An Efficient Algorithm for Inference in Rough Set Flow Graphs

C.J. Butz, W.Yan, B. Yang

Department of Computer Science, University of Regina, Regina, Canada, S4S 0A2,
{butz,yanwe111,boting}@cs.uregina.ca

Abstract. Pawlak recently introduced *rough set flow graphs* (RSFGs) as a graphical framework for reasoning from data. No study, however, has yet investigated the complexity of the accompanying inference algorithm, nor the complexity of inference in RSFGs. In this paper, we show that the traditional RSFG inference algorithm has exponential time complexity. We then propose a new RSFG inference algorithm that exploits the factorization in a RSFG. We prove its correctness and establish its polynomial time complexity. In addition, we show that our inference algorithm never does more work than the traditional algorithm. Our discussion also reveals that, unlike traditional rough set research, RSFGs make implicit independency assumptions regarding the problem domain.

Keywords: Reasoning under uncertainty, rough set flow graphs.

1 Introduction

Very recently, Pawlak [7,8] introduced *rough set flow graphs* (RSFGs) as a graphical framework for uncertainty management. RSFGs extend traditional rough set research [9,10] by organizing the rules obtained from decision tables as a *directed acyclic graph* (DAG). Each rule is associated with three coefficients, namely, *strength*, *certainty* and *coverage*, which have been shown to satisfy Bayes' theorem [7,8]. Pawlak also provided an algorithm to answer queries in a RSFG and stated that RSFGs are a new perspective on Bayesian inference [7]. No study, however, has yet investigated the complexity of Pawlak's inference algorithm, nor the complexity of inference in RSFGs.

In this paper, our analysis of the traditional RSFG inference algorithm [7,8] establishes that its time complexity is exponential with respect to the number of nodes in a RSFG. We then propose a new inference algorithm that exploits the factorization in a RSFG. We prove the correctness of our algorithm and establish its polynomial time complexity. In addition, we show that our algorithm never does more work than the traditional algorithm, where work is the number of additions and multiplications needed to answer a query. The analysis in this manuscript also reveals that RSFGs make implicit assumptions regarding the problem domain. More specifically, we show that the *flow conservation assumption* [7] is in fact a *probabilistic conditional independency* [13] assumption.

It should be noted that the work here is different from our earlier work [2] in several important ways. In this manuscript, we propose a new algorithm for

RSFG inference and establish its polynomial time complexity. On the contrary, we established the polynomial complexity of RSFG inference in [2] by utilizing the relationship between RSFGs and *Bayesian networks* [11]. Another difference is that here we show that RSFG inference algorithm in [7,8] has exponential time complexity, an important result not discussed in [2].

This paper is organized as follows. Section 2 reviews probability theory, RSFGs and a traditional RSFG inference algorithm [7,8]. That the traditional inference algorithm has exponential time complexity is shown in Section 3. In Section 4, we propose a new RSFG inference algorithm. We prove the correctness of this new algorithm and establish its polynomial time complexity in Section 5. Section 6 shows that it never does more work than the traditional algorithm. In Section 7, we observe that RSFGs make independence assumptions. The conclusion is presented in Section 8.

2 Definitions

In this section, we review probability theory and RSFGs.

2.1 Probability Theory

Let $U = \{v_1, v_2, \ldots, v_m\}$ be a finite set of variables. Each variable v_i has a finite domain, denoted $dom(v_i)$, representing the values that v_i can take on. For a subset $X = \{v_i, \ldots, v_j\}$ of U, we write $dom(X)$ for the Cartesian product of the domains of the individual variables in X, namely, $dom(X) = dom(v_i) \times \ldots \times dom(v_j)$. Each element $c \in dom(X)$ is called a *configuration* of X. If c is a configuration on X and $Y \subseteq X$, then by c_Y we denote the configuration on Y by dropping from c the values of those variables not in Y.

A *potential* [12] on $dom(U)$ is a function ϕ on $dom(U)$ such that the following two conditions both hold: (i) $\phi(u) \geq 0$, for each configuration $u \in dom(U)$, and (ii) $\phi(u) > 0$, for at least one configuration $u \in dom(U)$. For brevity, we refer to ϕ as a potential on U rather than $dom(U)$, and we call U, not $dom(U)$, its domain [12]. By XY, we denote $X \cup Y$.

A *joint probability distribution* (jpd) [12] on U is a function p on U such that the following two conditions both hold: (i) $0 \leq \phi(u) \leq 1$, for each configuration $u \in U$, and (ii) $\sum_{u \in U} \phi(u) = 1.0$.

Example 1. Consider five attributes Manufacturer (M), Dealership (D), Age (A), Salary (S), Position (P). One jpd $p(U)$ on $U = \{M, D, A, S, P\}$ is depicted in Appendix I.

We say X and Z are *conditionally independent* [13] given Y, denoted $I(X, Y, Z)$, in a joint distribution $p(X, Y, Z, W)$, if

$$p(X, Y, Z) = \frac{p(X, Y) \cdot p(Y, Z)}{p(Y)},$$

where $p(V)$ denotes the marginal [12] distribution of a jpd $p(U)$ onto $V \subseteq U$ and $p(Y) > 0$.

The following theorem provides a necessary and sufficient condition for determining when a conditional independence holds in a problem domain.

Theorem 1. *[5] $I(X, Y, Z)$ iff there exist potentials ϕ_1 and ϕ_2 such that for each configuration c on XYZ with $p(c_Y) > 0$, $p(c) = \phi_1(c_{XY}) \cdot \phi_2(c_{YZ})$.*

Example 2. Recall the jpd $p(U)$ in Example 1. The marginal $p(M, D, A)$ of $p(U)$ and two potentials $\phi(M, D)$, $\phi(D, A)$ are depicted in Table 1. By definition, conditional independence $I(M, D, A)$ holds in $p(U)$ as $p(M, D, A) = \phi(M, D) \cdot \phi(D, A)$.

Table 1: The marginal $p(M, D, A)$ of $p(U)$ in Example 1 and potentials $\phi(M, D)$ and $\phi(D, A)$.

M	D	A	$p(M,D,A)$
Toyota	Alice	Old	0.036
Toyota	Alice	Middle	0.072
Toyota	Alice	Young	0.012
Toyota	Bob	Old	0.024
Toyota	Bob	Middle	0.036
Toyota	Dave	Old	0.002
Toyota	Dave	Middle	0.006
Toyota	Dave	Young	0.012
Honda	Bob	Old	0.060
Honda	Bob	Middle	0.090
Honda	Carol	Middle	0.090
Honda	Carol	Young	0.060
Ford	Alice	Old	0.015
Ford	Alice	Middle	0.030
Ford	Alice	Young	0.005
Ford	Bob	Old	0.060
Ford	Bob	Middle	0.090
Ford	Carol	Middle	0.030
Ford	Carol	Young	0.020
Ford	Dave	Old	0.025
Ford	Dave	Middle	0.075
Ford	Dave	Young	0.150

M	D	$\phi(M,D)$
Toyota	Alice	0.120
Toyota	Bob	0.060
Toyota	Dave	0.020
Honda	Bob	0.150
Honda	Carol	0.150
Ford	Alice	0.050
Ford	Bob	0.150
Ford	Carol	0.050
Ford	Dave	0.250

D	A	$\phi(D,A)$
Alice	Old	0.300
Alice	Middle	0.600
Alice	Young	0.100
Bob	Old	0.400
Bob	Middle	0.600
Carol	Middle	0.600
Carol	Young	0.400
Dave	Old	0.100
Dave	Middle	0.300
Dave	Young	0.600

2.2 Rough Set Flow Graphs

Rough set flow graphs are built from decision tables. A *decision table* [10] represents a potential $\phi(C, D)$, where C is a set of conditioning attributes and D is a decision attribute.

Example 3. Recall the five attributes $\{M, D, A, S, P\}$ from Example 1. Consider the set $C = \{M\}$ of conditioning attributes and the decision attribute D. Then one decision table $\phi(M, D)$ is shown in Table 2. Similarly, decision tables $\phi(D, A)$, $\phi(A, S)$ and $\phi(S, P)$ are also depicted in Table 2.

Table 2: *Decision tables* $\phi(M, D)$, $\phi(D, A)$, $\phi(A, S)$ and $\phi(S, P)$.

