# Project 2: Approximate Inference: Sampling and Variational method

Zeming Fang
RIN 661958922

## 1. Introduction

One of the advantages of using a probabilistic graphic model (PGM) is to do machine reasoning on top of it. Reasoning using the PGM was formalized as an inference problem.

In the previous project, we implemented the exact method of inference, belief propagation. However, exactly computing the posterior is very hard when a Bayesian network (BN) model scales up because the computation complexity grows exponentially with the model's scale. In this case, we may seek an approximation method for the solution for a large scale BN. Among all current available approximate inference methods, sampling, and variational methods are the two most popular techniques.

In this project, we first briefly review the key concepts of both methods. Then we will use some experimental evidence to show the advantages and disadvantages of each algorithm.

## 2. Sampling Method

In this project, we first briefly review the key concepts of both methods. Then we will use some experimental evidence to show the advantages and disadvantages of each algorithm.

The sampling method avoids the theoretical derivation of any closed-form solutions. Instead, it approximates the underlying distribution by sampling using the conditional probability distribution (CPD). In the discrete scenario, the idea of sampling is much straight forward. When given parent configuration, the conditional probability table (CPT) within a node follows the multinomial distribution, of which the parameters are proportional to the frequencies of the data. With the growth of the sample size, the sampling method will approach the real distribution.

The most basic sampling method is ancestral sampling. This method samples from CPT in a sequence that follows the topological order of the nodes in BN. However, this method does not work when we calculate posterior in inference problem where evidence is introduced. In the following two subsections, we introduce two popular approaches to overcome the evidence introduction problem: the likelihood weighted sampling method and the Gibbs sampling method.

## 2.1 likelihood weighed sampling

The solution proposed by the likelihood weighted sampling method is rather than sampling the evidence nodes, we fix the evidence node and weight the sample using the CPT weight (See the Algorithm1 for more detail).

The likelihood weighed method has some disadvantages (see section 5 results for more details). The biggest among them is its sensitivity to the sample sequence. When sampling, the sample is drawn from $p(X_i|\pi(X_i)=j)$, which requires the known of the state of the parent nodes. In other words, the current nodes' parents should already generate a sample before sampling the current nodes. To generate a sequence that follows BN's topological structure is not easy when BN becomes complicated and includes loops. Since the sampling methods were proposed to solve complicated BN, this weakness tells that the likelihood weighted method is not an ideal sampling method.

## 2.2 Gibbs sampling

Any way to avoid deciding a sampling sequence? An intuitive idea is not to reset the BN before each sampling and use the previous samples. To formalize this idea, the MCMC sampling method was proposed, where a Markov chain is built with each sample called a state, denoted as $S$. We can see each node as a feature of the state $S=[X_1, X_2, ..., X_N]^\top$, and all features interact follows a BN model. As time passes, each state will evolve to the next state, follow a certain transition function $T(S_t + 1|S_t)$. This transition function between states depends on the change of multiple features.

**Algorithm 1** Likelihood weighted sampling

```
1: Input A Bayesian Network G
2: Input A sequence that order BN variables $X_1, X_2, ..., X_N$ according to the topological order
3: Input An evidence set $X_e = E$
4: Initialize weights for all samples $w_1, w_2, ..., w_T$ to 1.
5: for $t = 1$ to $T$ do
6:     for each node $X_i$ in the Bayesian network G do
7:         if $X_n^t$ is not evidence then
8:             sample $x_n^t$ from $p(X_n^t|\pi(X_n^t)$
9:         else
10:            $x_n^t = e_n$
11:            $w_t = w_t * p(X_n^t = e_n|\pi(X_n^t)$
12:        end if
13:    end for
14:    For sample $x^t = \{x_1^t, x_2^t, ..., x_N^t\}$ and compute its weight $w_t$
15: end for
16: Return $(\mathrm{x}^1, w_1), (\mathrm{x}^2, w_2), ..., (\mathrm{x}^T, w_T)$
```

