

Quantum Annealing for Dirichlet Process Mixture Models with Applications to Network Clustering

Issei Sato^a, Shu Tanaka^b, Kenichi Kurihara^c, Seiji Miyashita^d, Hiroshi Nakagawa^a

^aInformation Technology Center, University of Tokyo, 7-3-1, Hongo, Bunkyo-ku, Tokyo, 113-0033, Japan

^bDepartment of Chemistry, University of Tokyo, 7-3-1, Hongo, Bunkyo-ku, Tokyo, 113-0033, Japan

^cGoogle, 6-10-1, Roppongi, Minato-ku, Tokyo, 106-6126, Japan

^dDepartment of Physics, University of Tokyo, 7-3-1, Hongo, Bunkyo-ku, Tokyo, 113-0033, Japan

Abstract

We developed a new quantum annealing (QA) algorithm for Dirichlet process mixture (DPM) models based on the Chinese restaurant process (CRP). QA is a parallelized extension of simulated annealing (SA), i.e., it is a parallel stochastic optimization technique. Existing approaches (Kurihara et al., 2009; Sato et al., 2009) cannot be applied to the CRP because their QA framework is formulated using a fixed number of mixture components. The proposed QA algorithm can handle an unfixed number of classes in mixture models. We applied QA to a DPM model for clustering vertices in a network where a CRP seating arrangement indicates a network partition. A multi core processor was used for running QA in experiments, the results of which show that QA is better than SA, Markov chain Monte Carlo inference, and beam search at finding a maximum a posteriori estimation of a seating arrangement in the CRP. Since our QA algorithm is as easy as to implement the SA algorithm, it is suitable for a wide range of applications.

Keywords: Quantum annealing, Dirichlet process, Stochastic optimization, Maximum a posteriori estimation, Bayesian nonparametrics

1. Introduction

Clustering is one of the most important topics in machine learning because it is a fundamental approach to analyze differences and similarities of data. In statistical machine learning, a probabilistic latent variable model is used for clustering. The Dirichlet process mixture (DPM) models (Antoniak, 1974) are well studied and they enable us to handle an unfixed number of classes, which means that we do not have to decide the number of classes in advance. In other words, they can estimate the number of classes according to data. A DPM model is often represented by the Chinese restaurant process (CRP) (Aldous, 1985), in which clustering is represented as a seating arrangement of customers in a restaurant. This representation is a useful one helping us understand the clustering process in DPM models.

A clustering problem using a probabilistic model is generally formulated as a maximum a posteriori (MAP) estimation in statistical machine learning.

Since finding the exact MAP solution will be difficult in many cases, we have to search for an approximate one. Markov chain Monte Carlo (MCMC) inference is widely used for the CRP (Neal, 2000) but the MCMC is not necessarily appropriate for the MAP estimation. When we use MCMC for the MAP estimation, we extract a single class assignment with the highest probability in the class assignments sampled from the posterior distribution. The problem is that this sampling distribution (i.e., the posterior distribution) has to be stationary, and much iteration is needed before it converges.

DaumeIII (2007) showed that a beam search provides an attractive alternative to the MCMC in the CRP and another approach for the MAP estimation is a stochastic search. One of the most well-known stochastic search algorithms is simulated annealing (SA) (Kirkpatrick et al., 1983), which is similar to the MCMC but has an additional parameter, called a temperature, controlling the uncertainty of the search space. SA is known to find the global optimum when the cooling temperature reduction

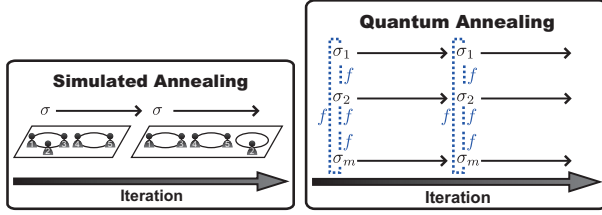


Figure 1: The left-hand panel shows the running of SA, in which σ indicates a seating arrangement of N customers in the CRP (i.e., a class assignment of N data points). The right-hand panel shows QA, in which multiple SAs interact through f . σ_j indicates a seating arrangement of the CRP running in the j -th process. Note in QA that σ_m is interacted with σ_{m-1} and σ_1 (i.e., $\sigma_{m+1} = \sigma_1$), which is mathematically derived from the QA framework (Theorem 3.1). During iterations, we control the hyper-parameters.

schedule is slow enough (Geman and Geman, 1984) but such a schedule is too slow for practical use. SA with a practical cooling schedule is therefore also affected by a local optimization problem.