M	D	$\phi(M, D)$
Toyota	Alice	120
Toyota	Bob	60
Toyota	Dave	20
Honda	Bob	150
Honda	Carol	150
Ford	Alice	50
Ford	Bob	150
Ford	Carol	50
Ford	Dave	250

D	A	$\phi(D, A)$
Alice	Old	51
Alice	Middle	102
Alice	Young	17
Bob	Old	144
Bob	Middle	216
Carol	Middle	120
Carol	Young	80
Dave	Old	27
Dave	Middle	81
Dave	Young	162

A	S	$\phi(A, S)$
Old	High	133
Old	Medium	67
Old	Low	22
Middle	High	104
Middle	Medium	311
Middle	Low	104
Young	High	26
Young	Medium	52
Young	Low	181

S	P	$\phi(S, P)$
High	Executive	210
High	Staff	45
High	Manager	8
Medium	Executive	13
Medium	Staff	387
Medium	Manager	30
Low	Executive	3
Low	Staff	12
Low	Manager	292

Each decision table defines a binary RSFG. The set of nodes in the flow graph are $\{c_1, c_2, \ldots, c_k\} \cup \{d_1, d_2, \ldots, d_l\}$, where $c_1, c_2, \ldots, c_k$ and $d_1, d_2, \ldots, d_l$ are the values of C and D appearing in the decision table, respectively. For each row in the decision table, there is a directed edge (c_i, d_j) in the flow graph, where c_i is the value of C and d_j is the value of D. Clearly, the defined graphical structure is a *directed acyclic graph* (DAG). Each edge (c_i, d_j) is labelled with three coefficients. The *strength* of (c_i, d_j) is $\phi(c_i, d_j)$ obtained from the decision table. From $\phi(c_i, d_j)$, we can compute the *certainty* $\phi(d_j|c_i)$ and the *coverage* $\phi(c_i|d_j)$.

Example 4. Consider the decision tables $\phi(M, D)$ and $\phi(D, A)$ in Table 2. The DAGs of the binary RSFGs are illustrated in Fig. 1, respectively. The strength, certainty and coverage of the edges of the flow graphs in Fig. 1 are shown in the top two tables of Table 3.

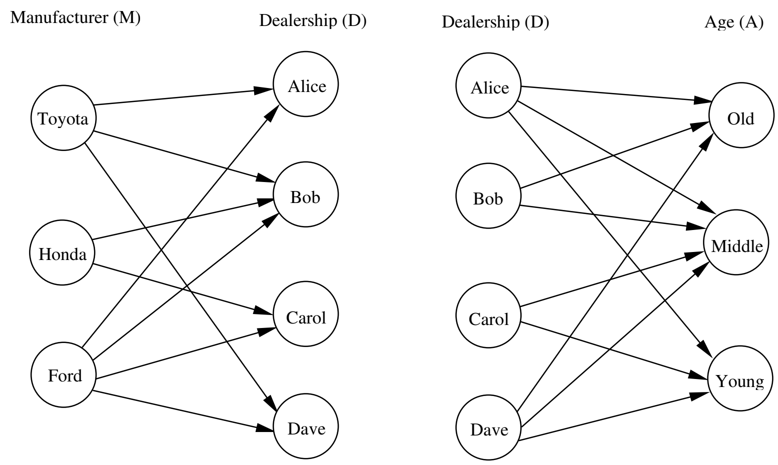

Fig. 1: The DAGs of the binary RSFGs for the decision tables $\phi(M, D)$ and $\phi(D, A)$ in Table 2, respectively. The coefficients are given in part of Table 3.

In order to combine the collection of binary flow graphs into a general flow graph, Pawlak makes the *flow conservation* assumption [7]. This means that, for an attribute A appearing as a decision attribute in one decision table $\phi_1(C_1, A)$ and also as a conditioning attribute in another decision table $\phi_2(A, D_2)$, we have

$$\sum_{C_1} \phi_1(C_1, A) \ = \ \sum_{D_2} \phi_2(A, D_2).$$

Example 5. The two binary RSFGs in Example 4 satisfy the flow conservation assumption, since in Table 3, $\phi_1(D) = \phi_2(D)$. For instance, $\phi_1(D = \text{"}Alice\text{"}) = 0.170 \ = \phi_2(D = \text{"}Alice\text{"})$.

A *rough set flow graph* (RSFG) [7,8] is a DAG, where each edge is associated with the strength, certainty and coverage coefficients from a collection of decision tables satisfying the flow conservation assumption.

Example 6. The RSFG for the decision tables in Table 2 is the DAG in Fig. 2 together with the strength, certainty and coverage coefficients in Table 3.

The task of RSFG inference is to compute a binary RSFG on $\{A_i, A_j\}$, namely, a DAG on $\{A_i, A_j\}$ and the coefficient table, denoted $Ans(A_i, A_j)$, which is a table with strength, certainty and coverage columns. We use the term *query* to refer to any request involving strength, certainty or coverage.

Example 7. Consider a query on $\{M, P\}$ posed to the RSFG in Example 6. The answer to this query is the binary RSFG defined by Table 4 and Fig. 3.

Table 3: The top two tables are the *strength* $\phi(a_i, a_j)$, *certainty* $\phi(a_j|a_i)$ and *coverage* $\phi(a_i|a_j)$ coefficients for the edges (a_i, a_j) in Fig. 1. These two tables together with the bottom two tables are the coefficients for the edges in Fig. 2.

M	D	$\phi_1(M,D)$	$\phi_1(D\|M)$	$\phi_1(M\|D)$
Toyota	Alice	0.120	0.600	0.710
Toyota	Bob	0.060	0.300	0.160
Toyota	Dave	0.020	0.100	0.070
Honda	Bob	0.150	0.500	0.420
Honda	Carol	0.150	0.500	0.750
Ford	Alice	0.050	0.100	0.290
Ford	Bob	0.150	0.300	0.420
Ford	Carol	0.050	0.100	0.250
Ford	Dave	0.250	0.500	0.930

D	A	$\phi_2(D,A)$	$\phi_2(A\|D)$	$\phi_2(D\|A)$
Alice	Old	0.050	0.300	0.230
Alice	Middle	0.100	0.600	0.190
Alice	Young	0.020	0.100	0.080
Bob	Old	0.140	0.400	0.630
Bob	Middle	0.220	0.600	0.420
Carol	Middle	0.120	0.600	0.230
Carol	Young	0.080	0.400	0.310
Dave	Old	0.030	0.100	0.140
Dave	Middle	0.080	0.300	0.150
Dave	Young	0.160	0.600	0.620

A	S	$\phi_3(A,S)$	$\phi_3(S\|A)$	$\phi_3(A\|S)$
Old	High	0.133	0.600	0.506
Old	Medium	0.067	0.300	0.156
Old	Low	0.022	0.100	0.072
Middle	High	0.104	0.200	0.395
Middle	Medium	0.311	0.600	0.723
Middle	Low	0.104	0.200	0.339
Young	High	0.026	0.100	0.099
Young	Medium	0.052	0.200	0.121
Young	Low	0.181	0.700	0.589

S	P	$\phi_4(S,P)$	$\phi_4(P\|S)$	$\phi_4(S\|P)$
High	Executive	0.210	0.800	0.929
High	Staff	0.045	0.170	0.101
High	Manager	0.008	0.030	0.024
Medium	Executive	0.013	0.030	0.058
Medium	Staff	0.387	0.900	0.872
Medium	Manager	0.030	0.070	0.091
Low	Executive	0.003	0.010	0.013
Low	Staff	0.012	0.040	0.027
Low	Manager	0.292	0.950	0.885

1: Algorithm 1. [7,8]

input : A RSFG and a query on $\{A_i, A_j\},\ i < j$.
output: The coefficient table $Ans(A_i, A_j)$ of the binary RSFG on $\{A_i, A_j\}$.
$\phi(A_j|A_i) = \sum_{A_{i+1},\ldots,A_{j-1}} \phi(A_{i+1}|A_i) \cdot \phi(A_{i+2}|A_{i+1}) \cdot \ldots \cdot \phi(A_j|A_{j-1})$;
$\phi(A_i|A_j) = \sum_{A_{i+1},\ldots,A_{j-1}} \phi(A_i|A_{i+1}) \cdot \phi(A_{i+1}|A_{i+2}) \cdot \ldots \cdot \phi(A_{j-1}|A_j)$;
$\phi(A_i, A_j) = \phi(A_i) \cdot \phi(A_j|A_i)$;
return($Ans(A_i, A_j)$);

Pawlak proposed Algorithm 1 to answer queries in a RSFG.

Algorithm 1 is used to compute the coefficient table of the binary RSFG on $\{A_i, A_j\}$. The DAG of this binary RSFG has an edge (a_i, a_j) provided that $\phi(a_i, a_j) > 0$ in $Ans(A_i, A_j)$. We illustrate Algorithm 1 with Example 8.

