A New Inference Algorithm of Dynamic Uncertain Causality Graph Based on Conditional Sampling Method for Complex Cases

HAO NIE\textsuperscript{1} AND QIN ZHANG\textsuperscript{1,2}
\textsuperscript{1}Institute of Nuclear and New Energy Technology, Tsinghua University, Beijing 100084, China
\textsuperscript{2}Department of Computer Science and Technology, Tsinghua University, Beijing 100084, China
Corresponding author: Qin Zhang (qinzhang@tsinghua.edu.cn)

This work was supported by the Institute for Guoqiang, Tsinghua University, under Grant 2020GQ0001.

\textbf{ABSTRACT} Dynamic Uncertain Causality Graph (DUCG) is a recently developed model for fault diagnoses of industrial systems and general clinical diagnoses. In some cases, however, when state-unknown intermediate variables are many, the variable state combination explosion may appear and result in the inefficiency or even disability in DUCG inference. Monte Carlo sampling is a typical algorithm to solve this type of problem. However, since the calculation values are very small, a huge number of samplings are needed. This paper proposes an algorithm based on conditional stochastic simulation, which obtains the final calculation result from the expectation of the conditional probability in sampling cycles instead of counting the sampling frequency. Compared with the early presented recursive algorithm, the proposed algorithm requires much less computation time in the case when state-unknown intermediate variables are many. An example for diagnosing Viral Hepatitis B shows that the new algorithm performs 3 times faster than the recursive algorithm and the error ratio is within 2.7%.

\textbf{INDEX TERMS} Monte Carlo methods, sampling methods, inference algorithms, clinical diagnosis, fault diagnosis.

\section{I. INTRODUCTION}
Computer-aided diagnosis for finding root causes of system abnormalities is desired for large and complex systems, e.g., fault diagnoses of nuclear power plants and general clinical diagnoses. As one of the models, DUCG (dynamic uncertain causality graph) is recently developed [1]–[6]. It is based on domain knowledge with uncertainty and has high diagnosis precision, strong interpretability, and no generalization problem that machine learning approaches usually have.

DUCG is developed from dynamic causality diagram (DCD) [7], formally presented in [1], and evolves in recent years. In [2], the matrix form of DUCG was presented. In [3], the algorithm for breaking the directed cyclic graph (DCG) in DUCG was presented. In [8], an algorithm called cubic-DUCG was introduced for temporal inference of DUCG. In [9], DUCG is extended as intuitionistic fuzzy (IFDUCG) to handle the problem of describing the vagueness and uncertainty of an event. In [10], cloud reasoning DUCG is proposed for handling the fuzziness and randomness of uncertain information simultaneously.

DUCG has been widely applied. In [11], DUCG was applied in the online fault diagnoses of the generator system of a nuclear power plant. In [12], DUCG was applied for the power supply system of spacecraft. In [13], DUCG was applied for fault diagnoses on a gearbox. In [6], DUCG was applied for medical diagnoses, etc.

In practice, however, the problem of state combination explosion caused by state-unknown intermediate variables decreases the online inference computation efficiency. In addition, the very small value of computation in our cases causes the difficulty of applying sampling methods.

In this paper, a new sampling-based algorithm is proposed to boost the inference on these given models. It is composed...
of two parts. The first is a conditional stochastic simulation method based on the recursive inference scheme [5]. The second is a cut-off estimation scheme to simplify the logical operation among states of variables. Detailed analysis on time complexity is given, which shows that the proposed algorithm can increase the inference computation efficiency significantly.

This new algorithm performs well for the large-scale DUCG models. Therefore its deployment can help to support more concurrent diagnosis instances with fewer computation resources, and diagnose with more complicated systems.

In section 2, as a background of this paper, DUCG and the earlier presented recursive inference algorithm are briefly introduced. The analysis on the combination explosion problem is given. In section 3, the detailed scheme for the sampling algorithm is presented. Two typical cases are shown in section 4 to verify the accuracy and performance of the sampling algorithm. Then, an example based on a real case of Viral hepatitis B is given for further comparison. Section 5 concludes this paper.

II. BACKGROUND TECHNOLOGY
A. BRIEF INTRODUCTION TO DUCG

The basic idea of DUCG is to represent the uncertain causal relationships between a child variable and its parent variables through introducing virtual functional random events and causal relationship intensities.

![Figure 1. The basic idea of the DUCG model [1].](image)

Fig.1 and (1),(2) explain the core idea of the DUCG model. \( V \in \{B, X, D, G\} \) represents the parent events. The \( i \) in \( V_{ij} \) indexes the event variable and \( j_i \) indexes the state of the variable. The lower-case letters represent the probabilities of the corresponding upper-case letters denoting events.

\[
x_{nk} = \sum_{i} \sum_{j_i} x_{nk;ij_i} = \sum_{i} \sum_{j_i} f_{nk;ij_i} v_{ij_i} = \sum_{i} \sum_{j} (r_{ni}/r_n)a_{nk;ij}v_{ij} \tag{1}
\]

\[
x_{nk} = \sum_{i} \sum_{j} (r_{ni}/r_n) a_{nk;ij}v_{ij} \tag{2}
\]

For simplicity, \( j_i \) is simplified as \( j \) in the last expression of (1) and in the rest of this paper. In (1), \( r_{ni} > 0 \) is defined as the causal relationship intensity between child variable \( X_n \) and parent variable \( V_i, r_n \equiv \sum_i r_{ni}, A_{nk;ij} \) denotes the causal mechanism that parent event \( V_{ij} \) causes child event \( X_{nk;ij} \) denoting the \( X_{nk} \) caused by \( V_{ij} \). Each evidence event \( X_{nk} \) observed can be expanded to its parent events according to (1), and its parents \( V_{ij} = X_{ij} \) can be further expanded as their parents in the same way. The expanding process will not stop until root causes are denoted as \( B \) or \( D \). Where \( B \)-type event means a basic/root event whose probability is given during DUCG construction, and \( D \)-type event is defined as default or unknown cause event with probability equal to 1. Then the evidence \( E = \prod_{x_{ij} \in E} x_{ij} \) can be expanded as a sum-of-products composed of only \{\( B \)-, \( D \)-, \( A \)-\} type events and \( r \)-type parameters, in which the logic absorption, exclusion, and \( r \)-type parameter calculation apply. During the expanding, once a state-unknown variable is a parent, all its states should be expanded leading to a sum-of-products. A set of state-unknown variables will cause the multiplication of a set of sum-of-products, and then the state combination exploration may occur.

After obtaining the final sum-of-products, its value can be calculated as illustrated in (2), i.e., by replacing the upper-case letters in the sum-of-products in terms of events with the corresponding lower-case letters in terms of probabilities of the events. Furthermore, once we get the sum-of-products of \( E \) in terms of events, the probability of \( E \) can be calculated. Note that the \{\( b \)-, \( a \)-, \( r \)-\} type parameters are encoded in the constructed DUCG. The DUCG inference is to calculate \( Pr\{B_{ij}|E\} = Pr\{B_{ij}E\}/Pr\{E\} \) for all possible \( B_{ij} \) in concern. Therefore, the major computation is \( Pr\{E\} \).

It is the same as the Bayesian network [14] that the inference problem for DUCG is an NP-hard problem. When there are no or only a few state-unknown intermediate variables, the recursive algorithm earlier introduced in [5] and summarized in section 2-B can be applied. However, when the state-unknown intermediate variables are more than a few, the computation efficiency is a big problem.