Gibbs sampling said we could make assumptions about the states' features by assuming only one feature change during the transition from one state to another. This assumption allows us to derive the transition function analytically, $T(S_t + 1|S_t) = p(X_n|X_{1:N\backslash n})$. We can further simplify the transition function using the structural independence properties from BN:

$$\begin{aligned} &p(X_n|MB(X_n)) \\ \propto &p(X_n|\pi(X_n)) \prod_{Y_k \in child(X_n)} p(Y_k|\pi(Y_k)) \end{aligned} \tag{2.1}$$

Where $MB(\cdot)$ means Markov blanket. Once we derive the Markov chain's transition function, we can simulate the Markov chain until its stationarity or call it the end of burn-in period. Once burned-in, the distribution underlying this Markov chain will not change anymore, and this unchanged distribution is called stationary distribution of the Markov chain. It has been proved that the stationary distribution is the distribution in BN, and we can start to collect the state as samples to estimate this stationary distribution (See algorithm2 for pseudo code).

There are a few issues about Gibbs sampling. One of them is that to implement Gibbs sampling, we need to determine when to start sampling (See more detail in the section 5 results).

**Algorithm 2** Gibbs sampling

```
1: Input A Bayesian Network G
2: Initialize $X = \{X_1, X_2, ..., X_N\}$ to $\mathrm{x}^0 = \{x_1, x_2, ..., x_N\}$
3: t = 0
4: while not end of burn-in period do
5:     Randomly select a node, $n \sim uniform(N)$
6:     Obtain a new sample $x_n^{t+1} \sim p(X_n|\mathrm{x}_{-n}^t)$
7:     Form a new sample $\mathrm{x}^{t+1} = \{x_1^t, x_2^t, ..., x_n^{t+1}, ..., x_N^t\}$
8:     t = t+1
9: end while
10: $\mathrm{x}^0 = \{x_1^t, x_2^t, ..., x_N^t\}$
11: for $t = 0$ to T do
12:     Randomly select a node, $n \sim uniform(N)$
13:     Obtain a new sample $x_n^{t+1} \sim p(X_n|\mathrm{x}_{-n}^t)$
14:     Form a new sample $\mathrm{x}^{t+1} = \{x_1^t, x_2^t, ..., x_n^{t+1}, ..., x_N^t\}$
15: end for
16: Return $\mathrm{x}^1, \mathrm{x}^{1+k}, \mathrm{x}^{1+2k}, ..., \mathrm{x}^T$
```

## 3. Mean-field Variational Method

The variational method was first developed by Jordan's group at UC Berkeley (cit). The idea of the Variational method is to identify a simpler surrogate distribution $q(X)$ to approximate the true distribution $p(X|e)$, and conduct inference with the surrogate function $q(X)$. Despite turning a one-step (infer) problem into a two-step (approximate and infer) problem, the variational method simplifies the inference problem in terms of computational complexity.

There is "no free lunch" (Wolpert and Macready, 1997). As the cost of enjoying the computational efficiency, we need to bring priors and assumptions into the inference problem. One simple and strong but effective prior is the mean-field assumption. In this project, we mainly discussed how to apply the mean-field assumption on the variational inference problem, including finding the surrogate function $p(x)$ and using this surrogation to do inference.

### 3.1 Key equations of variational method

As we mentioned, one objective of the variational method is to propose a surrogate function that is close to the real distribution $p(X|e)$ as possible. Building the surrogate function is usually under the guidance of some assumptions, and, in the section, it is guided by the mean-field assumption, denoted as $q_{\mathrm{mf}}$. This objective can be mathematically defined as:

$$\min_{q_{\mathrm{mf}}} KL(q_{\mathrm{mf}}(X)||p(X|e)) \tag{3.1}$$

Where $KL(p(A)||p(B))$ means the Kullback-Leiber divergence (KL Divergence) between distribution A and distribution B. Although KL divergence is not a distance measurement, it is used substantially as a measure the difference between distributions.

