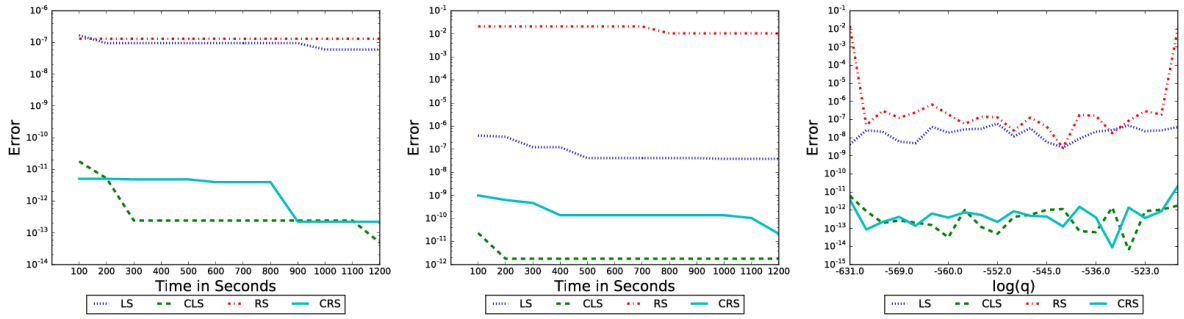
(a) $\log(q)$ =-4.17 (b) $\log(q)$ =173.36 (c) Time = 1200 seconds

Figure 2: Results on Grids_15 Markov network. The network has 400 variables and 1160 potentials.



(a) $\log(q)$ =126.10 (b) $\log(q)$ =170.35 (c) Time = 1200 seconds

Figure 3: Results on Relational3 Markov network. The network has 1000 variables and 2000 potentials.



(a) $\log(q)$ =-630.86 (b) $\log(q)$ =-441.32 (c) Time = 1200 seconds

Figure 4: Results on Segmentation13 Markov network. The network has 235 variables and 882 potentials.

Relational Markov Networks (RMNs)

Relational Markov networks are obtained by grounding the friends and smokers Markov logic networks [Domingos and Lowd, 2009]. Fig. 3 shows performance of various algorithms on a relational network for two values of q and time bound 1200 seconds. Results on these networks are similar to those obtained over Ising models in that CRS and CLS as the best performing schemes. However, unlike other networks CRS dominates CLS. This is because RMNs have tied parameters and as a result a large number of assignments have the same probability. Thus, there is a higher chance of hitting the correct answer via random sampling.

Image Segmentation

The goal of image segmentation is to extract objects from an image which allows us to separate the image into several meaningful regions. Graphical models in which variables correspond to regions and edges which model relationships between adjacent regions are often used for performing image segmentation. Results for a benchmark image segmentation network are shown in Fig. 4. We see a similar picture to other benchmark networks: CLS and CRS outperform both LS and RS.

6 Conclusion and Future Work

In this paper, we considered the nearest assignment problem (NAP) which is the problem of finding an assignment ω in a graphical model such that the difference between the probability of ω and a given value q is minimized. We introduced novel techniques that combine 0-cutset conditioning, sampling and local search algorithms with algorithms from the number partitioning literature for solving NAP. Theoretically, we showed that our new techniques have bounded approximation error and are likely to be accurate when the distribution is not skewed and q is not extreme. Via a detailed experimental evaluation, we showed that our new techniques are superior to baseline sampling and local search algorithms.

Future work includes: developing branch and bound algorithms for NAP; investigating incremental versions of Karmarkar-Karp and greedy algorithms and combining them with local search techniques; using algorithms for NAP to solve the rank estimation problem; using NAP to generate human understandable explanations; etc.

Acknowledgements

This work was supported in part by the DARPA Explainable Artificial Intelligence (XAI) Program under contract number N66001-17-2-4032, and by the National Science Foundation grants IIS-1652835 and IIS-1528037.

References

[Bidyuk and Dechter, 2004] B. Bidyuk and R. Dechter. On finding minimal w-cutset. In *UAI*, pages 43–50, 2004.

[Bidyuk and Dechter, 2007] B. Bidyuk and R. Dechter. Cutset Sampling for Bayesian Networks. *JAIR*, 28:1–48, 2007.

[Darwiche, 2009] A. Darwiche. *Modeling and Reasoning with Bayesian Networks*. Cambridge University Press, 2009.

[Dechter *et al.*, 2002] R. Dechter, K. Kask, E. Bin, and R. Emek. Generating random solutions for constraint satisfaction problems. In *AAAI*, pages 15–21, 2002.

[Dechter, 1990] R. Dechter. Enhancement schemes for constraint processing: Backjumping, learning, and cutset decomposition. *Artificial Intelligence*, 41(3):273–312, 1990.

[Domingos and Lowd, 2009] P. Domingos and D. Lowd. *Markov Logic: An Interface Layer for Artificial Intelligence*. Morgan & Claypool, San Rafael, CA, 2009.

[Fournier *et al.*, 1999] L. Fournier, Y. Arbetman, and M. Levinger. Functional verification methodology for microprocessors using the genesys test-program generator. In *DATE*, 1999.

[Gogate, 2014] V. Gogate. Results of the 2014 uai competition. http://www.hlt.utdallas.edu/∼vgogate/uai14-competition/index.html, 2014.

[Henrion *et al.*, 1992] M. Henrion, J. Breese, and E. Horvitz. Decision Analysis and Expert Systems. *AI Magazine*, 12:64–91, 1992.

[Karmarkar and Karp, 1983] N. Karmarkar and R. M. Karp. The differencing method of set partitioning. Technical Report UCB/CSD-83-113, EECS Department, University of California, Berkeley, 1983.

[Kask and Dechter, 1996] K. Kask and R. Dechter. A graph-based method for improving GSAT. In *AAAI*, pages 350–355, 1996.

[Korf, 1998] R. E Korf. A complete anytime algorithm for number partitioning. *Artificial Intelligence*, 106(2):181–203, 1998.

[Kulesza *et al.*, 2015] Todd Kulesza, Margaret Burnett, Weng-Keen Wong, and Simone Stumpf. Principles of explanatory debugging to personalize interactive machine learning. In *IUI*, pages 126–137. ACM, 2015.

[Matteo and Silvano, 1987] F. Matteo and M. Silvano. Worst-case analysis of the differencing method for the partition problem. *Mathematical Programming*, 37(1):117–120, Feb 1987.

[Pearl, 1988] J. Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann, San Francisco, CA, 1988.

[Richardson and Domingos, 2006] M. Richardson and P. Domingos. Markov Logic Networks. *Machine Learning*, 62:107–136, 2006.

[Smith *et al.*, 2017] D. B. Smith, S. Rouhani, and V. Gogate. Order statistics for probabilistic graphical models. In *IJCAI*, pages 4625–4631, 2017.

[Vazirani, 2001] V. V. Vazirani. *Approximation Algorithms*. Springer-Verlag New York, Inc., New York, NY, USA, 2001.