![Figure 2. State-unknown intermediate variables \( X_{65} \) and \( X_{66} \) in a DUCG about nasal septum deviation [6].](image)

Fig.2 describes an example of a patient suffering a nasal septum deviation (\( B_3 \)) [6]. His case record is: A patient with no history of external head injury (\( X_{64,0} \), does not suffer the headache (\( X_{12,0} \), has persistent nasal obstruction (\( X_{7,1} \), hemorrhina (\( X_{9,1} \), less volume of nasal bleeding (\( X_{101,1} \), and deviation of nasal septum found by physical examination (\( X_{55,1} \), while other symptoms and physical signs were
normal, no laboratory examination and imaging examination result existed.

The definitions of nodes in Fig.2 are listed in Table 1. For simplicity, the node shapes represent the types of variables, e.g., \( \square \) represents \( X \)-type variables. Nodes with other shapes \( \bigcirc, \vartriangle, \odot \) represent for \{\( SG \), \( D \), \( BX \)\}-type variables respectively (definitions of \( SG \) and \( BX \) are given in [6] but not given here because they are not used in the rest of this paper).

#### Table 1. \{\( B, X \}\)-Type variables in Fig.3.

| Variable | Variable description | Variable | Variable description |
|----------|----------------------|----------|----------------------|
| \( X_9 \) | Hemorrhinia          | \( B_1 \) | Nasal septum deviation |
| \( X_7 \) | Nasal obstruction    | \( X_{54} \) | History of external head injury |
| \( X_{12} \) | Headache             | \( X_{101} \) | Volume of nasal bleeding |
| \( X_{66} \) | Nasal mucosal erosion | \( X_{68} \) | Partial compression of ipsilateral turbine |
| \( X_{55} \) | Nasal septum bending (physical examination) |

#### Table 2. Shapes of some nodes in DUCG.

| Type name | Shape | Shape | Definition |
|-----------|-------|-------|------------|
| B-type    | \( B_{ij} \) | \( ij \) | The root cause event. |
| X-type    | \( X_{nk} \) | \( nk \) | The consequence or effect variable. |
| G-type    | \( G_i \) | \( i \) | The logic gate, has a attached table to show the logic conditions. |
| D-type    | \( D_n \) | \( n \) | the default or unclear parent variables of variable \( X_n \). |

#### Table 3. Shape of directed arcs in DUCG.

| Standard Shape | Definition |
|----------------|------------|
| \( \rightarrow \) | The weighted functional event. |
| \( \Rightarrow \) | A linkage event with condition \( Z \). |
| \( \bullet \) | Input for a logic gate. |

In Fig. 3, \( \square \) represents \( B \)-type nodes and \( \vartriangle, \bigcirc, \odot \) represent \( X \)-type events/variables. The query for inference of DUCG is to get the probability for each state of possible root causes conditional on evidence \( E \), i.e. \( Pr\{B_{kj}|E\} \).

In Fig. 2, \( E = X_{171}X_{91}X_{551}X_{1011}X_{2211}X_{2861} \). The basic inference algorithm of DUCG can be roughly summarized as follows:

1) Simplify the DUCG according to the observed evidence \( E \) according to rules presented in [1]. The \( B \)-type events remaining are the possible root causes included in \( S_H \);

2) Expand \( B_{kj}E \) and \( E \) with (1) as sum-of-products composed of \{\( B, D, A, r \)\}-type events/variables and parameters, where \( B_{kj} \in S_H \);

3) When there is more than one member in \( S_H \), calculate the numerical values of:

\[
Pr\{B_{kj}|E\} = \frac{Pr\{B_{kj}E\}}{Pr\{E\}} \quad \text{for all } B_{kj} \in S_H \quad (3)
\]

Fig.3 shows an example of a 5-layer DUCG which has two \( B \)-type nodes and three evidence nodes: \( X_{171} \), \( X_{181} \), and \( X_{191} \).
To calculate (3), we need to expand every member in $E$ and multiply them together. For example, evidence $X_{17,1}$ can be expanded with (1) as follows:

$$X_{17,1} = F_{17,1;13}X_{13} + F_{17,1;13}X_{14} + F_{17,1;13}X_{15} + F_{17,1;13}X_{16}$$

and then, $X_{13}, X_{14}, X_{15},$ and $X_{16}$ can be further expanded to a sum of 4 terms respectively, all including $X_9, X_{10}, X_{11},$ and $X_{12}$. Similarly, $X_9$ through $X_{12}$ can be expanded respectively. For example, $X_9 = F_{9,5}X_5 + F_{9,6}X_6 + F_{9,7}X_7 + F_{9,8}X_8$. Note that $F_{17,1;13}$ through $F_{17,1;15}$ and $F_{9,5}$ through $F_{9,8}$ are event matrices, not single events like $F_{17,2}$. Similarly, $X_5$ through $X_8$ can be expanded to $X_4$ through $X_4$, and $X_4$ through $X_4$ can be expanded to $B_1$ and $B_2$. It is easy to understand that the final expanded sum-of-products of $E = X_{17,1}X_{18,1}X_{19,1}$ have $4^5 \times 4^6 \times 4^7 \times 4^8 = 17, 179, 869, 184$ items, without considering the members of event matrices.

In many cases, evidence variables are in the furthest position from root causes, so that the expanded sum-of-products for each evidence is large. When we multiply them together during expanding $E$, the combination explosion problem appears. Therefore, it is hard to get the value of $Pr\{E\}$ and $Pr\{B_{ij}\}$ directly.

**B. RECURSIVE ALGORITHM**

The recursive algorithm is proposed in [5] and [18] for the inference of cubic-DUCG [8]. A typical cubic-DUCG model is shown in Fig. 4, in which the variables in different time layers are treated as different nodes even for a same variable. All variables are state-known indicated in colors. The cubic-DUCG is generated based on the original DUCG according to the online received evidence.

The basic idea of the recursive algorithm is to calculate $Pr\{E\}$ layer by layer, which can reduce the computation greatly, given the precondition that all nodes are state-known.

The recursive algorithm can be summarized as follows.

**Definition 1:** Let $l$ be the layer index, where $l$ is the maximal number of arcs between the X-type nodes in layer $l$ and root cause $B_k$ in a DUCG.

For the example in Fig. 4, $X_1$ through $X_4$ are in layer 1 ($l = 1$), $X_5$ through $X_8$ are in layer 2 ($l = 2$), $X_7$ through $X_9$ are in layer 3 ($l = 3$), $X_{11}$ and $X_{12}$ are in layer 4 ($l = 4$), and $X_{13}$ is in layer 5 ($l = 5$).

Denote the evidence events (state-known X-type nodes) in layer $l$ as $E(l)$, we have $E = \bigcap_{l=1}^{\text{max}} E(l)$. When there is no evidence in layer $l$, e.g. layers 1–4 in Fig. 4, $E(l) = 1$ (complete set). Then we have (4).

$$Pr\{E\} = \sum_j Pr\{E|B_{ij}\}Pr\{B_{ij}\}$$

$$= \sum_j Pr\{\bigcap_{l=1}^{\text{max}} E(l)|B_{ij}\}Pr\{B_{ij}\}$$

$$= \sum_j Pr\{E(\text{max})|B_{ij}\} \bigcap_{l=1}^{\text{max}-1} E(l) \cdots$$

$$Pr\{E(2)|B_{ij}\}Pr\{E(1)|B_{ij}\}Pr\{B_{ij}\}$$

(4)

In (4), $Pr\{B_{ij}\} = b_{ij}$ is easy to know from statistics. To obtain the result of (4), the following assumption is applied.