Example 8. Given a query on $\{M, P\}$ posed to the RSFG in Fig. 2. Let us focus on $M =$ "*Ford*" and $P =$ "*Staff*", which we succinctly write as "*Ford*" and "*Staff*", respectively. The certainty $\phi($"*Staff*"$|$"*Ford*"$)$ is computed as:

$$\phi(\text{``}\mathit{Staff}\text{''}|\text{``}\mathit{Ford}\text{''}) = \sum_{D,A,S} \phi(D|\text{``}\mathit{Ford}\text{''}) \cdot \phi(A|D) \cdot \phi(S|A) \cdot \phi(\text{``}\mathit{Staff}\text{''}|S).$$

The coverage $\phi($"*Ford*"$|$"*Staff*"$)$ is computed as:

$$\phi(\text{``}\mathit{Ford}\text{''}|\text{``}\mathit{Staff}\text{''}) = \sum_{D,A,S} \phi(\text{``}\mathit{Ford}\text{''}|D) \cdot \phi(D|A) \cdot \phi(A|S) \cdot \phi(S|\text{``}\mathit{Staff}\text{''}).$$

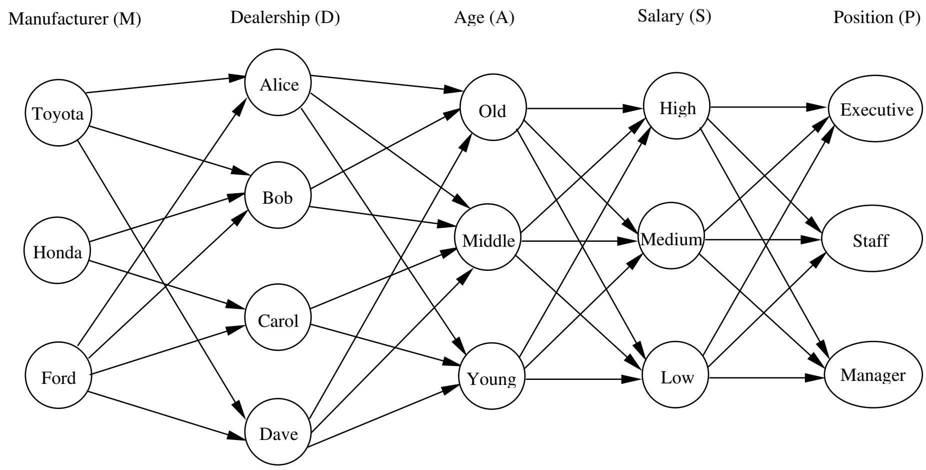

Fig. 2: The *rough set flow graph* (RSFG) for $\{M, D, A, S, P\}$, where the strength, certainty and coverage coefficient are given in Table 3.

The strength $\phi(\text{“}Ford\text{”}, \text{“}Staff\text{”})$ is computed as:

$$\phi(\text{“}Ford\text{”}, \text{“}Staff\text{”}) = \phi(\text{“}Ford\text{”}) \cdot \phi(\text{“}Staff\text{”}|\text{“}Ford\text{”}).$$

The DAG of this binary RSFG on $\{M, P\}$ is depicted in Fig. 3.

In Example 8, computing coefficients $\phi(\text{“}Ford\text{”}, \text{“}Staff\text{”})$, $\phi(\text{“}Staff\text{”}|\text{“}Ford\text{”})$ and $\phi(\text{“}Ford\text{”}|\text{“}Staff\text{”})$ in $Ans(M, P)$ in Table 4 required 181 multiplications and 58 additions. No study, however, has formalized the time complexity of Algorithm 1.

Table 4: Answering a query on $\{M, P\}$ posed to the RSFG in Fig. 2 consists of this coefficient table $Ans(M, P)$ and the DAG in Fig. 3.

M	P	$\phi(M, P)$	$\phi(P\|M)$	$\phi(M\|P)$
Toyota	Executive	0.053132	0.265660	0.234799
Toyota	Staff	0.095060	0.475300	0.214193
Toyota	Manager	0.051808	0.259040	0.157038
Honda	Executive	0.067380	0.224600	0.297764
Honda	Staff	0.140820	0.469400	0.317302
Honda	Manager	0.091800	0.306000	0.278259
Ford	Executive	0.105775	0.211550	0.467437
Ford	Staff	0.207925	0.415850	0.468505
Ford	Manager	0.186300	0.372600	0.564703

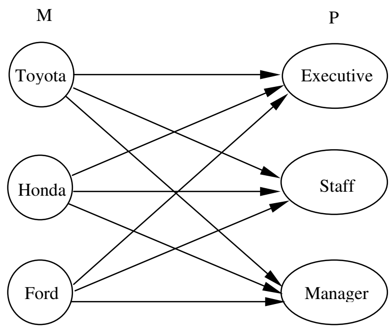

Fig. 3: Answering a query on $\{M, P\}$ posed to the RSFG in Fig. 2 consists of the coefficient table $Ans(M, P)$ in Table 4 and this DAG on $\{M, P\}$.

3 Complexity of Traditional Algorithm in RSFG

In this section, we establish the time complexity of Algorithm 1.

Theorem 2. *Consider a RSFG on m variables $U = \{A_1, A_2, \ldots, A_m\}$. Let $|dom(A_i)| = n$, for $i = 1, \ldots, m$. Let (a_i, a_{i+1}) be an edge in the RSFG, where $a_i \in dom(A_i)$, $a_{i+1} \in dom(A_{i+1})$ and $i = 1, \ldots, m-1$. To answer a query on $\{A_i, A_j\}$, the time complexity of Algorithm 1 is $O(ln^l)$, where $l = j - i + 1$.*

Proof. To compute the certainty $\phi(A_j|A_i)$, let

$$\psi_1(A_i, A_{i+1}, \ldots, A_j) \;=\; \phi(A_{i+1}|A_i) \cdot \phi(A_{i+2}|A_{i+1}) \cdot \ldots \cdot \phi(A_j|A_{j-1}). \quad (1)$$

The potential $\psi_1(A_i, A_{i+1}, \ldots, A_j)$ has n^l rows, since $|dom(A_i)| = n$ for each variable. By Equation (1), computing the certainty for one row requires $l-2$ multiplications. Therefore, $\psi_1(A_i, A_{i+1}, \ldots, A_j)$ is constructed by $(l-2)(n^l)$ multiplications. The second step is to determine

$$\phi(A_j|A_i) \;=\; \sum_{A_{i+1}, \ldots, A_{j-1}} \psi_1(A_i, A_{i+1}, \ldots, A_j). \quad (2)$$

There are exactly n^{l-2} rows in $\psi_1(A_i, A_{i+1}, \ldots, A_j)$ with $A_i = a_i$ and $A_j = a_j$. Thus, computing $\phi(A_j = a_j|A_i = a_i)$ requires $n^{l-2} - 1$ additions. Since there are n^2 configurations in $\phi(A_j|A_i)$, to compute $\phi(A_j|A_i)$ requires $(n^2)(n^{l-2} - 1)$ additions. That is, $n^l - n^2$ additions are required for Equation (2). As shown above, the complexity to compute Equation (1) is $O(ln^l)$ and that to compute Equation (2) is $O(n^l)$. Therefore, computing the certainty $\phi(A_j|A_i)$ has time complexity $O(ln^l)$. It is easily seen that computing the coverage $\phi(A_i|A_j)$ requires exactly the same amount of work as required for computing the certainty $\phi(A_j|A_i)$. Thus, computing the coverage $\phi(A_i|A_j)$ has time complexity $O(ln^l)$. The strength $\phi(A_i, A_j)$ is defined as the product $\phi(A_i) \cdot \phi(A_j|A_i)$, which involves n^2 multiplications. Since the computation of Algorithm 1 is dominated by that for certainty (coverage), the time complexity is $O(ln^l)$. □

The exponential time complexity of Algorithm 1 lies in the fact that it does not exploit the factorization during inference. However, this does not mean that Algorithm 1 is always inefficient in all practical situations.

4 An Efficient Algorithm for RSFG Inference

In this section, we will introduce an efficient algorithm to answer queries in a RSFG and establish its complexity.

The main idea is to exploit the factorization to eliminate variables one by one, instead of all at once as Algorithm 1 does. We focus on computing the coefficient table $Ans(A_i, A_j)$ with the DAG of the output RSFG understood.

2: **Algorithm 2.**

input : A RSFG and a query on $\{A_i, A_j\},\ i < j$.
output: The coefficient table $Ans(A_i, A_j)$ of the binary RSFG on $\{A_i, A_j\}$.
for $k = (i+1)$ *to* $(j-1)$ **do**
  $\phi(A_{k+1}|A_i) = \sum_{A_k} \phi(A_k|A_i) \cdot \phi(A_{k+1}|A_k)$;
  $\phi(A_i|A_{k+1}) = \sum_{A_k} \phi(A_i|A_k) \cdot \phi(A_k|A_{k+1})$;
end
$\phi(A_i, A_j) = \phi(A_i) \cdot \phi(A_j|A_i)$;
return($Ans(A_i, A_j)$);

We illustrate Algorithm 2 with the following example.