However, we cannot develop any practical solutions based on the equation 3.1, because it is hard to compute $p(X|e)$. We can consider to compute $p(X|e)$ using the joint distribution $p(X, e)$ which is much simpler in a Bayesian network. Thus, we can rewrite the objective function (equation $3.1$) as (see Appendix $7.1$ for derivation):

$$\min_{q_{\mathrm{mf}}} \sum_{\mathrm{x}} q_{\mathrm{mf}}(\mathrm{x}) \log q_{\mathrm{mf}}(x) + \sum_{\mathrm{x}} q_{\mathrm{mf}}(\mathrm{x}) \log p(\mathrm{x}, e) \tag{3.2}$$

Note that equation $3.2$ is also called free energy. To understand this term, I quoted the definition in (Friston, 2011): “An information theory measure that bounds or limits (by being greater than) the surprise on sampling some data, given a generative model.” The generative model if this project is notated as $q_{\mathrm{mf}}$. In simple, we can reiterate the free-energy as the degree to a generative model fails to predict the real condition. We want our model as accurate as possible, so we need to minimize the free energy.

We can further simplify the equation $3.2$ based on our mean-field assumption. The derivation is:

$$
\begin{aligned}
&\sum_{\mathrm{x}} q_{\mathrm{mf}}(\mathrm{x}) \log q_{\mathrm{mf}}(x) + \sum_{\mathrm{x}} q_{\mathrm{mf}}(\mathrm{x}) \log p(\mathrm{x}, e) \\
=& \sum_{x_i \in X} \sum_{x_i} q(x_i, \theta_i) \log q(x_i, \theta_i) - \sum_{\mathrm{x}} \prod_{x_i \in X} q(x_i, \theta_i) \log p(\mathrm{x}, e) \\
=& \sum_{x_i} q(x_i, \theta_i) \log(x_i, \theta_i) + \sum_{x_j \in \mathrm{x} \setminus x_i} \sum_{x_j} q(x_j, \theta_j) \log(q_j, \theta_j) \\
&- \sum_{x_i} q(x_i, \theta_i) \sum_{\mathrm{x} \setminus x_i} \prod_{x_j \neq x_i \in X} q(x_j, \theta_j) \log p(\mathrm{x}, e) \qquad (3.3) \\
=& \sum_{x_i} q(x_i, \theta_i) \log(x_i, \theta_i) + K_{\mathrm{x} \setminus x_i} - \sum_{x_i} q(x_i, \theta_i) \mathbb{E}_{\mathrm{x} \setminus x_i} [\log p(\mathrm{x}, e)] \\
=& \sum_{x_i} q(x_i, \theta_i) \log \frac{q(x_i, \theta_i)}{\exp(\mathbb{E}_{\mathrm{x} \setminus x_i} [\log p(\mathrm{x}, e)])} + K_{\mathrm{x} \setminus x_i} \\
=& KL(q(x_i, \theta_i) || \exp(\mathbb{E}_{\mathrm{x} \setminus x_i} [\log p(\mathrm{x}, e)])) + K_{\mathrm{x} \setminus x_i}
\end{aligned}
$$

To minimize the equation $3.3$, we only need to minimize the KL term, which means let $q(x_i, \theta_i) = \exp(\mathbb{E}\mathrm{x} \setminus x_i [\log p(\mathrm{x} \setminus x_i, x_i, e)])$. Since we are discussing the discrete BN, $q(x_i, \theta_i)$ is a categorical distribution. Thus we can derive the close-form solution for discrete BN with mean-field assumption as:

$$
\theta_{ik} = \frac{\exp(\mathbb{E}_{\mathrm{x} \setminus x_i} [\log p(\mathrm{x} \setminus x_i, x_i = k, e)])}{\sum_{j=1}^{K} \exp(\mathbb{E}_{\mathrm{x} \setminus x_i} [\log p(\mathrm{x} \setminus x_i, x_i = j, e)])} \qquad (3.4)
$$

We can apply the I-map to further simplify this equation 3.4, as (see Appendix 7.2 for detial):

$$
\begin{aligned}
\theta_{ik} \propto & \sum_{m \in \pi(x_i)} q(x_m, \theta_m) \log p(x_i = k | \pi(x_i)) + \\
& \sum_{x_e \in E} \sum_{x_m \in \pi(x_e)} q(x_m, \theta_m) \log p(x_e = e | \pi(x_e)) \qquad (3.5) \\
& + \sum_{x_l \in \mathrm{x} \sum_{\setminus} x_i} \sum_{x_m \in x_l, \pi(x_l)} \prod q(x_m, \theta_m) \log p(x_l | \pi(x_l))
\end{aligned}
$$

The equation $3.5$ is tractable, so based on this we can compute updated the paramter $\theta_{ik}$. Note that the summation is outside the expectation term, so we can conduct parallel computing for solving the equation $3.5$.