**Assumption 8:** $E(l)$ is independent of $E(l-2)$ through $E(1)$ and $B_{ij}$ conditional on $E(l-1)$.

Under Assumption 8, (4) becomes (5).

$$Pr\{E\} = \sum_j Pr\{E(\text{max})|E(\text{max}-1)\}$$

$$Pr\{E(\text{max}-1)|E(l-2)\} \cdots$$

$$Pr\{E(2)|E(1)\}Pr\{E(1)|B_{ij}\}Pr\{B_{ij}\}$$

(5)

**Assumption 9:** All parents of $E(l)$ are included in $E(l-1)$. Under Assumption 9, (5) becomes (6), in which (1) and (2) are applied.

$$Pr\{E\} = \sum_j Pr\{\bigcap_{l=1}^{\text{max}} X_{nlk} \bigcap_{l=1}^{\text{max}-1} X_{ij}\}$$

$$Pr\{\bigcap_{l=1}^{\text{max}-1} X_{nlk} \bigcap_{l=1}^{\text{max}-2} X_{ij}\} \cdots$$

$$Pr\{\bigcap_{l=1}^{\text{max}} X_{nlk} \bigcap_{l=1}^{\text{max}-1} X_{ij}\}$$

$$Pr\{\bigcap_{l=1}^{\text{max}} X_{nlk} \bigcap_{l=1}^{\text{max}-1} X_{ij}\}$$

$$Pr\{\bigcap_{l=1}^{\text{max}} X_{nlk} \bigcap_{l=1}^{\text{max}-1} X_{ij}\}$$

(6)
In which \( f_{nk;i;j} = \frac{r_{nk;i}/r_{nk;j}}{r_{nk;i}/r_{nk;j}} \) the \( \{r, \alpha\} \)-type parameters are given during the DUCG construction.

Equation (6) provides an analytical algorithm for inference and is frequently used in DUCG-aided clinical diagnoses.

In some cases, Assumptions 8 and 9 are not satisfied. For the example in Fig. 5, \( X_6 \) is the parent of \( E(5) = X_{13} \) but is not included in \( E(4) \). In such cases, we have

\[
Pr\{E\} = \sum_j Pr\{\bigcap_{X_{nk}\in E(\text{max})} V_{ij} \} \cdot Pr\{\bigcap_{X_{nk}\in B}\bigcap_{V_{ij}\in AEB(l_{\text{max})}} V_{ij} \} \cdot Pr\{\bigcap_{X_{nk}\in E(2)} V_{ij} \} \cdot Pr\{\bigcap_{X_{nk}\in E(1)} V_{ij} \} = \sum_j \left( \prod_{X_{nk}\in E(\text{max})} \sum_{V_{ij}\in AEB(l_{\text{max})}} f_{nk;i;j} \right) \cdot Pr\{\bigcap_{X_{nk}\in E(1)} V_{ij} \} \cdot Pr\{\bigcap_{X_{nk}\in B} V_{ij} \} \cdot Pr\{\bigcap_{X_{nk}\in E(2)} V_{ij} \} \cdot Pr\{\bigcap_{X_{nk}\in E(1)} V_{ij} \} \cdot \prod_{X_{nk}\in E(1)} \sum_{V_{ij}\in AEB(l_{\text{max})}} f_{nk;i;j} \cdot \prod_{X_{nk}\in E(1)} b_{kj} (7)
\]

In which, \( AEB(l) \) indicates the directly linked ancestor evidence/B-type event of the \( X \)-type event in layer 1. Equation (8) is used to calculate the cubic-DUCG presented in [6] and [25]. Fig. 4 is an example.

However, in some cases such as Fig. 4, Assumptions 8 and 9 are not satisfied and Eqn (7) is not applicable. We can only apply (4), which involves expanding the evidence nodes to the upper stream layer evidence nodes or B-type nodes through intermediate state-unknown nodes such as \( X_1 \) through \( X_16 \) in Fig. 3, where the recursive algorithm cannot be applied directly.

III. STOCHASTIC SIMULATION ALGORITHM FOR DUCG

A. CHALLENGES FOR DIRECT SAMPLING ALGORITHM

To solve the above problem, we apply the Stochastic simulation method in this section. As mentioned in (4), the DUCG inference can be transformed as calculating \( Pr\{E|B_{kj}\} \). Then, the most intuitive way is to expand evidence \( E \) to \( B_{kj} \) to get the distribution of \( Pr\{W_E|B_{kj}\} \) and sample on it directly, where \( W_E \) is the set of evidence variables. However, its time complexity will be too large.

Therefore, the time consumption for calculating \( Pr\{W_E|B\} \) has no advantage over the analytical algorithm in section 2. What’s worse, the value of \( Pr\{E|B_{2,1}\} \) in DUCG may be too little that the count of sampling cycles is too large, e.g. \( Pr\{E|B_{2,1}\} \) in Fig. 5 is 5.626 \( \times 10^{-14} \). In the sampling algorithm of BN, the cycle required is \( N = (4/\phi \epsilon^2)\ln(2/\Delta) \) [19] or \( N = (7/\epsilon^{\mu})\ln(4/\Delta) \) [20], where \( \phi \) is the exact result, \( \mu \) is the sampling result, \( \epsilon \) is the error ratio expected, and \( 1 - \Delta \) is confidence coefficient. When we apply these derivations to DUCG, the algorithm will require more than \( 10^{18} \) cycles given \( \epsilon = 1\% \), which is incalculable for modern hardware.

Therefore, a new algorithm based on the conditional sampling algorithm is proposed below, it samples the model layer by layer similar to the recursive algorithm. Because the sample result of each layer conditioned on a higher layer has a higher order of magnitude, the requirement for sampling cycles will be much less.

B. SAMPLING ALGORITHM FOR SINGLE EVIDENCE CASE

At first, the simplest case with only one evidence variable is discussed. In this situation, the combination explosion problem of inference is caused by the complex structure of the graph. Fig. 6 is a typical \( n \times n \) full-joined example, it is presented to show this problem.

For each path that expands \( X_{17,1} \) to \( B_{1,1} \), e.g. \( X_{17,1} \leftarrow X_{13} \leftarrow X_0 \leftarrow X_5 \leftarrow X_1 \leftarrow B_{1,1} \), \( n + 1 \) arcs are referred during calculation, e.g. to calculate \( A_{17,13}A_{13,9}A_{9,5}A_{5,1}A_{1,1} \). Suppose that every node has \( k \) states, then the dimension of all \( F \)-type matrices is \( k \times k \). Because the time complexity of multiplication between two \( k \times k \) matrices is \( k^3 \), the time complexity for expanding each path is \( O(k^3(n + 1)) = O(k^3n) \). In addition, because there are \( n^2 \) paths for the expansion, the computation complexity is \( O(k^3n^{n+1}) \). Therefore,
it is impossible to infer in such cases with the analytical algorithm.

The conditional sampling algorithm is proposed as shown below:

1) Denote the set of state-unknown variables as \( W_u \), and set all variables \( X_n \in W_u \) with a random initial state as \( X_{n,\text{ini}}(0) \), e.g., state 0;
2) Calculate the conditional probability distribution of \( X_n \) as shown in (8), then assign \( X_n \) with a new state \( S_n(t) \) which is sampled based on states of other nodes during the cycle.

\[
P_{\text{M}(t)} = \frac{1}{N} \sum_{t=1}^{N} P(t)
\]

5) Repeat step 3 to 5 until the convergence of \( |P_{\text{M}(t)}/P_{\text{M}(t-1)} - 1| \leq \epsilon \), where \( \epsilon \) is an error value that quantifies the computation accuracy.