Example 9. Recall Example 8. Again, we focus on the edge (“*Ford* ”, “*Staff* ”) in the DAG in Fig. 3. According to Algorithm 2, variables $\{D, A, S\}$ need be eliminated. Consider variable D. The certainty $\phi(A|\text{“}\mathit{Ford}\text{”})$ is

$$\phi(A|\text{“}\mathit{Ford}\text{”}) = \sum_D \phi(D|\text{“}\mathit{Ford}\text{”}) \cdot \phi(A|D),$$

while the coverage $\phi(\text{“}\mathit{Ford}\text{”}|A)$ is

$$\phi(\text{“}\mathit{Ford}\text{”}|A) = \sum_D \phi(\text{“}\mathit{Ford}\text{”}|D) \cdot \phi(D|A).$$

The consequence is that variable D has been eliminated, while variables M and A have been linked via the certainty $\phi(A|\text{“}\mathit{Ford}\text{”})$ and coverage $\phi(\text{“}\mathit{Ford}\text{”}|A)$. Similarly, eliminating A yields $\phi(S|\text{“}\mathit{Ford}\text{”})$ and $\phi(\text{“}\mathit{Ford}\text{”}|S)$. Finally, consider eliminating variable S. The certainty $\phi(\text{“}\mathit{Staff}\text{”}|\text{“}\mathit{Ford}\text{”})$ is

$$\phi(\text{“}\mathit{Staff}\text{”}|\text{“}\mathit{Ford}\text{”}) = \sum_S \phi(S|\text{“}\mathit{Ford}\text{”}) \cdot \phi(\text{“}\mathit{Staff}\text{”}|S),$$

while the coverage $\phi(\text{“}\textit{Ford}\text{”}|\text{“}\textit{Staff}\text{”})$ is

$$\phi(\text{“}\textit{Ford}\text{”}|\text{“}\textit{Staff}\text{”}) = \sum_{S} \phi(\text{“}\textit{Ford}\text{”}|S) \cdot \phi(S|\text{“}\textit{Staff}\text{”}).$$

The strength $\phi(\text{“}\textit{Ford}\text{”}, \text{“}\textit{Staff}\text{”})$ is determined as

$$\phi(\text{“}\textit{Ford}\text{”}, \text{“}\textit{Staff}\text{”}) = \phi(\text{“}\textit{Ford}\text{”}) \cdot \phi(\text{“}\textit{Staff}\text{”}|\text{“}\textit{Ford}\text{”}).$$

In Example 9, computing $\phi(\text{“}\textit{Ford}\text{”}, \text{“}\textit{Staff}\text{”})$, $\phi(\text{“}\textit{Staff}\text{”}|\text{“}\textit{Ford}\text{”})$ and $\phi(\text{“}\textit{Ford}\text{”}|\text{“}\textit{Staff}\text{”})$ in $Ans(M, P)$ in Table 4 only required 45 multiplications and 30 additions. Recall that Algorithm 1 required 181 multiplications and 58 additions.

5 Theoretical Foundation

In this section, we show correctness of Algorithm 2 and prove Algorithm 2 is efficient by analyzing its time complexity in the worst case.

5.1 Correctness of the New RSFG Inference Algorithm

Here we prove that Algorithm 2 is correct. Let us first review two well known results.

Lemma 1. *[12] If ϕ is a potential on U, and $X \subseteq Y \subseteq U$, then marginalizing ϕ onto Y and subsequently onto X is the same as marginalizing ϕ onto X.*

Lemma 1 indicates that a marginal can be obtained by a series of marginalizations in any order. For example,

$$\sum_{A,B} \phi(A, B, C) \quad = \quad \sum_{A} \left(\sum_{B} \phi(A, B, C)\right) \quad = \quad \sum_{B} \left(\sum_{A} \phi(A, B, C)\right).$$

Lemma 2. *[12] If ϕ is a potential on X and ψ is a potential on Y, then the marginalization of $\phi \cdot \psi$ onto X is the same as ϕ multiplied with the marginalization of ψ onto $X \cap Y$.*

For instance,

$$\sum_{C} \phi(A, B) \cdot \phi(B, C) \quad = \quad \phi(A, B) \cdot \sum_{C} \phi(B, C).$$

Now let us turn to the correctness of Algorithm 2.

Theorem 3. *Given a query on $\{A_i, A_j\}$ posed to a RSFG on $U = \{A_1, A_2, \ldots, A_m\}$, where $1 \leq i < j \leq m$. The answer produced by Algorithm 2 is correct.*

Proof. We show the claim by proving that the answer table $Ans(A_i, A_j)$ produced by Algorithm 2 contains the strength $\phi(A_i, A_j)$, the certainty $\phi(A_j|A_i)$ and the coverage $\phi(A_i|A_j)$ computed by Algorithm 1. To answer the certainty $\phi(A_j|A_i)$, Algorithm 1 is expressed by Equation (3),

$$\phi(A_j|A_i) = \sum_{A_{i+1}, A_{i+2}, \ldots, A_{j-1}} \phi(A_{i+1}|A_i) \cdot \phi(A_{i+2}|A_{i+1}) \cdot \ldots \cdot \phi(A_j|A_{j-1}). \quad (3)$$

By Lemma 1 and Equation (3), $\phi(A_j|A_i)$ is equal to

$$\sum_{A_{i+1}} \sum_{A_{i+2}} \cdots \sum_{A_{j-2}} \sum_{A_{j-1}} \phi(A_{i+1}|A_i) \cdot \phi(A_{i+2}|A_{i+1}) \cdot \ldots \cdot \phi(A_j|A_{j-1}). \quad (4)$$

By Lemma 2 and Equation (4), $\phi(A_j|A_i)$ is equal to

$$\sum_{A_{i+1}} \sum_{A_{i+2}} \cdots \sum_{A_{j-2}} \phi(A_{i+1}|A_i) \cdot \ldots \cdot \phi(A_{j-2}|A_{j-3}) \cdot \sum_{A_{j-1}} \phi(A_{j-1}|A_{j-2}) \cdot \phi(A_j|A_{j-1}). \quad (5)$$

By recursively using Lemma 2, Equation (5) can be rewritten as,

$$\sum_{A_{i+1}} \phi(A_{i+1}|A_i) \cdot \sum_{A_{i+2}} \phi(A_{i+2}|A_{i+1}) \cdot \ldots \cdot \sum_{A_{j-1}} \phi(A_{j-1}|A_{j-2}) \cdot \phi(A_j|A_{j-1}). \quad (6)$$

By Equations (3) - (6), the computation of the certainty $\phi(A_j|A_i)$ by Algorithm 1 is expressed as,

$$\phi(A_j|A_i) = \sum_{A_{i+1}} \phi(A_{i+1}|A_i) \cdot \ldots \cdot \sum_{A_{j-1}} \phi(A_{j-1}|A_{j-2}) \cdot \phi(A_j|A_{j-1}). \quad (7)$$

Equation (7) is the construction of the certainty $\phi(A_j|A_i)$ in Algorithm 2. It can be similarly shown that the strength $\phi(A_i, A_j)$ and coverage $\phi(A_i|A_j)$ produced by Algorithms 1 and 2 are the same. □

5.2 Complexity of the New RSFG Inference Algorithm

In this subsection, we establish the computational complexity of Algorithm 2.

Theorem 4. *Consider a RSFG on m variables $U = \{A_1, A_2, \ldots, A_m\}$. Let $|dom(A_i)| = n$, for $i = 1, \ldots, m$. Let (a_i, a_{i+1}) be an edge in the RSFG, where $a_i \in dom(A_i)$, $a_{i+1} \in dom(A_{i+1})$ and $i = 1, \ldots, m-1$. To answer a query on $\{A_i, A_j\}$, the time complexity of Algorithm 2 is $O(ln^3)$, where $l = j - i + 1$.*

Proof. The certainty $\phi(A_j|A_i)$ is computed by eliminating each variable A_k between A_i and A_j in the RSFG. For a variable A_k, Algorithm 2 first computes

$$\psi_2(A_{k-1}, A_k, A_{k+1}) \;=\; \phi(A_k|A_{k-1}) \cdot \phi(A_{k+1}|A_k). \quad (8)$$

The potential $\psi_2(A_{k-1}, A_k, A_{k+1})$ has n^3 rows, since $|dom(A_i)| = n$ for each variable. Computing the certainty for one row requires 1 multiplication. Therefore, potential $\psi_2(A_{k-1}, A_k, A_{k+1})$ is constructed by n^3 multiplications. The second step is to determine

$$\phi(A_{k+1}|A_{k-1}) \;=\; \sum_{A_k} \psi_2(A_{k-1}, A_k, A_{k+1}). \tag{9}$$

There are n rows in $\psi_2(A_{k-1}, A_k, A_{k+1})$ with $A_{k-1} = a_{k-1}$ and $A_{k+1} = a_{k+1}$. Thus, computing $\phi(A_{k+1} = a_{k+1}|A_{k-1} = a_{k-1})$ requires $n-1$ additions. Since there are n^2 configurations in $\phi(A_{k+1}|A_{k-1})$, $(n^2)(n-1)$ additions are required to compute $\phi(A_{k+1}|A_{k-1})$ in Equation (9). Therefore, the time complexity to compute the certainty $\phi(A_{k+1}|A_{k-1})$ is $O(n^3)$. Since there are $l-2$ variables between A_i and A_j, the time complexity to compute the desired certainty $\phi(A_j|A_i)$ has time complexity $O(ln^3)$. Similar to the proof of Theorem 2, it follows that the time complexity of Algorithm 2 is $O(ln^3)$. $\Box$

Theorem 4 shows that Algorithm 2 has polynomial time complexity in the worst case. Therefore, Algorithm 2 is an efficient algorithm for RSFG inference in all practical situations.

6 Related Work

In this section, we show Algorithm 2 never performs more work than Algorithm 1. To show this claim let us first characterize the computation performed by Algorithm 1 and Algorithm 2 when answering a query.

We need only focus on how the certainty $\phi(A_j|A_i)$ is computed from a RSFG on $U = \{A_1, A_2, \ldots, A_m\}$ with certainties $\phi(A_2|A_1), \phi(A_3|A_2), \ldots, \phi(A_m|A_{m-1})$. For simplicity, we eliminate variables in the following order: $A_{j-1}, A_{j-2}, \ldots, A_{i+1}$.