### 3.2 Pseudo code of mean-field variational method

**Algorithm 3** Mean field Variational Method

```
**Input** a Bayesian Graph $G$ with evidence $e$ and unobserved variables $X$.
**Output** mean field parameters $\Theta = \{\theta_{ik}\}$ for k = 1, 2, ..., K
Randomly initialize the parameters $\theta_{ik}$ for k = 1, 2, ..., K
**while** $\theta_{ik}$ not converging **do**
    **for** each node $X_i$ in the graph **do**
        **for** each state $k$ for $X_i$ **do**
            Compute $\theta_{ik}$ using equation 3.5
        **end for**
    **end for**
**end while**
```

## 4. Experiment Setting

In this project we are instructed to implement approximate inference methods to perform inference on the CHILD BN, see figure 4.1 for the structure.

This CHILD network is for diagnosing congenital heart disease in a new born “blue” babies (project2 instruction, 2020)

The inference problem we need to solve is:

- Posterior probability inference: p( BirthAsphyxia = yes| CO2report = “<7.5”, LVHReport=yes, X-rayReport = Plethoric)
- MAP inference: Disease* = $\arg\max_{\text{Disease}}$ p(Disease|CO2report = “<7.5”, LVHReport=yes, X-rayReport = Plethoric)

## 5. Results and Discussions

In this section, we will show the performance of each approximation algorithms respectively. Some disclaimers:

1. I guess my implementation might be slower than the other people. To reuse some code across algorithms, I use a general method that allows me to calculate expectations and choose the evidence using the same function. However, since I treat all evidence selection as expectation sum and product,

the algorithm's speed should slow down.
2. Analyzing the experimental results turns out to be unexpectedly thought due to the lack of ground truth. Alternatively, we use the data from other methods during the evaluation. I hope my implementation is not too faulty.

### 5.1 results generated with chosen hyperparameters

In this section, I post the result with the tuned hyperparameters (see Table1).

| | $p(\text{BirthAsphyxia} = \text{yes} \mid E)$ | $\arg\max p(\text{Disease} \mid E)$ |
|---|---|---|
| Weighted sampling | 0.099 | TGA |
| Gibbs sampling | 0.084 | not stable |
| Mean field approximation | 0.090 | TGA |

When implementing likelihood weighted sampling, I choose sample size as 3000. The performance of likelihood weighted sampling is quite stable in both questions. For question one, the answer range from (0.93 - 1.05); question two, stably return TGA.

My Gibbs sampling implementation does not work. The answer is fickle. I used uniform initialization, 6000 burn-steps, 10000 samples with skip step as 4.

The mean-field method is very stable. It always returns 0.09008, and TGA. If we look at the posterior of $p(\text{Disease} \mid E)$ of mean-field algorithm, we may find the probability pile up at TGA, reaching .99.

### 5.2 Results for likelihood weighted method

According to the instruction, the main issue in the likelihood weighted method is bias and the sampling order. I would like to add a discussion about the number os the samples.

#### *5.2.1 Number of the samples.*

Table 2 summarizes the impact of the number of samples on estimation. I ran sampling with each sampling size 3 times (More would be better) and evaluated the impact of samples size from three aspects. Here we use the answer of question1 as p( BirthAsphyxia = yes| CO2report = "<7.5", LVHReport=yes, X-rayReport = Plethoric) as target for my evaluations.

- Mean of estimation: I was supposed to use “correctness”. The problem is we do not know the ground truth. Here we use the answer of question1 as p( BirthAsphyxia = yes| CO2report = “<7.5”, LVHReport=yes, X-rayReport = Plethoric) as data for evaluation.
- Reliability: The standard deviation over 3 trials is used to evaluate the reliability of sampling with different sample sizes.
- Time: Very straight forwards.