6) Output the value of \( P_{\text{M}(t)} \) as the sampling result of \( P[E|B_{\text{ij}}] \)

Similarly, denote \( W_{pE} \) as the parent set of \( E \). For the cases with only one evidence, the (12) will have a similar form as (11).

\[
P(t) = Pr[E|B_{\text{ij}} \cap X_{n,S_n(t)}] = Pr[E|V_{i,S_i(t)}]
\]

**FIGURE 7. Basic example for the sampling algorithm.**

Fig.7 presents a basic example to verify this algorithm. Denote \( B_{1,1} \) as the root cause, whereas \( X_3, \ldots, X_5 \) are parents of evidence. According to (12):

\[
P(t) = Pr[X_E|X_{2,5},X_{2,3},X_3(\ldots)]
\]

Then:

\[
P_{\text{M}(t)} = \frac{1}{N} \sum_{t=1}^{N} Pr[X_E | X_{2,5},X_{2,3},X_3(\ldots)]
\]

Because \( X_{2,5}, X_{2,3} \) are sampled directly conditioned on \( B_{1,1} \), the frequencies of the event \( X_n = S \) will satisfy \( 1/N \cdot \text{freq}(X_n = S) = Pr[X_{n,S_n}|B_{1,1}] \), where \( S_n \) is a certain state of \( X_n \), then \( P_{\text{M}(t)} \) can be reformed as (13).

\[
P_{\text{M}(t)} = \sum_{X_{2,5}, X_{2,3}} Pr[X_E|X_{2,5},X_{2,3}] = \sum_{X_{2,5}, X_{2,3}} \frac{1}{N} \cdot \text{freq}(X_{2,5},X_{2,3})
\]

Meanwhile, the inference result of the analytical algorithm is (14), which is equal to (13).

\[
Pr[X_E|B_{1,1}] = Pr[X_E|2F_{2,1} + X_E|3F_{3,1}]
\]
Evidence $X_{E,1}$ is state-known, say state 1. The expanding of its children will only refer to its known state 1 according to the recursive algorithm. In other words, evidence nodes do not require state updating during sampling cycles.

If $X_{E,1}$ is replaced by a state-unknown variable $X_l$, frequency of $X_l = S_l$ during cycles will also satisfy $1/N \cdot \text{freq}(X_l = S_l) = Pr[X_{l|S_l}]$ according to (13).

If we apply this derivation from root cause to leaf node layer by layer in the graph, it can be figured out that the frequency of every single node $X_{n}$’s sampling state satisfies (15).

$$\frac{1}{N} \text{freq}(X_n = S_n) = Pr[X_{n,S_n}|B_{1,1}]$$

**C. STOPPING CRITERION AND TIME COMPLEXITY**

In the basic conditional algorithm, the stopping criterion is $|P_M(t)/P_M(t-1) - 1| \leq \epsilon$. It is simple but too rough. A more accurate method is discussed in this section.

As mentioned in [21], it is difficult to determine the convergence of the MCMC algorithm theoretically, therefore, the stopping criterion is based on estimations. $\epsilon - \delta$ estimate is a common tool for the evaluation of sampling algorithm, the sampling cycles stop after meeting the condition:

$$P \{ |(1 - \epsilon) < \mu < (1 + \epsilon) | \geq 1 - \Delta \epsilon \}, \quad \Delta < 1$$

In this expression, $\mu$ is the sampling result, $\phi$ is the analytical result, $\epsilon$ is the error ratio expected for the algorithm, and $1 - \Delta$ is confidence coefficient.

An estimation based on the law of large numbers and the central limit theorem is proposed. After sufficient sampling cycles, $\mu$ will fit the normal distribution i.e. $\mu \sim N(\mu, \delta^2/n)$, where $n$ indexes the count of cycles, where $\mu$ is the value of $P_M(t)$ and $\delta$ is its variance.

Denote the burn-in value as $b$ which means ignoring first $b$ cycles, and denote the window-width value as $\omega$ which means $\mu$ and $\delta$ are the mean value and variance of last $\omega$ cycles. Then the halting problem algorithm is:

1) Set an upper limit of the sampling cycle as $\text{Cycle}_{\text{max}}$ in case that the sampling cannot converge;
2) Execute the basic sampling algorithm for $b + \omega$ cycles;
3) Execute the sampling algorithm for one cycle, record the current cycle as the cycle $t$;
4) Calculate the means of $P_M(t)$ in last $\omega$ sampling cycles as:

$$\mu_t = \frac{1}{\omega} \sum_{j=t-\omega}^{t} P_M(j)$$

5) Calculate the standard deviation of $P_t$ in last $\omega$ sampling cycles as:

$$\delta_t = \left[ \frac{1}{\omega} \sum_{j=t-\omega}^{t} (P_M(j) - \mu_t)^2 \right]^{\frac{1}{2}}$$

6) Repeat step 3 to 5 until $\mu_t$ and $\delta_t$ satisfy:

$$2\int_{(1+\epsilon)\mu_t}^{\infty} N(\mu_t, \delta_t^2)dx < \Delta$$

7) Output the value of $\mu_t$ as sampling result and $N = t$

Then, the sampling algorithm takes around $N > b + \omega + (c(\delta/\epsilon \mu)^2$ cycles. $c = Q^{-1}(\Delta/2)$, where $Q$ is the right tail function of the normal distribution; the relationship between $c$ and $\Delta$ value is presented in Table 4.

As $b + \omega \ll N$, and $\delta$, $\mu$ have similar orders of magnitude in most cases, the value of $N$ is not sensitive to either scale of graph or value of $\phi$, so $N \gg (c(\delta/\epsilon \mu)^2$ can be used as a proper halting rule for the algorithm. Because the diagnoses focus on the sort of root causes’ probability instead of the exact numerical value, about a 5% error ratio will be enough for practical use. According to the law of large numbers, the practical errors of the algorithm can be larger than $\epsilon$ when $N$ is not large enough. Therefore, $\epsilon$ is assigned less in practical for accuracy demanded.

As the sampling process during each cycle needs to be repeated $N$ times, we can get the performance of the algorithm.

For simplicity, the example in Fig.6 is used for derivation. As the expanding process required for updating a single node $X_l$ only refers to its parents, the time complexity for updating $X_l$ is $kn$, where $k$ is count of $X_l$’s states. Because all of the $n^2$ state-unknown variables need to be updated, the time complexity is $O(kn^2)$ during one sampling cycle. Then the time complexity with the new stopping criteria is (16), which means the algorithm manages to reduce the time complexity from exponential to polynomial.

$$O \left( kn^3 \left( \frac{c\delta}{\epsilon \mu} \right)^2 \right)$$

In section 4-A, tests are presented to show the performance between analytical and sampling algorithms on the example in Fig.6. In this case, the sampling algorithm is 24 times faster than the analytical one when $n = 8$.