In this work, we focus on a novel stochastic search algorithm, quantum annealing (QA), which has attracted attention as an alternative annealing method for optimization problems (Kadowaki and Nishimori, 1998; Farhi et al., 2001; Santoro et al., 2002) in quantum information science (Lloyd, 1996; Nielsen and Chuang, 2000). QA has been shown experimentally converge faster than to find better local optimums for Ising spin models. It has a parameter inducing quantum fluctuation, so the search space is controlled in a way different from that in SA. The details are explained below.

QA is a parallelized extension of SA in which quantum fluctuation is induced by running multiple SAs with interactions. Let us consider running m SAs, and let σ_j ($j = 1, \dots, m$) indicate a state (e.g., a class assignment) of N data points in the j -th simulation. In the CRP, we formulate σ_j as a table-seating arrangement of customers in the j -th CRP (see Fig. 1). We denote N data points as $\mathbf{x} = (x_1, x_2, \dots, x_N)$. The log-likelihood model given σ_j (i.e., $\log p(\mathbf{x}, \sigma_j)$) is based on the way the data is modeled. For simplicity, we denote the log-likelihood by $\log p(\sigma_j)$. In this work, we use the Newman model (Newman and Leicht, 2007) for clustering network data (explained in Sec.4). QA runs multiple *dependent* SAs with *dependent* here meaning that there is interaction f among neighboring SAs (see right-hand panel in Fig. 1).

We describe QA in terms of an optimization

problem. When we run m SAs with different random initializations independently, we optimize $\log p(\sigma_j)$ individually. That is, we find $\sigma_j^* = \operatorname{argmax}_{\sigma_j} \log p(\sigma_j)$ for each j and we choose σ that has the highest $\log p(\sigma_j^*)$ of all j . In QA, we optimize the joint probability of m CRPs' states $\{\sigma_j\}_{j=1}^m$:

$$\max_{(\sigma_1, \sigma_2, \dots, \sigma_m)} \log p_{\text{QA}}(\{\sigma_1, \sigma_2, \dots, \sigma_m\}), \quad (1)$$

where $p_{\text{QA}}(\cdot)$ is a probability measure over a set of states, which means that each state σ_j ($j = 1, \dots, m$) can take an independent state and QA gives the probability for these states. A set of states $(\sigma_1, \sigma_2, \dots, \sigma_m)$ represents (quantum) superposition of different states. That is, $p_{\text{QA}}(\cdot)$ is a probability measure over superposition of different states in the limit of $m \rightarrow \infty$ in quantum physics. In the CRP, $(\sigma_1, \sigma_2, \dots, \sigma_m)$ represents a superposition of m seating arrangements. The optimization problem (1) is actually formulated as

$$\max_{(\sigma_1, \sigma_2, \dots, \sigma_m)} \sum_{j=1}^m \log p_{\text{SA}}(\sigma_j) + f \cdot R(\sigma_1, \sigma_2, \dots, \sigma_m), \quad (2)$$

where the first term corresponds to the summation over m SA objectives, and $R(\cdot)$ is regarded as a regularizer among m states, which are described in Sec.3.5. This optimization is derived from the QA framework explained in Sec. 3. QA was recently used for solving practical optimization problems, such as clustering (Kurihara et al., 2009) and variational Bayes inference (Sato et al., 2009), and it outperformed SA. Figure 2 summarizes QA and related work.

Problems: Existing approaches (Kurihara et al., 2009; Sato et al., 2009) cannot be applied to the CRP because they need a fixed number of mixture components. Moreover, these approaches have to use a heuristic such as purity to apply their QA algorithms to the clustering problem. Therefore, a different formulation is needed.