Table2: The impact of sample size on sampling.

| | **n=100** | **n=1000** | **n=3000** | **n=10000** |
|---|---|---|---|---|
| $P_{\text{mean}}$ | 0.1067 | 0.0973 | 0.0973 | 0.100 |
| reliability | 0.0330 | 0.00287 | 0.00287 | 0.0007 |
| Time | 0.8s | 7s | 20s | 70s |

We may have the following claims:

1. Small sample size results in higher variance, but not too bad. The sample size does not have much impact on the expectation of the estimation.
2. The time costs by the algorithm grows linearly with the number of samples.
3. 1000 samples might be a reasonable choice as a hyperparameter in this problem set. It has a better trade-off between time and accuracy compared with other candidates.

### ***5.2.2 The importance of choosing a good sequence.***

Finding a sequence that follows BN’s topological structure might be one of the fundamental problems in the likelihood weighted sampling method because the algorithms do not even work without fixing a reasonable sequence. Sampling from a node $X_i$ requires the acknowledgment of the value in its parents’ node $\pi(X_i)$, or the algorithm does not know from what CPT to sample.

### ***5.2.3 The bias in likelihood weighed sampling***

Maybe it is because my Gibbs sampling does not work, but what I found that even with little sample (see section 5.2.1), the performance of likelihood weighed sampling is very stable. Maybe this is a bias-variance trade off.

### 5.3 Results for Gibbs sampling method

In this section we are instructed to discuss the effect of initialization, burn-in time, and skip time. Prior to the discussion of these concepts, we should mention that for Gibbs sampling, generating a sample is much easily than that in likelihood weighted method, because in Gibbs sampling, to generate a new sample, we only need to do 1 sampling operation.

#### *5.3.1 The effect of initialization*

During the experiments, I tried to initiate the Markov chain using both uniform initialization and sampling from CPT. No signal explicitly shows the difference between these two initializations, so I am not posting in the report. One possible explanation of this phenomenon is that my experiment was at a rough scale, and maybe through fine analytics, we can see the impact of the initialization. Another explanation is that what we need from the Markov chain is the final stationary distribution, which is initialization-invariant. Maybe the burn-in time is one of the variables that are sensitive to the Markov chain's initialization. Perhaps a good initialization can reduce the burn-in time step.

#### *5.3.2 Burnin time as a hyperparameter Gibbs sampling.*

In MCMC sampling, the only useful distribution for approximating the underlying distribution is the stationary distribution after the burn-in period. The choice of the burn-in period is usually an art in doing Gibbs sampling. We discussed several adaptive ways to decide the burn-in period in class, but I found them not as exciting as it should be, at least in this project. To tune one important hyper-parameter, you need to add several new hyperparameters, like the implementation of statistical properties between two consecutive windows, the size of the consecutive windows, the number of parallel chains. One reasonable and simple method for this project is to pick several candidates arbitrarily and experimentally find the best one.

#### 5.3.3 The impact of skip step size

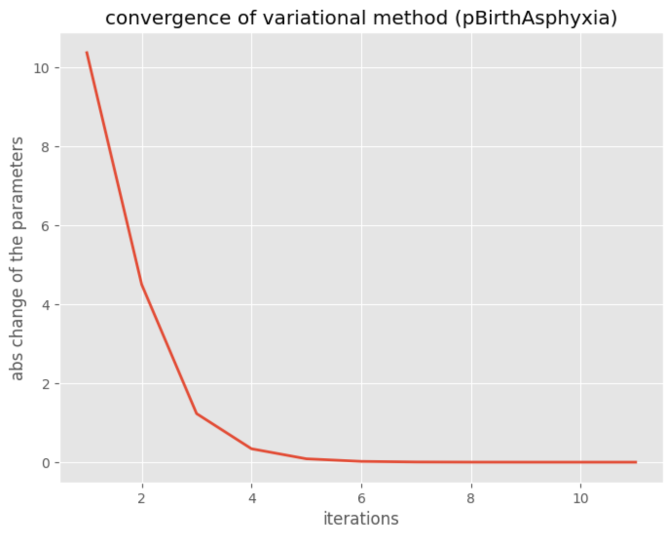
We mentioned introducing a skipping window during the sampling period to decorate the corrected samples in class. This is a straight forward idea because, in the Markov chain, the current state will be biased by some recent states. In this project, we created a table to offer how correlation will affect the sampling performance.

```
Here we did a grid search over the burin-in time step ([ 30, 3000,
10000]) and skip step ([1, 4, 10, 20], note that 1 means no skip
steps). We repeated the sampling with each  "burnin"-"skipstep" pairs
six times, can calculate the mean as "correctness" and standard
deviation as "reliability".
```

I was planning to make a heat map to show how burn-in and skip step affect sampling performance. However, there is some bug in my Gibbs sampling implementation and I cannot debug it.