Algorithm 1 computes the following product $\psi_1(A_i, A_{i+1}, \ldots, A_j)$:

$$\begin{aligned} & \psi_1(A_i, A_{i+1}, \ldots, A_j) \\ = \;& \phi(A_{i+1}|A_i) \cdot \ldots \cdot \phi(A_{j-2}|A_{j-3}) \cdot \phi(A_{j-1}|A_{j-2}) \cdot \phi(A_j|A_{j-1}) \end{aligned}$$

via a series of binary multiplications, namely,

$$\begin{aligned} & \psi_1(A_i, A_{i+1}, \ldots, A_j) \\ = \;& \phi(A_{i+1}|A_i) \cdot [\ldots \cdot [\phi(A_{j-2}|A_{j-3}) \cdot [\phi(A_{j-1}|A_{j-2}) \cdot \phi(A_j|A_{j-1})]] \ldots]. \end{aligned} \tag{10}$$

According to Equation (10), the first multiplication is as follows,

$$\psi_1(A_{j-2}, A_{j-1}, A_j) \;=\; \phi(A_{j-1}|A_{j-2}) \cdot \phi(A_j|A_{j-1}). \tag{11}$$

The intermediate multiplications are performed as follows,

$$\psi_1(A_{k-1}, A_k, \ldots, A_j) \;=\; \phi(A_k|A_{k-1}) \cdot \psi_1(A_k, A_{k+1}, \ldots, A_j), \tag{12}$$

where $k = (j-2), \ldots, (i+1)$.

After computing $\psi_1(A_i, A_{i+1}, \ldots, A_j)$, Algorithm 1 eliminates variables A_{i+1}, $A_{i+2}, \ldots, A_{j-1}$ via a series of marginalizations, namely,

$$\sum_{A_{i+1}} \sum_{A_{i+2}} \cdots \sum_{A_{j-1}} \psi_1(A_i, A_{i+1}, \ldots, A_j).$$

An intermediate marginalization takes the form,

$$\psi_1(A_i, \ldots, A_{l-1}, A_j) \;=\; \sum_{A_l} \psi_1(A_i, \ldots, A_{l-1}, A_l, A_j), \tag{13}$$

where $l = (j-1), \ldots, (i+2)$. The final marginalization yields

$$\phi(A_j|A_i) \;=\; \sum_{A_{i+1}} \psi_1(A_i, A_{i+1}, A_j). \tag{14}$$

Now consider how Algorithm 2 computes the certainty $\phi(A_j|A_i)$. As previously mentioned, Algorithm 2 eliminates variables $A_{j-1}, \ldots, A_{i+1}$ one by one. Algorithm 2 computes,

$$\begin{aligned} &\phi(A_j|A_i) \\ =\; &\sum_{A_{i+1}} \phi(A_{i+1}|A_i) \cdot \ldots \cdot \sum_{A_{j-2}} \phi(A_{j-2}|A_{j-3}) \cdot \sum_{A_{j-1}} \phi(A_{j-1}|A_{j-2}) \cdot \phi(A_j|A_{j-1}). \end{aligned} \tag{15}$$

According to Equation (15), the first multiplication in Algorithm 2 is,

$$\psi_2(A_{j-2}, A_{j-1}, A_j) \;=\; \phi(A_{j-1}|A_{j-2}) \cdot \phi(A_j|A_{j-1}). \tag{16}$$

Algorithm 2 then performs intermediate additions and multiplications, iteratively,

$$\begin{aligned} \phi(A_j|A_{j-2}) \;&=\; \sum_{A_{j-1}} \psi_2(A_{j-2}, A_{j-1}, A_j), \\ \psi_2(A_{j-3}, A_{j-2}, A_j) \;&=\; \phi(A_{j-2}|A_{j-3}) \cdot \phi(A_j|A_{j-2}), \\ \phi(A_j|A_{j-3}) \;&=\; \sum_{A_{j-2}} \psi_2(A_{j-3}, A_{j-2}, A_j), \\ &\;\vdots \\ \psi_2(A_i, A_{i+1}, A_j) \;&=\; \phi(A_{i+1}|A_i) \cdot \phi(A_j|A_{i+1}). \end{aligned}$$

Therefore, an intermediate marginalization takes the form,

$$\phi(A_j|A_{l-1}) \;=\; \sum_{A_l} \psi_2(A_{l-1}, A_l, A_j), \tag{17}$$

where $l = (j-1), \ldots, (i+2)$. An intermediate multiplication takes the form,

$$\psi_2(A_{k-1}, A_k, A_j) \;=\; \phi(A_k|A_{k-1}) \cdot \phi(A_j|A_k), \tag{18}$$

where $k = (j-2), \ldots, (i+1)$. After these intermediate additions and multiplications, the final marginalization yields the desired certainty $\phi(A_j|A_i)$:

$$\phi(A_j|A_i) \;=\; \sum_{A_{i+1}} \psi_2(A_i, A_{i+1}, A_j). \tag{19}$$

Lemma 3 shows that the intermediate potentials computed in the multiplication process of Algorithm 2 are marginalizations of the larger potentials computed in Algorithm 1. Lemma 4 shows that the intermediate potentials computed in the marginalization process of Algorithm 2 have no more rows than the marginalizations of the larger potentials computed in Algorithm 1.

Lemma 3. *To answer a query on $\{A_i, A_j\}$ posed to a RSFG on $U = \{A_1, A_2, \ldots, A_m\}$, $\phi(A_j|A_k)$ in Equation (18) of Algorithm 2 is a marginal of $\psi_1(A_k, A_{k+1}, \ldots, A_j)$ in Equation (12) of Algorithm 1.*

Proof. By definition, the marginal of $\psi_1(A_k, A_{k+1}, \ldots, A_j)$ onto $\{A_k, A_j\}$ is:

$$\sum_{A_{k+1},\ldots,A_{j-1}} \psi_1(A_k, A_{k+1}, \ldots, A_j). \tag{20}$$

By Algorithm 1, Equation (20) is equal to,

$$\sum_{A_{k+1},\ldots,A_{j-1}} \phi(A_{k+1}|A_k) \cdot \ldots \cdot \phi(A_{j-1}|A_{j-2}) \cdot \phi(A_j|A_{j-1}). \tag{21}$$

By Lemmas 1 and 2, Equation (21) can be rewritten as:

$$\sum_{A_{k+1}} \phi(A_{k+1}|A_k) \cdot \ldots \cdot \sum_{A_{j-1}} \phi(A_{j-1}|A_{j-2}) \cdot \phi(A_j|A_{j-1}). \tag{22}$$

By Equation (7),

$$\phi(A_j|A_k) \;=\; \sum_{A_{k+1}} \phi(A_{k+1}|A_k) \cdot \ldots \cdot \sum_{A_{j-1}} \phi(A_{j-1}|A_{j-2}) \cdot \phi(A_j|A_{j-1}). \tag{23}$$

By Equations (20) - (23),

$$\phi(A_j|A_k) \;=\; \sum_{A_{k+1},\ldots,A_{j-1}} \psi_1(A_k, A_{k+1}, \ldots, A_j).$$

Therefore, $\phi(A_j|A_k)$ is the marginal of $\psi_1(A_k, A_{k+1}, \ldots, A_j)$ onto variables $\{A_k, A_j\}$. □

Lemma 4. *To answer a query on $\{A_i, A_j\}$ posed to a RSFG on $U = \{A_1, A_2, \ldots, A_m\}$, $\psi_2(A_{l-1}, A_l, A_j)$ in Equation (17) of Algorithm 2 has no more rows than the marginal of $\psi_1(A_i, \ldots, A_{l-1}, A_l, A_j)$ in Equation (13) of Algorithm 1 onto variables $\{A_{l-1}, A_l, A_j\}$.*

Proof. By definition, the marginal of $\psi_1(A_i, \ldots, A_{l-1}, A_l, A_j)$ onto variables $\{A_{l-1}, A_l, A_j\}$ is:

$$\sum_{A_i,\ldots,A_{l-2}} \psi_1(A_i, \ldots, A_{l-1}, A_l, A_j). \tag{24}$$

By Algorithm 1, Equation (24) is equal to,

$$\sum_{A_i,\ldots,A_{l-2}} \phi(A_{i+1}|A_i) \cdot \ldots \cdot \phi(A_{l-2}|A_{l-3}) \cdot \phi(A_{l-1}|A_{l-2}) \cdot \phi(A_l|A_{l-1}) \cdot \phi(A_j|A_l). \tag{25}$$

By Lemma 2, Equation (25) is equal to,

$$\phi(A_l|A_{l-1}) \cdot \phi(A_j|A_l) \cdot \sum_{A_i,\ldots,A_{l-2}} \phi(A_{i+1}|A_i) \cdot \ldots \cdot \phi(A_{l-2}|A_{l-3}) \cdot \phi(A_{l-1}|A_{l-2}). \tag{26}$$

By Lemmas 1 and 2, Equation (26) can be rewritten as:

$$\phi(A_l|A_{l-1}) \cdot \phi(A_j|A_l) \cdot \sum_{A_i} (\sum_{A_{i+1}} \phi(A_{i+1}|A_i) \cdot \ldots \cdot \sum_{A_{l-2}} \phi(A_{l-2}|A_{l-3}) \cdot \phi(A_{l-1}|A_{l-2})). \tag{27}$$