Contributions: The purpose of this study is to propose a QA algorithm for the CRP. The key point is how to represent the states of data in the CRP. The existing work represents the data states as “which class a data point is assigned to.” That is, they require K -dimensional indicator vectors to represent the data states, and K (the number of classes) is given and fixed. We instead represent the

	MCMC	target field Optimization (MAP estimation)
No Interaction	Metropolis Hastings, Gibbs sampler, Slice sampler, etc.	EM algorithm, Beam search, Simulated annealing, etc.
Interaction	Exchange Monte Carlo (Parallel Tempering), Particle filter, etc.	Quantum annealing

Figure 2: QA and related algorithms. We categorize the algorithms into two groups according to their purposes: MCMC and optimization. MCMC aims at approximating the expectation by a finite sum of samples drawn from the posterior distribution and therefore needs a large number of iterations. Optimization algorithms search for optimal solutions in a small number of iterations. The algorithms are also classified according to the presence of interactions among multiple processes. Note that expectation maximization (EM) algorithms (Dempster et al., 1977) and variational inferences (Blei and Jordan, 2005; Kurihara et al., 2007) cannot be applied to the CRP.

states of data as “which data points a data point shares the table with” in the CRP. That is, we use an N -by- N bit matrix to represent the data states, and this matrix indicates a seating arrangement in the CRP and does not depend on K . This bit-matrix representation of the CRP is a novel idea and a key point in applying QA to the CRP. Note that the bit matrix is only used for mathematically deriving QA for the CRP and is not used in the actual algorithm. Mathematically, this novelty appears in interaction function f in Eq.(17), where our derived f does not include the number of classes, K , whereas f in existing work is formulated by using (fixed) K . Moreover, our algorithm does not require heuristics such as purity. We also use parallel processing in QA, whereas Kurihara et al. (2009) and Sato et al. (2009) used a single processing.

The idea of using a matrix that represents the relationship among data points has been used in several studies (L. Xu and Oja, 1993; Frey and Dueck, 2007; Wang and Lai, 2011). These studies used a similarity matrix based on the feature vectors among data points. For example, x_i and x_j denote the feature vectors of the i -th and j -th data points and the similarity was calculated by using the Gaussian kernel between x_i and x_j . The formulation in this paper is different from these existing approaches because we do not use the similarity matrix based on the feature vectors. We used

the matrix of data points to formulate the clustering state of data in a learning process. For example, a bit-matrix in this study changes in a learning process, while the similarity (kernel) matrix in the existing works does not change but is fixed. Moreover, we use a bit-matrix only for mathematically deriving QA for the CRP. We do not directly use the matrix in an actual algorithm, whereas the existing approaches directly use the similarity matrix for clustering data.

2. Chinese Restaurant Process (CRP)

The CRP is a distribution over partitions such as clustering and is composed of three elements: a customer, table, and restaurant. In a clustering problem, the customer denotes a data point and the table denotes a data class. A seating arrangement of customers in a restaurant indicates a class assignment of data. In QA, we run multiple CRPs, i.e., we consider the seating arrangements in multiple restaurants.

The CRP assigns a probability for the seating arrangement of the customers in which $\mathbf{Z} = \{z_i\}_{i=1}^N$ denotes the seating arrangement of the customers and $z_i = k$ indicates that customer i sits at the k -th table. N indicates the number of customers. When customer i enters a restaurant with K occupied tables at which other customers are already seated, customer i sits at a table with the following probability:

$$p(z_i | \mathbf{Z} \setminus z_i; \alpha) \propto \begin{cases} \frac{N_k}{\alpha + N - 1} & (k\text{-th occupied table}), \\ \frac{\alpha}{\alpha + N - 1} & (\text{new unoccupied table}), \end{cases} \quad (3)$$

where N_k denotes the number of customers sitting at the k -th table, and α is the hyper parameter of the CRP. A customer tends to select a new table when α takes large value.

The log-likelihood of \mathbf{Z} is given by $p(\mathbf{Z}) = \frac{\alpha^{K(\mathbf{Z})}}{\prod_{i=1}^N (\alpha + N - i + 1)} \prod_{k=1}^{K(\mathbf{Z})} (N_k - 1)!$, where $K(\mathbf{Z})$ is the number of occupied tables in \mathbf{Z} .