## 5.3 Results for variational method

Unlike Gibbs sampling, the variational method does not have many hyperparameters to tune thanks to its strong mean-field assumption. Here we briefly should the convergence rate of the mean-field algorithm in this project (see figure3).



Here we only plot the convergence plot for question 1, because it is the same in question 2 because both questions share the same evidence. We can see from the plot, the mean-field algorithm converges quickly after 9 iterations (I forced the algorithm to run 3 extra iterations for the purpose of visualization). What's more, the mean-field algorithm only takes 7 seconds, which is faster than previous sampling method.

An unexpected observation in the mean-field, if we check the posterior distribution in Question2, we may find the probability on the “TGA” piles up to 99.9% percentage, which is different from that from likelihood weighed sampling and Gibbs sampling, which are pretty close to CPT. Maybe, this is the results of mean-field assumption (or my mistake).

## 6. Conclusion

My likelihood weighed sampling and mean-field algorithm works stably on this problem. However, the Gibbs sampling does not work well.

## 7. Appendix

### 7.1 Derivation of the objective function

$$\begin{aligned}
&\min_{q_{\mathrm{mf}}} KL(q_{\mathrm{mf}}(X)||p(X|e)) \\
&= \min_{q_{\mathrm{mf}}} \sum_{\mathrm{x}} q_{\mathrm{mf}}(\mathrm{x}) \log q_{\mathrm{mf}}(\mathrm{x}) + \sum_{\mathrm{x}} q_{\mathrm{mf}}(\mathrm{x}) \log p(\mathrm{x}, e) \\
&\qquad - \sum_{\mathrm{x}} q_{\mathrm{mf}}(\mathrm{x}) \log p(e) \\
&= \min_{q_{\mathrm{mf}}} \sum_{\mathrm{x}} q_{\mathrm{mf}}(\mathrm{x}) \log q_{\mathrm{mf}}(x) + \sum_{\mathrm{x}} q_{\mathrm{mf}}(\mathrm{x}) \log p(\mathrm{x}, e) - \log p(e) \\
&\Rightarrow \min_{q_{\mathrm{mf}}} \sum_{\mathrm{x}} q_{\mathrm{mf}}(\mathrm{x}) \log q_{\mathrm{mf}}(x) + \sum_{\mathrm{x}} q_{\mathrm{mf}}(\mathrm{x}) \log p(\mathrm{x}, e)
\end{aligned} \tag{7.1}$$

### 7.2. Simplify the close form solution of the parameter

$$\begin{aligned}
&\mathbb{E}_{\mathrm{x}\backslash x_i}[\log p(\mathrm{x}\backslash x_i, x_i = k, E)]) \\
&= \sum_{x_l \in \mathrm{x}\backslash x_i} \prod_{x_l \in \mathrm{x}\backslash x_i} q(x_l, \theta_l)[\log \prod_{x_l \in \mathrm{x}\backslash x_i} p(x_l|\pi(x_l)) p(x_i = k|\pi(x_i)) p(x_e = e|\pi(x_e))] \\
&= \sum_{x_l \in \mathrm{x}\backslash x_i} \prod_{x_l \in \mathrm{x}\backslash x_i} q(x_l, \theta_l)[\sum_{x_l \in \mathrm{x}\backslash x_i} \log p(x_l|\pi(x_l)) + \log p(x_i = k|\pi(x_i)) \\
&\quad + \sum_{x_e \in E} \log p(x_e = e|\pi(x_e))]
\end{aligned}$$