By Equation (7), $\sum_{A_{i+1}} \phi(A_{i+1}|A_i) \cdot \ldots \cdot \sum_{A_{l-2}} \phi(A_{l-2}|A_{l-3}) \cdot \phi(A_{l-1}|A_{l-2})$ yields $\phi(A_{l-1}|A_i)$. Thus, Equation (27) can be rewritten as:

$$\phi(A_l|A_{l-1}) \cdot \phi(A_j|A_l) \cdot \sum_{A_i} \phi(A_{l-1}|A_i). \tag{28}$$

By Equation (18),

$$\psi_2(A_{l-1}, A_l, A_j) \;=\; \phi(A_l|A_{l-1}) \cdot \phi(A_j|A_l). \tag{29}$$

Substituting Equation (29) into Equation (28), we obtain:

$$\psi_2(A_{l-1}, A_l, A_j) \cdot \sum_{A_i} \phi(A_{l-1}|A_i).) \tag{30}$$

By Equations (24) - (30),

$$\sum_{A_i,\ldots,A_{l-2}} \psi_1(A_i, \ldots, A_{l-1}, A_l, A_j) \;=\; \psi_2(A_{l-1}, A_l, A_j) \cdot \sum_{A_i} \phi(A_{l-1}|A_i).$$

Therefore, $\psi_2(A_{l-1}, A_l, A_j)$ has no more rows than the marginal of $\psi_1(A_i, \ldots, A_{l-1}, A_l, A_j)$ onto variables $\{A_{l-1}, A_l, A_j\}$. $\Box$

We use the above analysis to show the following two results. Lemma 5 says that Algorithm 2 never performs more multiplications than Algorithm 1 when answering a query. Lemma 6 says the same except for additions.

Lemma 5. *Given a query on $\{A_i, A_j\}$ posed to a RSFG on $U = \{A_1, A_2, \ldots, A_m\}$, Algorithm 2 never performs more multiplications than Algorithm 1.*

Proof. It can be seen from Equations (11) and (16) that Algorithms 1 and 2 use the same number of multiplications to compute the first potential $\psi_1(A_{j-2}, A_{j-1}, A_j)$ and $\psi_2(A_{j-2}, A_{j-1}, A_j)$. Therefore, Algorithm 1 and Algorithm 2 perform the same number of multiplications provided that precisely two potentials need be multiplied to answer a query. On the other hand, Algorithm 2 never performs more multiplications than Algorithm 1 provided that there are at least three potentials to be multiplied. By Lemma 3, $\phi(A_j|A_k)$ is the marginal of $\psi_1(A_k, A_{k+1}, \ldots, A_j)$ onto $\{A_k, A_j\}$. Therefore, all multiplications in Equation (18) performed by Algorithm 2 for computing the certainty $\phi(A_j|A_i)$ must necessarily be performed in Equation (12) by Algorithm 1. It can be similarly shown that Algorithm 2 never performs more multiplications than Algorithm 1 when computing the strength $\phi(A_i, A_j)$ or coverage $\phi(A_i|A_j)$. Therefore, Algorithm 2 never performs more multiplications than Algorithm 1 when answering a query. □

Lemma 6. *Given a query on $\{A_i, A_j\}$ posed to a RSFG on $U = \{A_1, A_2, \ldots, A_m\}$, Algorithm 2 never performs more additions than Algorithm 1.*

Proof. It can be seen from Equations (14) and (19) that Algorithms 1 and 2 use the same number of additions to eliminate the last variable A_{i+1} from the potential $\psi_1(A_i, A_{i+1}, A_j)$ and $\psi_2(A_i, A_{i+1}, A_j)$. Therefore, Algorithm 1 and Algorithm 2 perform the same number of additions provided that precisely one variable need be eliminated to answer a query. On the other hand, Algorithm 2 never performs more additions than Algorithm 1, provided that there are at least two variables to be eliminated. By Lemma 4, $\psi_2(A_{l-1}, A_l, A_j)$ has no more rows than the marginal of $\psi_1(A_i, \ldots, A_{l-1}, A_l, A_j)$ onto $\{A_{l-1}, A_l, A_j\}$. Therefore, summing out A_l from $\psi_2(A_{l-1}, A_l, A_j)$ combines no more rows than needed from $\psi_1(A_i, \ldots, A_{l-1}, A_l, A_j)$. Since combining n rows requires $n-1$ additions, Algorithm 2 never performs more additions than Algorithm 1 for computing the certainty $\phi(A_j|A_i)$. That Algorithm 2 never performs more additions than Algorithm 1 when computing the strength $\phi(A_i, A_j)$ or coverage $\phi(A_i|A_j)$ follows in a similar fashion. Therefore, Algorithm 2 never performs more additions than Algorithm 1 when answering a query. □

Lemmas 5 and 6 indicate that Algorithm 2 never performs more work than Algorithm 1.

7 Other Remarks on Rough Set Flow Graphs

One salient feature of rough sets is that they serve as a tool for uncertainty management without making assumptions regarding the problem domain. On the contrary, we establish in this section that RSFGs, in fact, make implicit independency assumptions regarding the problem domain.

Two tables $\phi_1(A_i, A_j)$ and $\phi_2(A_j, A_k)$ are *pairwise consistent* [3,13], if

$$\phi_1(A_j) \;=\; \phi_2(A_j). \tag{31}$$

Example 10. In Table 3, $\phi_1(M, D)$ and $\phi_2(D, A)$ are pairwise consistent. For instance, $\phi_1(D = \text{“}Alice\text{”}) = 0.170 = \phi_2(D = \text{“}Alice\text{”})$.

Consider $m-1$ potentials $\phi_1(A_1, A_2), \phi_2(A_2, A_3), \ldots, \phi_{m-1}(A_{m-1}, A_m)$, such that each consecutive pair is pairwise consistent, namely,

$$\phi_i(A_{i+1}) \;=\; \phi_{i+1}(A_{i+1}), \tag{32}$$

for $i = 1, 2, \ldots, m-2$. Observe that the schemas of these decision tables form an *acyclic hypergraph* [1]. Dawid and Lauritzen [3] have shown that if a given set of potentials satisfies Equation (32) and are defined over an acyclic hypergraph, then the potentials are marginals of a unique potential $\phi(A_1, A_2, \ldots, A_m)$, defined as:

$$\phi(A_1, A_2, \ldots, A_m) = \frac{\phi_1(A_1, A_2) \cdot \phi_2(A_2, A_3) \cdot \ldots \cdot \phi_{m-1}(A_{m-1}, A_m)}{\phi_1(A_2) \cdot \ldots \cdot \phi_{m-2}(A_{m-1})}. \tag{33}$$

In [7,8], the flow conservation assumption is made. This means that a given set of $m-1$ decision tables $\phi_1(A_1, A_2), \phi_2(A_2, A_3), \ldots, \phi_{m-1}(A_{m-1}, A_m)$ satisfies Equation (32). By [3], these potentials are marginals of a unique potential $\phi(A_1, A_2, \ldots, A_m)$ defined by Equation (33), which we will call the *collective* potential. The collective potential $\phi(A_1, A_2, \ldots, A_m)$ represents the problem domain from a rough set perspective.

In order to test whether independencies are assumed to hold, it is necessary to normalize $\phi(A_1, A_2, \ldots, A_m)$. (Note that the normalization process has been used in [7,8].) Normalizing $\phi(A_1, A_2, \ldots, A_m)$ yields a jpd $p(A_1, A_2, \ldots, A_m)$ by multiplying $1/N$, where N denotes the number of all cases. It follows from Equation (33) that

$$\begin{aligned} p(A_1, A_2, \ldots, A_m) &= \frac{1}{N} \cdot \phi(A_1, A_2, \ldots, A_m) \\ &= \frac{1}{N} \cdot \frac{\phi_1(A_1, A_2) \cdot \phi_2(A_2, A_3) \cdot \ldots \cdot \phi_{m-1}(A_{m-1}, A_m)}{\phi_1(A_2) \cdot \ldots \cdot \phi_{m-2}(A_{m-1})}. \end{aligned} \tag{34}$$

We now show that RSFGs make implicit independency assumptions regarding the problem domain.