**D. MULTI EVIDENCE SITUATION AND ABSORING PROBLEM**

Another combination explosion problem is caused by the absorbing process for the logic operation during expansion. Denote the expression after one step expanding of each evidence $X_{nj} \in E(l)$ as $Exp_{nj}$, then these $Exp_{nj}$ will be multiplied/AND together and the absorbing process to simplify the full expression. We cannot just multiply the result in section 3-B of each $X_{nj} \in E$ together directly when different evidence variables have the same parent nodes. In addition, because each $Exp_{nj}$ is the sum/OR of $X_n$’s parents according to (1), this multiplication/AND operation is also

**TABLE 4. Value of $\Delta$ and $c$.**

| $\Delta$ | $c$ | $\epsilon$ |
|------|-----|-----|
| 31.7% | 1   | 4.6% | 2   |
| 0.3%  | 3   |     |     |
an exponential problem. Fig. 8 shows an example for this absorption process. $E(3) = X_{7,1}X_{8,1}X_{9,1}$ is reformed as (17) after expanding to layer 2.

$$E = X_{7,1} \cdot X_{8,1} \cdot X_{9,1}$$

$$= \text{Exp}_{7,1} \cdot \text{Exp}_{8,1} \cdot \text{Exp}_{9,1}$$

$$= F_{9,1;4} \cdot F_{8,1;4} \cdot F_{7,1;4}X_4 + F_{7,1;5} \cdot F_{8,1;5} \cdot F_{9,1;5}X_5$$

$$F_{7,1;6} \cdot F_{8,1;6} \cdot F_{9,1;6}X_6 + (F_{7,1;5} \cdot F_{9,1;4} \cdot F_{8,1;4})$$

$$+ F_{7,1;6} \cdot F_{8,1;6} \cdot F_{9,1;6} + F_{7,1;4} \cdot F_{8,1;4} \cdot F_{9,1;5}X_5$$

$$+ F_{7,1;5} \cdot F_{8,1;5} \cdot F_{9,1;5}X_5 + F_{7,1;4} \cdot F_{8,1;4} \cdot F_{9,1;5}X_5$$

Therefore, new methods are required for multi-evidence variables situations.

An initiative way is to sample on the joint conditional probability distribution of each layer instead of a single variable $X_n$ in step 2. In other words, use (18) to replace (11) in Step 3, where $W(l)$ means the set of variables in layer $l$ and $S(l)$ is the joint states of them in cycle $t$.

$$P_r[W(l)_{S(l)}|W(l-1)_{S(l-1)}]$$

This term can be expanded as (23).

$$Z_{45} \cdot X_4X_5 = Z_{45}[F_{4,2}F_{5,2}X_2 + F_{4,3}F_{5,3}X_3$$

$$+(F_{4,2}F_{5,3} + F_{4,3}F_{5,2})X_2X_3]$$

As all terms in expanding expression have the form $F[l]X$, the definition of $n$-order $F$ is given.

Definition 3: For a term $F[l]X$ in the expanding expression, if the count of $F$-type events in products $F[l]$ is $n$, this term is called $n$-order $F$.

The expanding process in (20) ends after two steps, meanwhile, more complex terms will need more expanding steps. The last term with $X_4X_5X_6$ in (17) is presented as an example. Similarly, $Z_{456}$ is defined for simplicity:

$$Z_{456} = +F_{7,1;5} \cdot F_{8,1;6} \cdot F_{9,1;4} + F_{7,1;6} \cdot F_{9,1;5} \cdot F_{8,1;4}$$

$$+ F_{7,1;6} \cdot F_{9,1;4} \cdot F_{8,1;6} \cdot F_{9,1;5}X_4X_5X_6$$

The term can be expanded expression as (23).

$$Z_{456} \cdot X_4X_5X_6 = Z_{456}[F_{4,2}F_{5,2}X_2 + F_{4,3}F_{5,3}F_{6,3}X_3$$

$$+(F_{4,2}F_{5,3}F_{6,3} + F_{4,3}F_{5,2}F_{6,3} + F_{4,3}F_{5,3}F_{6,2})X_2X_3]$$

The term in (20) with $X_2$ is $Z_{45}F_{4,2}F_{5,2}X_2$, whereas the similar one in (23) is $Z_{456}F_{4,2}F_{5,2}F_{6,2}X_2$. Both of $S_{45}$ and $S_{456}$ are sums of three 3-order $F$ terms, therefore, $Z_{45}F_{4,2}F_{5,2}X_2$ is 5-order $F$ and $Z_{456}F_{4,2}F_{5,2}F_{6,2}X_2$ is 6-order $F$. Because (20) is expanded from $Z_{45}X_4X_5$ and (23) is expanded from $Z_{456}X_4X_5X_6$, it can be concluded that the more $X$ variables a term contain, the higher-order $F$ its sub-terms will be after expanding.

After that, the 5-order $F$ term becomes 7-order $F$ after it is expanded from $Z_{45}(F_{4,2}F_{5,3} + F_{4,3}F_{5,2})X_2X_3$ to $Z_{45}(F_{4,2}F_{5,3} + F_{4,3}F_{5,2})F_{2,1;1}F_{3,1;1}B_{1,1;1}$. Therefore, the order

$$+ F_{7,1;5} \cdot F_{9,1;6} \cdot F_{8,1;4} + F_{7,1;6} \cdot F_{9,1;5} \cdot F_{8,1;4} + F_{7,1;4} \cdot F_{8,1;5} \cdot F_{9,1;4}$$

$$(19)$$
$F$ of a term will also increase when further expanding steps are applied.

$X_i$ terms will be replaced with the event $X_{n,S_i(t)}$ after expanding, so terms will be reformed as $\prod f$. Multiply operation for $f$ events follows the AND/multiplication matrix operator specially defined in Corollary 15 in [2], in brief, elements in the result matrix can be represented as sums-of-products of $a$ parameters’ product, i.e. $\sum \prod a_{ij}$ for a $n$-order $F$ term.

As $a_i$ parameters are elements in the relation matrices, they are all less than one. Therefore, the higher $n$-order $F$ a term is, the less value it is. As a result, large $n$-order $F$ terms can be ignored during the calculation in (9).

Two hyperparameters are denoted for this estimation. $IG_x$ defines the threshold for the count of $X-$ type variable, and $IG_{layer}$ defines the threshold for the count of layers to expand from variable expansions. All terms beyond the threshold will be ignored during calculation. The example in Fig.8 is tested in section 4-B, and the error ratio of all tests is less than 1%.

The example in Fig.8 is extended as Fig.9 to show the time complexity of the cut-off estimation method.

The expanding expression for $E = X_{3n-2}X_{3n-1}X_{3n-1}$ is similar to (17), it has 27 terms: 3 one-$X$ terms, 18 two-$X$ terms, and 6 three-$X$ terms. One-$X$ term means this term contains only one $X$-type variable, e.g., $F_{3n-5}X_{3n-8}$.

In the analytical algorithm, the whole expression needs to be expanded to the above layers. In one step of expanding, a single one-$X$ term will be expanded to 3 one-$X$ terms, for example:

$$X_{3n-5} = F_{3n-5}X_{3n-8} + F_{3n-5}X_{3n-7} + F_{3n-5}X_{3n-6}$$

The two $X$ terms will be reformed as an expression with 9 terms after one step expanding, and it consists of 3 one $X$ terms and 6 two $X$ terms, e.g.,

$$X_{3n-3}X_{3n-4} = (F_{3n-3}X_{3n-8} + F_{3n-3}X_{3n-7} + F_{3n-3}X_{3n-6}) \cdot (F_{3n-4}X_{3n-8} + F_{3n-4}X_{3n-7} + F_{3n-4}X_{3n-6})$$

$$= (F_{3n-3}X_{3n-8} + F_{3n-3}X_{3n-7} + F_{3n-3}X_{3n-6}) \cdot (F_{3n-4}X_{3n-8} + F_{3n-4}X_{3n-7} + F_{3n-4}X_{3n-6})$$

The expanding expression for three terms is also similar to (17), therefore it has 27 terms, and 3 of them are one-$X$ terms, 18 terms are two-$X$ terms and 6 terms are three-$X$ terms.