3. Quantum Annealing for CRP

This section explains how we derive QA for the CRP (QACRP). First, we introduce some notations and explain QACRP intuitively. Second, we introduce a bit matrix to reformulate the CRP for using QA independent of the number of classes, which

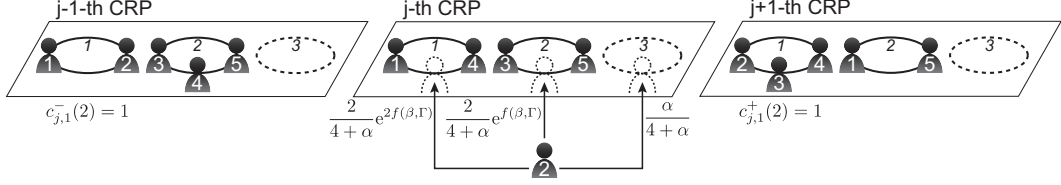


Figure 3: Example of approximation inference in QACRP where $\beta = m$. Let us consider adding customer 2 to restaurant j (j -th CRP). The classical CRP seats customers at existing tables in proportion to the number of customers already seated (see Eq.(3)). The QACRP sampler derived in Eq.(4) introduces the effect of customers who share tables with customer 2 in the $(j-1)$ -th and $(j+1)$ -th CRPs. In this case, since customers 1, 3, and 4 are customers who share tables with customer 2 in the $(j-1)$ -th and $(j+1)$ -th CRPs, the 1st table in the j -th CRP has these “two” customers and so takes the effect $e^{“2”f(\beta, \Gamma)}$ into account ($c_{j,1}^-(2) + c_{j,1}^+(2) = 2$ in Eq. (4)). That is, in the QACRP sampler, when interaction $f(\beta, \Gamma)$ is large, a customer tends to sit with customers sharing the table in other CRPs.

is a key idea in this work. Third, we formulate the CRP by using a “density matrix” that is a basic formulation in quantum mechanics. Finally, we apply the Suzuki-Trotter expansion (Trotter, 1959; Suzuki, 1976) to approximate QACRP because the first-derived QACRP is intractable because of the computational cost of a matrix exponential.

3.1. Main result (See Fig. 3 for an intuitive image)

QACRP uses multiple restaurants. $z_{j,i} = k$ indicates that customer i sits at the k -th table in the j -th restaurant. $\mathbf{Z}_j = \{z_{j,i}\}$ denotes the seating arrangements of customers in the j -th restaurant. $c_{j,k}^+(i)$ denotes the number of customers who sit at the k -th table in the j -th restaurant and share tables with customer i in the $(j+1)$ -th restaurant. $c_{j,k}^-(i)$ denotes the number of customers who sit at the k -th table in the j -th restaurant and share tables with customer i in the $(j-1)$ -th restaurant. Customer i sits at a table in the j -th restaurant with the following probability:

$$p_{\text{QA}}(z_{j,i} | \{\mathbf{Z}_d\}_{d=1}^m \setminus \{z_{j,i}\}; \beta, \Gamma) \propto \begin{cases} \left(\frac{N_{j,k}}{\alpha + N - 1} \right)^{\frac{\beta}{m}} e^{(c_{j,k}^-(i) + c_{j,k}^+(i))f(\beta, \Gamma)} & (k\text{-th occupied table}), \\ \left(\frac{\alpha}{\alpha + N - 1} \right)^{\frac{\beta}{m}} & (\text{new unoccupied table}), \end{cases} \quad (4)$$

where $N_{j,k}$ denotes the number of customers sitting at the k -th table in the j -th restaurant. $f(\beta, \Gamma)$ is derived in Sec.3.5 Eq.(17) where β and Γ are hyper parameters that are called inverse temperature and quantum effect, respectively. The inverse temperature is also the hyper parameter of SA. When you

change the CRP in Eq.(3) into QACRP in Eq.(4), all you have to do is to count the customers sharing tables in neighboring CRPs and introduce $f(\beta, \Gamma)$. Figure 3 shows an example of QACRP and provides an intuitive explanation. When $f(\beta, \Gamma) = 0$, QACRP is equivalent to m independent CRPs with inverse temperature β/m , which we call SACRPs. We provide the details of the derivation in the next sections.

3.2. Bit matrix representation for CRP

We represent seating arrangement \mathbf{Z} by using a bit matrix \mathbf{B} in order to reformulate the CRP without fixing the number of tables (see Contributions in Sec. 1). Although this bit matrix representation seems to have high computational complexity, in an actual algorithm of QA, we do not need the direct calculation to the bit matrix.