Theorem 5. *Consider a RSFG defined by $m-1$ decision tables $\phi_1(A_1, A_2)$, $\phi_2(A_2, A_3), \ldots, \phi_{m-1}(A_{m-1}, A_m)$. Then $m-2$ probabilistic independencies $I(A_1, A_2, A_3 \ldots A_m), I(A_1 A_2, A_3, A_4 \ldots A_m), \ldots, I(A_1 \ldots A_{m-2}, A_{m-1}, A_m)$ are satisfied by the jpd $p(A_1, A_2, \ldots, A_m)$, where $p(A_1, A_2, \ldots, A_m)$ is the normalization of collective potential $\phi(A_1, A_2, \ldots, A_m)$ representing the problem domain.*

Proof. Consider $I(A_1, A_2, A_3 \ldots A_m)$. By Equation (34), let

$$\phi'(A_1, A_2) \;=\; \phi_1(A_1, A_2) \tag{35}$$

and

$$\phi''(A_2, A_3, \ldots, A_m) \;=\; \frac{1}{N} \cdot \frac{\phi_2(A_2, A_3) \cdot \ldots \cdot \phi_{m-1}(A_{m-1}, A_m)}{\phi_1(A_2) \cdot \ldots \cdot \phi_{m-2}(A_{m-1})}. \tag{36}$$

By substituting Equations (35) and (36) into Equation (34),

$$p(A_1, A_2, \ldots, A_m) \;=\; \phi'(A_1, A_2) \cdot \phi''(A_2, A_3, \ldots, A_m). \tag{37}$$

By Theorem 1, Equation (37) indicates that $I(A_1, A_2, A_3 \ldots A_m)$ holds. It can be similarly shown that $I(A_1A_2, A_3,\ A_4 \ldots A_m), \ldots, I(A_1 \ldots A_{m-2}, A_{m-1}, A_m)$ are also satisfied by the jpd $p(A_1, A_2, \ldots, A_m)$. □

Example 11. Decision tables $\phi(M, D), \phi(D, A), \phi(A, S)$ and $\phi(S, P)$ in Table 2 satisfy Equation (32) and are defined over an acyclic hypergraph $\{MD, DA, AS, SP\}$. This means they are marginals of a unique collective potential,

$$\phi(M, D, A, S, P) = \frac{\phi(M, D) \cdot \phi(D, A) \cdot \phi(A, S) \cdot \phi(S, P)}{\phi(D) \cdot \phi(A) \cdot \phi(S)}. \tag{38}$$

The normalization of $\phi(M, D, A, S, P)$ is a jpd $p(M, D, A, S, P)$,

$$p(M, D, A, S, P) \;=\; \frac{1}{1000} \cdot \frac{\phi(M, D) \cdot \phi(D, A) \cdot \phi(A, S) \cdot \phi(S, P)}{\phi(D) \cdot \phi(A) \cdot \phi(S)}, \tag{39}$$

where the number of all cases $N = 1000$. To show $I(M, D, ASP)$ holds, let

$$\phi'(M, D) = \phi(M, D) \tag{40}$$

and

$$\phi''(D, A, S, P) = \frac{1}{1000} \cdot \frac{\phi(D, A) \cdot \phi(A, S) \cdot \phi(S, P)}{\phi(D) \cdot \phi(A) \cdot \phi(S)}. \tag{41}$$

Substituting Equations (40) and (41) into Equation (39),

$$p(M, D, A, S, P) = \phi'(M, D) \cdot \phi''(D, A, S, P). \tag{42}$$

By Theorem 1, the independence $I(M, D, ASP)$ holds in $p(M, D, A, S, P)$. It can be similarly shown that $I(MD, A, SP)$ and $I(MDA, S, P)$ are also satisfied by $p(M, D, A, S, P)$.

The important point is that the flow conservation assumption [7] used in the construction of RSFGs implicitly implies probabilistic conditional independencies holding in the problem domain.

8 Conclusion

Pawlak [7,8] recently introduced the notion of rough set flow graph (RSFGs) as a graphical framework for reasoning from data. In this paper, we established that the RSFG inference algorithm suggested in [7,8] has exponential time complexity. The root cause of the computational explosion is a failure to exploit the factorization defined by a RSFG during inference. We proposed a new RSFG algorithm exploiting the factorization. We showed its correctness and established its time complexity is polynomial with respect to number of nodes in a RSFG.

In addition, we showed that it never performs more work than the traditional algorithm [7,8]. These are important results, since they indicate that RSFGs are an efficient framework for uncertainty management. Finally, our study has revealed that RSFGs, unlike previous rough set research, make implicit independency assumptions regarding the problem domain. Future work will report on the complexity of the inference in generalized RSFGs [4]. As the order in which variables are eliminated affects the amount of computation performed [6], we will also investigate this issue in RSFGs.

Acknowledgments

The authors would like to thank A. Skowron for constructive comments and suggestions.

References

1. Beeri, C., Fagin, R., Maier, D. and Yannakakis, M.: On The Desirability of Acyclic Database Schemes. Journal of the ACM, 30(3) (1983) 479-513
2. Butz, C.J., Yan, W. and Yang, B.: The Computational Complexity of Inference Using Rough Set Flow Graphs. The Tenth International Conference on Rough Sets, Fuzzy Sets, Data Mining, and Granular Computing, Vol. 1 (2005) 335-344
3. Dawid, A.P. and Lauritzen, S.L.: Hyper Markov Laws in The Statistical Analysis of Decomposable Graphical Models. The Annals of Satistics, Vol. 21 (1993) 1272-1317
4. Greco, S., Pawlak, Z. and Slowinski, R.: Generalized Decision Algorithms, Rough Inference Rules and Flow Graphs. The Third International Conference on Rough Sets, and Current Trends in Computing (2002) 93-104
5. Hajek, P., Havranek T. and Jirousek R.: Uncertain Information Processing in Expert System. (1992)
6. Madson, A.L. and Jensen, F.V.: Lazy Propagation: A Junction Tree Inference Algorithm based on Lazy Evaluation, Artificial Intelligence, 113 (1-2) (1999) 203-245.
7. Pawlak, Z.: Flow Graphs and Decision Algorithms. The Ninth International Conference on Rough Sets, Fuzzy Sets, Data Mining, and Granular Computing (2003) 1-10
8. Pawlak, Z.: In Pursuit of Patterns in Data Reasoning from Data - The Rough Set Way. The Third International Conference on Rough Sets, and Current Trends in Computing (2002) 1-9
9. Pawlak, Z.: Rough Sets. International Journal of Computer and Information Sciences, Vol. 11, Issue 5 (1982) 341-356
10. Pawlak, Z.: Rough Sets: Theoretical Aspects of Reasoning about Data. Kluwer Academic (1991)
11. Pearl, J.: Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference. Morgan Kaufmann, San Francisco, California (1988)
12. Shafer, G.: Probabilistic Expert Systems. Society for the Institute and Applied Mathematics, Philadelphia (1996)
13. Wong, S.K.M., Butz, C.J. and Wu, D.: On the Implication Problem for Probabilistic Conditional Independency, IEEE Transactions on Systems, Man, and Cybernetics, Part A: Systems and Humans, Vol. 30, Issue 6. (2000) 785-805

Appendix I

Table 5: A jpd $p(U)$ is shown in Tables 5 and 6, where $U = \{M, D, A, S, P\}$.