For convenience, we denote the count of one-$X$ terms in $i_{th}$ step expanding expression as $a_i$, count of two-$X$ terms as $b_i$ and three-$X$ terms as $c_i$. So $a_0 = 0$, $b_0 = 0$, $c_0 = 1$, and $a_i + 1 = 3(a_i + b_i + c_i)$, $b_i + 1 = 6b_i + 18c_i$, $c_i + 1 = 6c_i$. With these conditions, it can be figured that:

$$c_n = 6^n$$

$$b_n = 6b_{n-1} + 18c_{n-1} = 3(n - 1)6^n$$

$$a_n = 3(a_{n-1} + b_{n-1} + c_{n-1})$$

$$= 3^n - 1 + \sum_{i=1}^{n} 3^{n-i}(3n - 2)2^{i-1}3^{i-1}$$

$$= (3n - 2)6^n - 7 \times 3^n$$

Therefore, the expression will have $a_n + b_n + c_n = 2(3n - 2)6^n - 7 \times 3^n$ terms when $E$ is expanded to $X_1X_2X_3$.

The terms with the largest $F$-order are the expanding result of three $X$ terms in every layer, and they contain $3nF$ events. Meanwhile, the one-$X$ term will be $n$-order $F$ events. Therefore, the time complexity for calculating the value of each term is $O(n)$. So, the time complexity of the analytical algorithm is:

$$O(n[(3n - 2)6^n - 7 \times 3^n])$$

For simplify:

$$O(n^2e^n)$$

The situation is simpler for the sampling algorithm with a cutting-off algorithm. As all one-$X$ terms can be substituted by numerical value in each step of expanding process, the number of one-$X$ term changed to $a'_i = 3(b_i + 1 + c_i)$ in $i_{th}$ expanding step, whereas $b_i$, $c_i$ staying the same as the analytical method. Because the expanding stops after $IG_{layer}$ steps, the count of terms in the sampling algorithm is:

$$b_{IG_{layer}} + c_{IG_{layer}} + \sum_{i=1}^{IG_{layer}-1} 3(b_i + c_i)$$

$$= (3IG_{layer} - 2)e^{IG_{layer}} + 3 \sum_{i=1}^{IG_{layer}-1} (3i - 2)6^i$$

$$= (4.2IG_{layer} - 3.44)e^{IG_{layer}} + 0.24$$

(28)

Like the derivation shown before, terms are $3IG_{layer}$-order $F$ or $2IG_{layer}$-order $F$ after the expanding, so the time complexity is:

$$O(IG_{layer}(4.2IG_{layer} - 3.44)e^{IG_{layer}} + 0.24IG_{layer})$$
For simplify:

\[ O(I_{\text{layer}}^2, e^{\text{Trj}}) \]

Although the complexity of the sampling algorithm is still exponential, \( I_{\text{layer}} \) is less than \( n \) so the time consumption will be less. For instance, if \( I_{\text{layer}} = 1/2n \) and \( n \geq 6 \), the time consumption full expanding will be 1500 times larger than cut-off method. Therefore, a sampling process ending within 1500 cycles will be faster than the analytical method. The effect of \( I_{\text{layer}} \) is similar to \( I_{\text{layer}} \).

An example based on a real case is tested in section 4-C, where the sampling method is 3 times faster than the analytical algorithm.

IV. APPLICATION OF DUCG SAMPLING ALGORITHM

Three examples of the DUCG model are presented in this section. The first one is a set of \( n \)-size full-joined models in Fig.6 with randomly assigned parameters, it shows the accuracy and performance for the basic algorithm shown in section 3-B, the influence of hyperparameters \( \omega, b \) are also tested. The second one is the multiple evidences model illustrated in Fig.8, which verifies the estimate method of the method proposed in section 3-D. The last one is adapted from a model for Viral Hepatitis B deployed in hospitals, and it shows the performance of the new algorithm in the real application. Note that all examples have been simplified and contain only one root cause.

All experiments except the one in section 4-C are tested with a python-based demo. In section 4-C, the tests for the numerical algorithm are tested on a java-based system that is designed for a production environment. Meanwhile, the sampling algorithm is tested on the python demo, but it is still much faster than the numerical algorithm. All of the experiments are tested on the same 8-core server.

A. FULL JOINED \( n \times n \) MODEL

To provide an intuitive presentation to the performance advantages over analytical algorithm in expanding process, several full joined models with \( n \times n \) variables are tested in this section, where \( n \) grows from 2 to 10.

All directed-arcs are assigned with random relation matrix \( a \) with the constraint that the sum of each column is 1, then all causal relations of \( F \)-type events are assigned as \( r_{i:j} = 1 \).

For instance, the \( 2 \times 2 \) example in Fig.10 is presented to show the initialization of data.

![FIGURE 10. 2 × 2 full joined model.](image-url)

As it is shown in the model, the root cause is \( B_{0,1} \) and the evidence is \( E = X_{5,1} \). The relation matrixes are randomly generated:

\[ r_{1:0} = 1.00, \quad r_{2:0} = 1.00 \]

### TABLE 5. Accuracy of the sampling algorithm.

| n  | Loop count | Analytical result | Sampling result | Error ratio |
|----|------------|-------------------|-----------------|-------------|
| 2  | 2037       | 0.365             | 0.364           | 0.176%      |
| 3  | 1417       | 0.231             | 0.232           | 0.327%      |
| 4  | 1131       | 0.152             | 0.152           | -0.061%     |
| 5  | 1010       | 0.191             | 0.191           | 0.175%      |
| 6  | 1747       | 0.119             | 0.119           | -0.397%     |
| 7  | 1856       | 0.082             | 0.082           | 0.823%      |
| 8  | 1385       | 0.101             | 0.102           | 0.685%      |
| 9  | 1541       | -                 | 0.061           | 0.000%      |
| 10 | 1678       | -                 | 0.062           | 0.000%      |

The parameters in sampling process are assigned as burn-in loops \( b = 300 \), window width \( \omega = 200 \), \( \epsilon = 10^{-3} \) and \( \Delta \approx 5\% (C = 2) \). Because this example has only one evidence variable, the cutting-off scheme for the absorbing process is not required. Therefore, the analytical algorithm-based program is also simplified by removing the absorbing logic in this test.

With all mentioned above, full-joined models with \( n = 2, 3, 4 \ldots 10 \) are tested by both analytical-numerical and stochastic sampling algorithms for comparison.

Because the time required for the analytical method is too large when \( n > 8 \), the \( n = 9, 10 \) cases are only tested in the sampling algorithm.
Fig. 11 shows the time required for the two algorithms. Analytical method spends less time than sampling one when $n < 6$, then its time consumption raises rapidly when $n \geq 7$, which shows a trend of typical combinatorial explosion. In comparison, the increment of time required by the sampling method is flat.

Approximate curves for analytical algorithm present as an exponential function, whereas the curve for sampling one displays as a polynomial function, which is consistent with the derivation of complexity in previous sections.

Fig. 12 shows the convergence process of the 9 cases. The situation is similar to Fig. 14: after the peak at the beginning, the curves shocks slightly in a tiny interval around the final result. More detailed data about loop counts and the error ratio is listed in the 5. All tests end with an error of less than 1%, which verifies the accuracy of the sampling algorithm as an approximate method.

Note that loop counts for all examples are about 1000 to 2000 regardless of the $n$ value, which verifies the discussion about loop counts in section 3-C. Moreover, as the path from root cause to the leaf nodes are denoted by domain engineers in practical use, most of the models are less than 10 layers, so that thousands of loops will be enough for practical uses.

We test the $6 \times 6$ full-joined model with different $\omega, b$ as $b = 100, 200 \sim 1000$ and $\omega = 100, 200 \sim 1000$. The result of the absolute error ratio is shown in Fig. 14(a), and all error ratios less than 1% and the result shows a negative relationship to these two parameters. Meanwhile, the count of loops required is shown in Fig. 14(b), and all 100 tests end within 4000 loops, it also shows that the time complexity will increase a lot when $b$ is small and $\omega$ is large.