A bit matrix \mathbf{B} looks like an adjacency matrix of customers (see Fig. 4) and \mathbf{B} denotes an N -by- N bit matrix where B_i is the i -th row vector, i.e., $B_i = (B_{i,1}, B_{i,2}, \dots, B_{i,N})$, and $B_{i,n}$ is the i -th row and the n -th column element of \mathbf{B} or the n -th element of B_i . $\tilde{\sigma}_{i,n}$ ($i, n = 1, \dots, N$) is a two-dimensional indicator vector, i.e., it takes $(1, 0)^\top$ or $(0, 1)^\top$. We correspond $B_{i,n} = 1$ to $\tilde{\sigma}_{i,n} = (1, 0)^\top$ and $B_{i,n} = 0$ to $\tilde{\sigma}_{i,n} = (0, 1)^\top$, which means we can represent \mathbf{B} by using the 2^{N^2} dimensional indicator vector, σ , as follows:

$$\mathbf{B} \Leftrightarrow \sigma = \bigotimes_{i=1}^N \bigotimes_{n=1}^N \tilde{\sigma}_{i,n}. \quad (5)$$

\bigotimes is the Kronecker product, which is a special case of a tensor product. If A is a k -by- l matrix and B is an m -by- n matrix, then the Kronecker

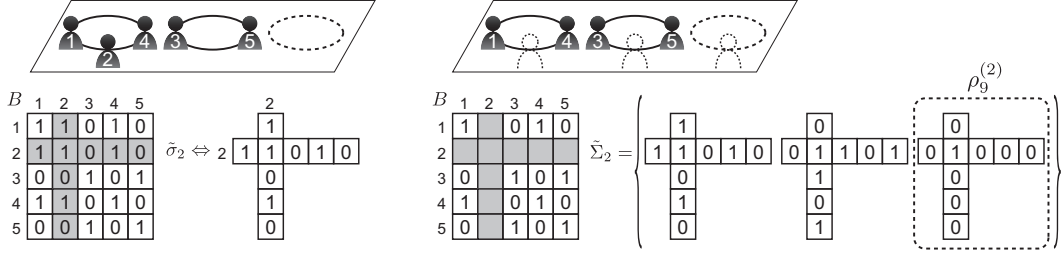


Figure 4: Example of bit matrix representation. A seating arrangement \mathbf{Z} is represented as a bit matrix \mathbf{B} , which enables us to formulate the CRP without fixing the number of tables. For example, $\tilde{\sigma}_2$ represents customers who share a table with customer 2. $\tilde{\Sigma}_2$ represents a set of the states that customer 2 can take under the seating conditions, and $\rho_9^{(2)}$ indicates that customer 2 sits alone at a table.

product $A \otimes B$ is the following km -by- ln block matrix: $A \otimes B = \begin{pmatrix} a_{11}B & \cdots & a_{1l}B \\ \vdots & \ddots & \vdots \\ a_{k1}B & \cdots & a_{kl}B \end{pmatrix}$. For example, $(1, 0)^\top \otimes (0, 1)^\top = (0, 1, 0, 0)^\top$. Σ denotes a set of σ , i.e., $|\Sigma| = 2^{N^2}$.

The bit matrix \mathbf{B} is regarded as a seating arrangement as follows. If $B_{i,n} = 1$, the i -th and the n -th customers share a table. Note that we need the following conditionals to represent seating arrangements with the bit matrix

1. $B_{i,n} = B_{n,i}$ (symmetric matrix)
2. $B_{i,i} = 1 (i = 1, 2, \dots, N)$, i.e., $\text{Tr } B = N$
3. $\forall i$ and l , $\frac{B_i}{|B_i|} \cdot \frac{B_l}{|B_l|} = 1$ or 0 , where \cdot is the inner product.

$\text{Tr } X$ is the trace of X . $\tilde{\Sigma}(\subset \Sigma)$ denotes a set of σ corresponding to \mathbf{B} satisfying the above conditions. We call these conditions “seating conditions.”

Here, $\tilde{\sigma}_i$ indicates the state of the i -th customer, i.e., with whom the i -th customer shares a table (see the left-hand side of Fig. 4) and $\tilde{\sigma}_i$ is a $(2N-1)$ -dimensional indicator vector given by

$$\tilde{\sigma}_i = \bigotimes_{n=1}^{i-1} \tilde{\sigma}_{i,n} \otimes \tilde{\sigma}_{i,i} \otimes \bigotimes_{n=i+1}^N \tilde{\sigma}_{i,n} \bigotimes_{n=1, n \neq i}^N \tilde{\sigma}_{n,i}. \quad (6)$$

Let $\tilde{\Sigma}_i$ be a set of the states that $\tilde{\sigma}_i$ can take under the seating conditions (i.e., $\sigma \in \tilde{\Sigma}$) when the i -th row elements and the i -th column elements are blank and the others are filled (see the right-hand side of Fig. 4). Since $\tilde{\Sigma}_i$ is a set of table-assignment states of the i -th customer, $|\tilde{\Sigma}_i| = K(\mathbf{Z} \setminus \{z_i\}) + 1$. For example, the right-hand side of Fig. 4 shows table assignments of the 2nd customer when customers 1, 3, 4, and 5 have already been seated.