$$
\begin{aligned}
&= \sum_{x_l \in \mathrm{x} \setminus x_i} \prod_{x_l \in \mathrm{x} \setminus x_i} q(x_l, \theta_l) \sum_{x_l \in \mathrm{x} \setminus x_i} \log p(x_l | \pi(x_l)) \\
&\quad + \sum_{x_l \in \mathrm{x} \setminus x_i} \prod_{x_l \in \mathrm{x} \setminus x_i} q(x_l, \theta_l) \log p(x_i = k | \pi(x_i)) \\
&\quad + \sum_{x_l \in \mathrm{x} \setminus x_i} \prod_{x_l \in \mathrm{x} \setminus x_i} q(x_l, \theta_l) \sum_{x_e \in E} \log p(x_e = e | \pi(x_e))] \\
&= \sum_{x_l \in \mathrm{x} \setminus x_i} \prod_{x_l \in \mathrm{x} \setminus x_i} q(x_l, \theta_l) \sum_{x_l \in \mathrm{x} \setminus x_i} \log p(x_l | \pi(x_l)) + 1 * \log p(x_i = k | \pi(x_i)) \\
&\quad + 1 * \sum_{x_e \in E} \log p(x_e = e | \pi(x_e)) \\
&= \log p(x_i = k | \pi(x_i)) + \sum_{x_e \in E} \log p(x_e = e | \pi(x_e)) \\
&\quad + \sum_{x_l \in \mathrm{x} \setminus x_i} \prod_{x_l \in \mathrm{x} \setminus x_i} q(x_l, \theta_l) [\log p(x_p | \pi(x_p) + \sum_{x_l \in \mathrm{x} \setminus x_i, x_p} \log p(x_l | \pi(x_l))] \\
&= \log p(x_i = k | \pi(x_i)) + \sum_{x_e \in E} \log p(x_e = e | \pi(x_e)) \qquad (7.2) \\
&\quad + \sum_{x_l \in \mathrm{x} \setminus x_i} \prod_{x_l \in \mathrm{x} \setminus x_i} q(x_l, \theta_l) \log p(x_p | \pi(x_p)) \\
&\quad + \sum_{x_l \in \mathrm{x} \setminus x_i} \prod_{x_l \in \mathrm{x} \setminus x_i} q(x_l, \theta_l) \sum_{x_l \in \mathrm{x} \setminus x_i, x_p} \log p(x_l | \pi(x_l))] \\
&= \log p(x_i = k | \pi(x_i)) + \sum_{x_e \in E} \log p(x_e = e | \pi(x_e)) \\
&\quad + \sum_{x_p} q(x_p, \theta_p) \log p(x_p | \pi(x_p)) \sum_{x_l \in \mathrm{x} \setminus x_i, x_p} \prod_{x_l \in \mathrm{x} \setminus x_i, x_p} q(x_l, \theta_l) \\
&\quad + \sum_{x_l \in \mathrm{x} \setminus x_i} \prod_{x_l \in \mathrm{x} \setminus x_i} q(x_l, \theta_l) \sum_{x_l \in \mathrm{x} \setminus x_i, x_p} \log p(x_l | \pi(x_l))] \\
&= \sum_{m \in \pi(x_i)} q(x_m, \theta_m) \log p(x_i = k | \pi(x_i)) + \sum_{x_e \in E} \sum_{x_m \in \pi(x_e)} q(x_m, \theta_m) \log p(x_e = e | \pi(x_e)) \\
&\quad + \sum_{x_p, \pi(x_p)} \prod_{x_m \in x_p, \pi(x_p)} q(x_m, \theta_m) \log p(x_p | \pi(x_p)) \\
&\quad + \sum_{x_l \in \mathrm{x} \setminus x_i} \prod_{x_l \in \mathrm{x} \setminus x_i} q(x_l, \theta_l) \sum_{x_l \in \mathrm{x} \setminus x_i, x_p} \log p(x_l | \pi(x_l))] \\
&= \sum_{m \in \pi(x_i)} q(x_m, \theta_m) \log p(x_i = k | \pi(x_i)) + \sum_{x_e \in E} \sum_{x_m \in \pi(x_e)} q(x_m, \theta_m) \log p(x_e = e | \pi(x_e)) \\
&\quad + \sum \quad \sum \quad \prod q(x_m, \theta_m) \log p(x_p | \pi(x_p))
\end{aligned}
$$

$$+ \sum_{x_l \in \mathrm{x} \sum \setminus x_i} \sum_{x_m \in x_l, \pi(x_l)} \prod q(x_m, \theta_m) \log p(x_l | \pi(x_l))$$