This result shows the robustness among different hyper-parameters.

B. A COMPACT EXAMPLE
The example in Fig. 8 is tested for verification of the accuracy of the cut-off estimation.

To simplify, causal relations $r_{ij}$ for all of the $F$-type events are assigned 1, whereas the relation matrices $a_{ij}$ are assigned
as follows, they are generated randomly and then normalized:

\[
\begin{align*}
  r_{2;1} &= 1.00, \quad r_{3;1} = 1.00 \\
  a_{2;1} &= \begin{pmatrix} 0.1890 & 0.2490 \\ 0.3440 & 0.4200 \\ 0.4670 & 0.3310 \end{pmatrix}, \\
  a_{3;1} &= \begin{pmatrix} 0.7850 & 0.6390 \\ 0.2150 & 0.3610 \end{pmatrix} \\
  r_{4;2} &= 1.00, \quad r_{4;3} = 1.00 \\
  a_{4;2} &= \begin{pmatrix} 0.9080 & 0.7730 & 0.4440 \\ 0.0920 & 0.2270 & 0.5560 \end{pmatrix}, \\
  a_{4;3} &= \begin{pmatrix} 0.5970 & 0.1770 \\ 0.4030 & 0.8230 \end{pmatrix} \\
  r_{5;2} &= 1.00, \quad r_{5;3} = 1.00 \\
  a_{5;2} &= \begin{pmatrix} 0.1810 & 0.2030 & 0.5180 \\ 0.8190 & 0.7970 & 0.4820 \end{pmatrix}, \\
  a_{5;3} &= \begin{pmatrix} 0.0910 & 0.2110 \\ 0.9090 & 0.7890 \end{pmatrix} \\
  r_{6;2} &= 1.00, \quad r_{6;3} = 1.00 \\
  a_{6;2} &= \begin{pmatrix} 0.5640 & 0.2390 & 0.5600 \\ 0.4360 & 0.7610 & 0.4400 \end{pmatrix}, \\
  a_{6;3} &= \begin{pmatrix} 0.4760 & 0.6420 \\ 0.5240 & 0.3580 \end{pmatrix} \\
  r_{7;4} &= 1.00, \quad r_{7;5} = 1.00, \quad r_{7;6} = 1.00 \\
  a_{7;4} &= \begin{pmatrix} 0.0100 & 0.3030 \\ 0.9900 & 0.6970 \end{pmatrix}, \\
  a_{7;5} &= \begin{pmatrix} 0.4660 & 0.9520 \\ 0.5340 & 0.0480 \end{pmatrix}, \\
  a_{7;6} &= \begin{pmatrix} 0.4990 & 0.7070 \\ 0.5010 & 0.2930 \end{pmatrix} \\
  r_{8;4} &= 1.00, \quad r_{8;5} = 1.00, \quad r_{8;6} = 1.00 \\
  a_{8;4} &= \begin{pmatrix} 0.5170 & 0.4750 \\ 0.4830 & 0.5250 \end{pmatrix}, \\
  a_{8;5} &= \begin{pmatrix} 0.7490 & 0.1190 \\ 0.2510 & 0.8810 \end{pmatrix}, \\
  a_{8;6} &= \begin{pmatrix} 0.5020 & 0.5100 \\ 0.4980 & 0.4900 \end{pmatrix} \\
  r_{9;4} &= 1.00, \quad r_{9;5} = 1.00, \quad r_{9;6} = 1.00 \\
  a_{9;4} &= \begin{pmatrix} 0.4300 & 0.4480 \\ 0.1430 & 0.0040 \\ 0.4270 & 0.5480 \end{pmatrix}, \\
  a_{9;5} &= \begin{pmatrix} 0.3570 & 0.1530 \\ 0.4880 & 0.4430 \\ 0.1540 & 0.0400 \end{pmatrix} \\
  a_{9;6} &= \begin{pmatrix} 0.5260 & 0.4240 \\ 0.2360 & 0.4750 \\ 0.2380 & 0.1010 \end{pmatrix}
\end{align*}
\]

In the analytical algorithm-based system, the exact value of \( Pr\{E|B_{1,1}\} \) is \( 7.939915 \times 10^{-2} \), and it cost 71ms.

### Table 6. Sampling result for example in Fig.8.

| Attempts | Loop count | Time consumed | Exact result | Sampling result | Error Ratio |
|----------|------------|---------------|-------------|----------------|-------------|
| 1st      | 1282       | 0.375S        | 7.940E-02   | 7.946E-02      | 0.07%       |
| 2nd      | 1217       | 0.363S        | 7.940E-02   | 8.014E-02      | 0.93%       |
| 3rd      | 1876       | 0.550S        | 7.940E-02   | 7.931E-02      | -0.11%      |
| 4th      | 1967       | 0.572S        | 7.940E-02   | 7.908E-02      | -0.41%      |
| 5th      | 1105       | 0.319S        | 7.940E-02   | 7.897E-02      | -0.54%      |
| 6th      | 1536       | 0.458S        | 7.940E-02   | 7.973E-02      | 0.42%       |

Meanwhile, assign the burn-in steps value as \( b = 300 \), window width as \( \omega = 200 \), and confidence 95% (\( \Delta = 5\% \) and \( C = 2 \)). Then set the layer threshold \( IG_{layer} = 2 \) and \( X \) threshold \( IG_{x} = 6 \). Repeat the algorithm 6 times, then the sampling result is presented in Table 6, and the convergence process is shown in Fig.14.

As it is shown in Table 6, all of the 6 tests converge within 2000 loops, which verifies the stability of the algorithm’s halting rule. In addition, all of the 6 tests end with an error ratio within \( -0.6\% \sim 1\% \), and the fluctuation is slight after the convergence.

Because the scale of the graph is small, the sampling method cost more time than the analytical one, but this example managed to verify the accuracy of the new sampling algorithm.

### C. AN EXAMPLE FOR VIRAL HEPATITIS B

An example adapted from the practical model for viral hepatitis B is presented for further verification. For more accuracy, the result of the analytical algorithm is calculated by a production edition program deployed in hospitals, instead of the experimental one used in previous examples.

This example is adapted from the graph built in [22]. As it is shown in Fig.11, the model consists of 49 nodes and 109 directed arcs. The structure of this example is simpler than the full-joined model, but the relationship between evidence nodes is more complex.

Definitions of nodes are listed in Table 7. The evidences are \( E = X_{65,1}X_{66,1}X_{72,1}X_{81,1}X_{82,1} \). A part of \( a \)-type matrices encoded in \( F \)-type variables are listed below (All data can be obtained from authors by request), some of these parameters are assigned by doctors, and normalization is executed to fill the missing parameters. Most causal relations \( r_{ij} \) are 1, due to a rapid and flexible construction demanded by doctors.

\[
\begin{align*}
  r_{3013;2} &= 1.00, \quad r_{1012;2} = 1.00, \quad r_{108;2} = 1.00 \\
  a_{3013;2} &= \begin{pmatrix} 0.8250 & 0.2350 \\ 0.1750 & 0.7650 \end{pmatrix}, \\
  a_{1012;2} &= \begin{pmatrix} 0.6970 & 0.8660 \\ 0.3030 & 0.1340 \end{pmatrix} \\
  r_{81;1020} &= 1.00, \quad r_{81;1014} = 1.00 \\
  a_{81;1020} &= \begin{pmatrix} 0.3990 & 0.6870 \\ 0.6010 & 0.3130 \end{pmatrix}, \\
  a_{81;1014} &= \begin{pmatrix} 0.6450 & 0.3840 \\ 0.3550 & 0.6160 \end{pmatrix}
\end{align*}
\]
TABLE 7. Definition of nodes in Fig. 15.