M	D	A	S	P	$p(U)$
Toyota	Alice	Old	High	Executive	0.017280
Toyota	Alice	Old	High	Staff	0.003672
Toyota	Alice	Old	High	Manager	0.000648
Toyota	Alice	Old	Medium	Executive	0.000324
Toyota	Alice	Old	Medium	Staff	0.009720
Toyota	Alice	Old	Medium	Manager	0.000756
Toyota	Alice	Old	Low	Executive	0.000036
Toyota	Alice	Old	Low	Staff	0.000144
Toyota	Alice	Old	Low	Manager	0.003420
Toyota	Alice	Middle	High	Executive	0.011520
Toyota	Alice	Middle	High	Staff	0.002448
Toyota	Alice	Middle	High	Manager	0.000432
Toyota	Alice	Middle	Medium	Executive	0.001296
Toyota	Alice	Middle	Medium	Staff	0.003888
Toyota	Alice	Middle	Medium	Manager	0.003024
Toyota	Alice	Middle	Low	Executive	0.000144
Toyota	Alice	Middle	Low	Staff	0.000576
Toyota	Alice	Middle	Low	Manager	0.013680
Toyota	Alice	Young	High	Executive	0.000960
Toyota	Alice	Young	High	Staff	0.000204
Toyota	Alice	Young	High	Manager	0.000036
Toyota	Alice	Young	Medium	Executive	0.000072
Toyota	Alice	Young	Medium	Staff	0.002160
Toyota	Alice	Young	Medium	Manager	0.000168
Toyota	Alice	Young	Low	Executive	0.000084
Toyota	Alice	Young	Low	Staff	0.000336
Toyota	Alice	Young	Low	Manager	0.007980
Toyota	Bob	Old	High	Executive	0.011520
Toyota	Bob	Old	High	Staff	0.002448
Toyota	Bob	Old	High	Manager	0.000432
Toyota	Bob	Old	Medium	Executive	0.000216
Toyota	Bob	Old	Medium	Staff	0.006480
Toyota	Bob	Old	Medium	Manager	0.000504
Toyota	Bob	Old	Low	Executive	0.000024
Toyota	Bob	Old	Low	Staff	0.000096
Toyota	Bob	Old	Low	Manager	0.002280
Toyota	Bob	Middle	High	Executive	0.005760
Toyota	Bob	Middle	High	Staff	0.001224
Toyota	Bob	Middle	High	Manager	0.000216
Toyota	Bob	Middle	Medium	Executive	0.000648
Toyota	Bob	Middle	Medium	Staff	0.019440
Toyota	Bob	Middle	Medium	Manager	0.001512
Toyota	Bob	Middle	Low	Executive	0.000072
Toyota	Bob	Middle	Low	Staff	0.000288
Toyota	Bob	Middle	Low	Manager	0.006840
Toyota	Dave	Old	High	Executive	0.000960
Toyota	Dave	Old	High	Staff	0.000204
Toyota	Dave	Old	High	Manager	0.000036
Toyota	Dave	Old	Medium	Executive	0.000018
Toyota	Dave	Old	Medium	Staff	0.000540
Toyota	Dave	Old	Medium	Manager	0.000042
Toyota	Dave	Old	Low	Executive	0.000002
Toyota	Dave	Old	Low	Staff	0.000008
Toyota	Dave	Old	Low	Manager	0.000190
Toyota	Dave	Middle	High	Executive	0.000960
Toyota	Dave	Middle	High	Staff	0.000204
Toyota	Dave	Middle	High	Manager	0.000036
Toyota	Dave	Middle	Medium	Executive	0.000108
Toyota	Dave	Middle	Medium	Staff	0.003240
Toyota	Dave	Middle	Medium	Manager	0.000252
Toyota	Dave	Middle	Low	Executive	0.000012
Toyota	Dave	Middle	Low	Staff	0.000048
Toyota	Dave	Middle	Low	Manager	0.001140
Toyota	Dave	Young	High	Executive	0.000960
Toyota	Dave	Young	High	Staff	0.000204
Toyota	Dave	Young	High	Manager	0.000036
Toyota	Dave	Young	Medium	Executive	0.000072
Toyota	Dave	Young	Medium	Staff	0.002160
Toyota	Dave	Young	Medium	Manager	0.000168
Toyota	Dave	Young	Low	Executive	0.000084
Toyota	Dave	Young	Low	Staff	0.000336
Toyota	Dave	Young	Low	Manager	0.007980
Honda	Bob	Old	High	Executive	0.028800
Honda	Bob	Old	High	Staff	0.006120
Honda	Bob	Old	High	Manager	0.001080
Honda	Bob	Old	Medium	Executive	0.000540
Honda	Bob	Old	Medium	Staff	0.016200
Honda	Bob	Old	Medium	Manager	0.001260
Honda	Bob	Old	Low	Executive	0.000060
Honda	Bob	Old	Low	Staff	0.000240
Honda	Bob	Old	Low	Manager	0.005700
Honda	Bob	Middle	High	Executive	0.014400
Honda	Bob	Middle	High	Staff	0.003060
Honda	Bob	Middle	High	Manager	0.000540
Honda	Bob	Middle	Medium	Executive	0.001620
Honda	Bob	Middle	Medium	Staff	0.048600
Honda	Bob	Middle	Medium	Manager	0.003780
Honda	Bob	Middle	Low	Executive	0.000180
Honda	Bob	Middle	Low	Staff	0.000720
Honda	Bob	Middle	Low	Manager	0.017100
Honda	Carol	Middle	High	Executive	0.014400
Honda	Carol	Middle	High	Staff	0.003060
Honda	Carol	Middle	High	Manager	0.000540
Honda	Carol	Middle	Medium	Executive	0.001620
Honda	Carol	Middle	Medium	Staff	0.048600
Honda	Carol	Middle	Medium	Manager	0.003780
Honda	Carol	Middle	Low	Executive	0.000180
Honda	Carol	Middle	Low	Staff	0.000720

Table 6: A jpd $p(U)$ is shown in Tables 5 and 6, where $U = \{M, D, A, S, P\}$.

M	D	A	S	P	$p(U)$	M	D	A	S	P	$p(U)$
Honda	Carol	Middle	Low	Manager	0.017100	Ford	Bob	Middle	Medium	Staff	0.048600
Honda	Carol	Young	High	Executive	0.004800	Ford	Bob	Middle	Medium	Manager	0.003780
Honda	Carol	Young	High	Staff	0.001020	Ford	Bob	Middle	Low	Executive	0.000180
Honda	Carol	Young	High	Manager	0.000180	Ford	Bob	Middle	Low	Staff	0.000720
Honda	Carol	Young	Medium	Executive	0.000360	Ford	Bob	Middle	Low	Manager	0.017100
Honda	Carol	Young	Medium	Staff	0.010800	Ford	Carol	Middle	High	Executive	0.004800
Honda	Carol	Young	Medium	Manager	0.000840	Ford	Carol	Middle	High	Staff	0.001020
Honda	Carol	Young	Low	Executive	0.000420	Ford	Carol	Middle	High	Manager	0.000180
Honda	Carol	Young	Low	Staff	0.001680	Ford	Carol	Middle	Medium	Executive	0.000540
Honda	Carol	Young	Low	Manager	0.039900	Ford	Carol	Middle	Medium	Staff	0.016200
Ford	Alice	Old	High	Executive	0.007200	Ford	Carol	Middle	Medium	Manager	0.001260
Ford	Alice	Old	High	Staff	0.001530	Ford	Carol	Middle	Low	Executive	0.000060
Ford	Alice	Old	High	Manager	0.000270	Ford	Carol	Middle	Low	Staff	0.000240
Ford	Alice	Old	Medium	Executive	0.000135	Ford	Carol	Middle	Low	Manager	0.005700
Ford	Alice	Old	Medium	Staff	0.004050	Ford	Carol	Young	High	Executive	0.001600
Ford	Alice	Old	Medium	Manager	0.000315	Ford	Carol	Young	High	Staff	0.000340
Ford	Alice	Old	Low	Executive	0.000015	Ford	Carol	Young	High	Manager	0.000060
Ford	Alice	Old	Low	Staff	0.000060	Ford	Carol	Young	Medium	Executive	0.000120
Ford	Alice	Old	Low	Manager	0.001425	Ford	Carol	Young	Medium	Staff	0.003600
Ford	Alice	Middle	High	Executive	0.004800	Ford	Carol	Young	Medium	Manager	0.000280
Ford	Alice	Middle	High	Staff	0.001020	Ford	Carol	Young	Low	Executive	0.000140
Ford	Alice	Middle	High	Manager	0.000180	Ford	Carol	Young	Low	Staff	0.000560
Ford	Alice	Middle	Medium	Executive	0.000540	Ford	Carol	Young	Low	Manager	0.013300
Ford	Alice	Middle	Medium	Staff	0.016200	Ford	Dave	Old	High	Executive	0.012000
Ford	Alice	Middle	Medium	Manager	0.001260	Ford	Dave	Old	High	Staff	0.002550
Ford	Alice	Middle	Low	Executive	0.000060	Ford	Dave	Old	High	Manager	0.000450
Ford	Alice	Middle	Low	Staff	0.000240	Ford	Dave	Old	Medium	Executive	0.000225
Ford	Alice	Middle	Low	Manager	0.005700	Ford	Dave	Old	Medium	Staff	0.006750
Ford	Alice	Young	High	Executive	0.000400	Ford	Dave	Old	Medium	Manager	0.000525
Ford	Alice	Young	High	Staff	0.000085	Ford	Dave	Old	Low	Executive	0.000025
Ford	Alice	Young	High	Manager	0.000015	Ford	Dave	Old	Low	Staff	0.000100
Ford	Alice	Young	Medium	Executive	0.000030	Ford	Dave	Old	Low	Manager	0.002375
Ford	Alice	Young	Medium	Staff	0.000900	Ford	Dave	Middle	High	Executive	0.012000
Ford	Alice	Young	Medium	Manager	0.000070	Ford	Dave	Middle	High	Staff	0.002550
Ford	Alice	Young	Low	Executive	0.000035	Ford	Dave	Middle	High	Manager	0.000450
Ford	Alice	Young	Low	Staff	0.000140	Ford	Dave	Middle	Medium	Executive	0.001350
Ford	Alice	Young	Low	Manager	0.003325	Ford	Dave	Middle	Medium	Staff	0.040500
Ford	Bob	Old	High	Executive	0.028800	Ford	Dave	Middle	Medium	Manager	0.003150
Ford	Bob	Old	High	Staff	0.006120	Ford	Dave	Middle	Low	Executive	0.000150
Ford	Bob	Old	High	Manager	0.001080	Ford	Dave	Middle	Low	Staff	0.000600
Ford	Bob	Old	Medium	Executive	0.000540	Ford	Dave	Middle	Low	Manager	0.014250
Ford	Bob	Old	Medium	Staff	0.016200	Ford	Dave	Young	High	Executive	0.012000
Ford	Bob	Old	Medium	Manager	0.001260	Ford	Dave	Young	High	Staff	0.002550
Ford	Bob	Old	Low	Executive	0.000060	Ford	Dave	Young	High	Manager	0.000450
Ford	Bob	Old	Low	Staff	0.000240	Ford	Dave	Young	Medium	Executive	0.000900
Ford	Bob	Old	Low	Manager	0.005700	Ford	Dave	Young	Medium	Staff	0.027000
Ford	Bob	Middle	High	Executive	0.014400	Ford	Dave	Young	Medium	Manager	0.002100
Ford	Bob	Middle	High	Staff	0.003060	Ford	Dave	Young	Low	Executive	0.001050
Ford	Bob	Middle	High	Manager	0.000540	Ford	Dave	Young	Low	Staff	0.004200
Ford	Bob	Middle	Medium	Executive	0.001620	Ford	Dave	Young	Low	Manager	0.099750