$\rho_{2N-1}^{(i)}$ is defined as a $2N-1$ dimensional indicator vector given by

$$\rho_{2N-1}^{(i)} = \bigotimes_{n=1}^{i-1} (0, 1)^\top \otimes (1, 0)^\top \otimes \bigotimes_{n=i+1}^N (0, 1)^\top \bigotimes_{n=1, n \neq i}^N (0, 1)^\top. \quad (7)$$

The right-hand side of Fig. 4 shows an example of $\rho_{2N-1}^{(i)}$. We use $\rho_{2N-1}^{(i)}$ only in Appendix A.

3.3. Density matrix representation for classical CRP

We define the energy function E over $\sigma^{(l)} \in \Sigma$ ($l = 1, \dots, 2^{N^2}$) by $E(\sigma^{(l)}) = -\log p(\sigma^{(l)})$, where if $\sigma^{(l)} \in \Sigma \setminus \tilde{\Sigma}$, then $p(\sigma^{(l)}) = 0$, i.e., $E(\sigma^{(l)}) = +\infty$.

The probability of a state $\sigma \in \Sigma$ is given by

$$p(\sigma) = \frac{1}{Z} \sigma^\top e^{-\mathcal{H}_c} \sigma, \quad (8)$$

where $\mathcal{H}_c = \text{diag} [E(\sigma^{(1)}), E(\sigma^{(2)}), \dots, E(\sigma^{(2^{N^2})})]$, and $\text{diag}[\cdot]$ denotes a diagonal matrix. Note that $Z = \sum_{\sigma} \sigma^\top e^{-\mathcal{H}_c} \sigma = \text{Tr } e^{-\mathcal{H}_c}$, where \mathcal{H}_c is called the classical Hamiltonian. If $\sigma \in \tilde{\Sigma}$, then $p(\sigma)$ is equal to $p(\mathbf{Z})$, i.e., $p(\sigma)$ is the probability over a seating arrangement. Since \mathcal{H}_c is diagonal, $e^{-\mathcal{H}_c}$ is also diagonal with the k -th diagonal element $e^{-E(\sigma^{(k)})}$. That is, $p(\sigma^{(k)}) = \frac{1}{Z} \sigma^{(k)\top} e^{-\mathcal{H}_c} \sigma^{(k)} = \frac{1}{Z} e^{-E(\sigma^{(k)})}$.

3.4. Formulation for quantum CRP

The basic approach to expanding a classical system to a quantum one is to make the Hamiltonian

non-diagonal, i.e., add some off-diagonal elements while keeping hermiticity. We define a non-diagonal matrix \mathcal{H} by

$$\mathcal{H} = \mathcal{H}_c + \mathcal{H}_q, \quad (9)$$

where \mathcal{H}_q is a non-diagonal matrix (we describe the definition of \mathcal{H}_q later). Intuitively, diagonal elements are filled with zero, and some off-diagonal elements are filled with Γ in \mathcal{H}_q . That is, \mathcal{H} is filled with energy $E(\sigma)$ in diagonal elements and quantum effect Γ in off-diagonal elements. The above scheme that adds a non-diagonal matrix (\mathcal{H}_q) to a diagonal matrix (\mathcal{H}_c) is a basic approach in quantum physics and has also worked well in (Kurihara et al., 2009; Sato et al., 2009). The meaning of this formulation was described in (Sato et al., 2009) in terms of uncertainty.

The probability of a state $\sigma (\in \Sigma)$ in a quantum system is given by

$$p_{QA}(\sigma; \beta, \Gamma) = \frac{1}{Z} \sigma^\top e^{-\beta(\mathcal{H}_c + \mathcal{H}_q)} \sigma, \quad (10)$$

where $Z = \sum_{\sigma} \sigma^\top e^{-\beta(\mathcal{H}_c + \mathcal{H}_q)} \sigma = \text{Tr}[e^{-\beta(\mathcal{H}_c + \mathcal{H}_q)}]$.