| Node Id | Node Definition | Node Definition                  |
|---------|-----------------|----------------------------------|
| B1      | Viral hepatitis B | $X_{100}$                  |
| X7      | Fever            | $X_{104}$                        |
| X8      | Jaundice         | $X_{20}$                         |
| X14     | Diarrhea         | $X_{106}$                        |
| X15     | Bloating         | $X_{107}$                        |
| X16     | Darker urine     | $X_{108}$                        |
| X105    | ARDS             | $X_{109}$                        |
| X42     | Urticaria        | $X_{94}$                         |
| X15     | Constipation     | $X_{82}$                         |
| X24     | Fatigue          | $X_{100}$                        |
| X65     | Headache         | $X_{1004}$                       |
| X66     | Dizziness        | $X_{1018}$                       |
| X2        | Trembling        | $X_{1008}$                       |
| X1007    | Ascites          | $X_{1012}$                       |
| X1001    | Large liver       | $X_{1013}$                       |
| X95      | Joint pain       | $X_{102}$                        |
| X35      | Liver cirrhosis  | $X_{1020}$                       |
| X97      | Vasculitis        | $X_{1030}$                       |
| X101     | Irritability     | $X_{1031}$                       |
| X98      | glomerulus nephritis | $X_{1032}$                  |
| X99      | Acute liver failure | $X_{100}$                  |
| X5019    | Anti-HBc-IgG     | $X_{1014}$                       |
| X91      | Shortness of breath after activity | $X_{1035}$                  |
| X5013    | Laboratory test for viral hepatitis B | $X_{1034}$                  |
| X5006    | Mildly elevated alpha-fetoprotein | $X_{1036}$                  |

The result of the analytical method is $2.8773 \times 10^{-2}$ and its time consumption is 105.9s. Because the relationships between evidence variables are complex, the sampling parameters are adapted for better performance. The burn-in step is assigned $b = 200$, windows width is $\omega = 100$, $\epsilon = 3 \times 10^{-3}$, $\Delta = 5\%$ (C = 2), whereas IG layer = 2 and IG len = 5. The test is repeated 5 times for a stable assessment. The results are listed in Table V and the convergence path is illustrated in Fig. 16.

All tests converge within 800 cycles, and the final results are in the interval $2.8046 \times 10^{-2}$ to $2.9550 \times 10^{-2}$. The average time consumption of 5 runnings is 29.8s.

Despite the error ratio increases due to the change of parameters, the sampling algorithm costs only 1/3 time of the analytical one, and the error ratio is less than 3% which meets the accuracy demanded.

Finally, this new sampling algorithm provides a highly efficient and accurate method for inference on large-scale DUCG.

V. CONCLUSION AND FUTURE WORK

In this paper, the stochastic simulation algorithm for DUCG probabilistic reasoning is proposed to solve the state combination explosion problems, which is based on the early presented recursive algorithm so that it is applicable for the practical case with a small computation value. Two typical
examples are used to analyze the efficiency to reduce the computation time. For the expanding problem, the time complexity is reduced from $O(n^{n+1})$ to $O(n^3(C_b/\epsilon \mu)^2)$, in which $(C_b/\epsilon \mu)^2$ is less than $10^8$ in practical cases. For the logic operation problem, the complexity is reduced from $O(n^2\epsilon^3)$ to $O(IG(n_\varepsilon)^2)$, and $IG$ is a hyperparameter less than $0.5\pi$ in most cases.

Two ideal examples are presented to verify the accuracy and derivations about time complexity, and both cases converge with an error rate of less than 1%. And the practical test case of Viral Hepatitis B runs three times faster than the recursive algorithm with an error rate of less than 2.7%.

In future work, two shortcases should be overcome. First, the burn-in parameters $b$ and window width parameters $\omega$ are assigned based on experience, so that a little deviation between actual and theoretical results remains. Second, estimation for logic operation is based on hyperparameters $IG_{layer}$ and $IG_{len}$, resulting in that this scheme may fail in some edge situations.

REFERENCES

[1] Q. Zhang, “Dynamic uncertain causality graph for knowledge representa-
and reasoning: Discrete DAG cases,” J. Comput. Sci. Technol., vol. 27,
nob. 1, pp. 1–23, Jan. 2012, doi: 10.1007/s11390-012-2002-7.

[2] Q. Zhang, C. Dong, Y. Cui, and Z. Yang, “Dynamic uncertain causality graph for knowledge representation and probabilistic reasoning: Statistics base, matrix, and application,” IEEE Trans. Neu-
ral Netw. Learn. Syst., vol. 25, no. 4, pp. 645–663, Apr. 2014, doi: 10.1109/TPNN.2013.2279329.

[3] Q. Zhang, “Dynamic uncertain causality graph for knowledge represen-
tation and probabilistic reasoning: Directed cyclic graph and joint proba-
bility distribution,” IEEE Trans. Neural Netw. Learn. Syst., vol. 26, no. 7,
pp. 1503–1517, Jul. 2013, doi: 10.1109/TPNN.2015.242016.

[4] Q. Zhang and Z. Zhang, “Dynamic uncertain causality graph applied to
dynamic fault diagnoses and predictions with negative feedbacks,” IEEE Trans. Rel., vol. 65, no. 2, pp. 1030–1044, Jun. 2016, doi: 10.1109/TR.2015.2503759.

[5] Q. Zhang, K. Duan, and Q. Yao, “Recursive algorithm in DUCG,” (in Chinese), Beijing Tsingrui Intell. Technol., Beijing, China, Tech. Rep.
2017-2, 2017.

[6] Q. Zhang, X. Bu, M. Zhang, Z. Zhang, and J. Hu, “Dynamic uncertain causality graph for computer-aided general clinical diagnoses with nasal obstruc-
tion as an illustration,” Artif. Intell. Rev., vol. 54, no. 1, pp. 27–61,
Jan. 2021, doi: 10.1007/s10462-020-09871-0.

[7] Q. Zhang, “Probabilistic reasoning based on dynamic causality trees/diagrams,” Rel. Eng. Syst. Saf., vol. 46, no. 3, pp. 209–220,
1994, doi: 10.1016/0951-8320(94)90114-7.

[8] C. Dong, Z. Zhou, and Q. Zhang, “Cubic dynamic uncertain causality graph: A new methodology for modeling and reasoning about com-
plex faults with negative feedbacks,” IEEE Trans. Rel., vol. 67, no. 3,
pp. 920–932, Sep. 2018, doi: 10.1109/TR.2018.2822479.

[9] L. Li and W. Yue, “Dynamic uncertain causality graph based on intuition-
istic fuzzy sets and its application to root cause analysis,” Int. J. Speech Tech.
vol. 50, no. 1, pp. 241–255, Jan. 2020, doi: 10.1016/j.procs.2017.06.023.

[10] X. Zhao, F. D. Maio, E. Zio, Q. Zhang, C.-L. Dong, and J.-Y. Zhang, “Optimization of a dynamic uncertain causality graph for fault diagnosis in nuclear power plant,” Nucl. Sci. Techn., vol. 28, no. 3, p. S81, 2017, doi: 10.1016/j.nucstr.2017.06.023.

[11] Q. Yao, Q. Zhang, P. Liu, P. Yang, M. Zhu, and X. Wang, “Application of dynamic uncertain causality graph in spacecraft fault diagnosis: Multi-
conditions,” AIP Conf. Proc., vol. 1834, no. 1, p. 30012, Mar. 2017, doi: 10.1063/1.4981577.