The optimization problem

$$\max_{\sigma} \log p_{QA}(\sigma; \beta, \Gamma) \quad (11)$$

could be solved by using the eigenvalue decomposition of the density matrix $\frac{1}{Z} e^{-\beta(\mathcal{H}_c + \mathcal{H}_q)}$, but, this approach is intractable because of its large computational cost.

One approximation approach for solving the optimization problem (11) is a stochastic search by drawing a state of the i -th customer, $\tilde{\sigma}_i$, from

$$p(\tilde{\sigma}_i | \sigma \setminus \tilde{\sigma}_i) = \frac{\sigma^\top e^{-\mathcal{H}_c} \sigma}{\sum_{\tilde{\sigma}_i} \sigma^\top e^{-\mathcal{H}_c} \sigma}, \quad (12)$$

$$p_{QA}(\tilde{\sigma}_i | \sigma \setminus \tilde{\sigma}_i; \beta, \Gamma) = \frac{\sigma^\top e^{-\beta(\mathcal{H}_c + \mathcal{H}_q)} \sigma}{\sum_{\tilde{\sigma}_i} \sigma^\top e^{-\beta(\mathcal{H}_c + \mathcal{H}_q)} \sigma}, \quad (13)$$

where $\sigma \setminus \tilde{\sigma}_i$ indicates that bits excluding the i -th row and the i -th column elements are standing. The summation over $\tilde{\sigma}_i$ is actually the summation of $\tilde{\sigma}_i \in \tilde{\Sigma}_i$; therefore, the classical system $p(\tilde{\sigma}_i | \sigma \setminus \tilde{\sigma}_i)$ is tractable when $p(\tilde{\sigma}_i | \sigma \setminus \tilde{\sigma}_i)$ in Eq. (12) is another expression of Eq. (3). Calculation of the probability of the quantum system $p_{QA}(\tilde{\sigma}_i | \sigma \setminus \tilde{\sigma}_i; \beta, \Gamma)$, however, is intractable because of the exponential operation of a non-diagonal matrix $\mathcal{H} = \mathcal{H}_c + \mathcal{H}_q$. We therefore need another approach described in Sec.3.5.

We define \mathcal{H}_q as follows.

$$\mathcal{H}_q = -\Gamma \sum_{i=1}^N \sum_{n=1}^N \sigma_{i,n}^x, \quad \mathbb{E} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}, \quad \sigma^x = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix},$$

$$\sigma_{i,n}^x = \left(\bigotimes_{t=1}^{i-1} \bigotimes_{u=1}^N \mathbb{E} \right) \otimes \left[\left(\bigotimes_{t=1}^{n-1} \mathbb{E} \right) \otimes \sigma^x \otimes \left(\bigotimes_{t=n+1}^N \mathbb{E} \right) \right] \otimes \left(\bigotimes_{t=i+1}^N \bigotimes_{u=1}^N \mathbb{E} \right), \quad (14)$$

where Γ is the quantum effect parameter. This formulation means that diagonal elements are filled with zeros, and some off-diagonal elements are filled with Γ in \mathcal{H}_q . Although other definitions can be considered, we define this formulation so that we can make the derivation of the search algorithm tractable by using an approximation method that is easy to implement.

3.5. Approximation inference for QACRP

We cannot calculate $p_{QA}(\tilde{\sigma}_i | \sigma \setminus \tilde{\sigma}_i; \beta, \Gamma)$ in Eq.(13) because $\sigma^\top e^{-\beta \mathcal{H}} \sigma$ is intractable because of the non-diagonal matrix \mathcal{H} . We use the Suzuki-Trotter expansion (Trotter, 1959; Suzuki, 1976) to approximate $p_{QA}(\tilde{\sigma}_i | \sigma \setminus \tilde{\sigma}_i; \beta, \Gamma)$.

We consider multiple running CRPs in which $\sigma_j (j = 1, \dots, m)$ indicates the seating arrangement of the j -th CRP and represents the j -th bit matrix \mathbf{B}_j . We correspond $B_{j,i,n} = 1$ to $\tilde{\sigma}_{j,i,n} = (1, 0)^\top$ and $B_{j,i,n} = 0$ to $\tilde{\sigma}_{j,i,n} = (0, 1)^\top$, which means that we can represent \mathbf{B}_j as σ_j by using Eq.(5). We derive the following theorem:

Theorem 3.1. $p_{QA}(\sigma; \beta, \Gamma)$ in Eq.(10) is approximated by the Suzuki-Trotter expansion as follows:

$$\begin{aligned} p_{QA}(\sigma; \beta, \Gamma) &= \frac{1}{Z} \sigma^\top e^{-\beta(\mathcal{H}_c + \mathcal{H}_q)} \sigma \\ &= \sum_{\sigma_j (j \geq 2)} p_{QA-ST}(\sigma, \sigma_2, \dots, \sigma_m; \beta, \Gamma) + \mathcal{O}\left(\frac{\beta^2}{m}\right), \end{aligned} \quad (15)$$

where we rewrite σ as σ_1 , and

$$p_{QA-ST}(\sigma_1, \sigma_2, \dots, \sigma_m; \beta, \Gamma) = \prod_{j=1}^m \frac{1}{Z(\beta, \Gamma)} e^{-\frac{\beta}{m} E(\sigma_j)} e^{f(\beta, \Gamma) s(\sigma_j, \sigma_{j+1})}, \quad (16)$$

$$f(\beta, \Gamma) = 2 \log \coth \left(\frac{\beta}{m} \Gamma \right), \quad (17)$$

$$s(\sigma_j, \sigma_{j+1}) = \sum_{i=1}^N \sum_{n=1}^N \delta(\tilde{\sigma}_{j,i,n}, \tilde{\sigma}_{j+1,i,n}), \quad (18)$$

$$Z(\beta, \Gamma) = \left[\sinh \left(\frac{\beta}{m} \Gamma \right) \right]^{2N} \sum_{\sigma} e^{-\frac{\beta}{m} E(\sigma)}. \quad (19)$$

Note that $\sigma_{m+1} = \sigma_1$. The proof is given in Appendix A. Note that our derived f in Eq.(17) does not include the number of classes, K , whereas the f in existing work (Kurihara et al., 2009; Sato et al., 2009) is formulated by using a fixed K .

Equation (15) is interpreted as follows. $p_{QA}(\sigma; \beta, \Gamma)$ is approximated by marginalizing out other states $\{\sigma_j\}_{j \geq 2}$ of $p_{QA-ST}(\sigma_1, \sigma_2, \dots, \sigma_m; \beta, \Gamma)$. As shown in Eq.(16), $p_{QA-ST}(\sigma_1, \sigma_2, \dots, \sigma_m; \beta, \Gamma)$ looks like the joint probability of the states of m dependent CRPs. In Eq.(16), $e^{-\frac{\beta}{m} E(\sigma_j)}$ corresponds to the classical CRP with inverse temperature and $e^{f(\beta, \Gamma) s(\sigma_j, \sigma_{j+1})}$ indicates the quantum effect part. If $f(\beta, \Gamma) = 0$, which means CRPs are independent, $p_{QA-ST}(\sigma_1, \sigma_2, \dots, \sigma_m; \beta, \Gamma)$ is equal to the products of probability of m classical CRPs. $s(\sigma_j, \sigma_{j+1}) (> 0)$ is regarded as a similarity function between the j -th and $(j+1)$ -th bit matrices. If they are the same matrices, then $s(\sigma_j, \sigma_{j+1}) = N^2$. In Eq.(2), $\log p_{SA}(\sigma_j)$ corresponds to $\log e^{-\frac{\beta}{m} E(\sigma_j)} / Z$ and the regularizer term $f \cdot R(\sigma_1, \dots, \sigma_m)$ is $\log \prod_{j=1}^m e^{f(\beta, \Gamma) s(\sigma_j, \sigma_{j+1})} = f(\beta, \Gamma) \sum_{j=1}^m s(\sigma_j, \sigma_{j+1})$.

Note that we aim at deriving the approximation inference for $p_{QA}(\tilde{\sigma}_i | \sigma \setminus \tilde{\sigma}_i; \beta, \Gamma)$ in Eq.(13). Using Theorem 3.1, we can derive Eq.(4) as the approximation inference. The details of the derivation are provided in Appendix B.

4. Experiments

We evaluated QA in a real application. We applied QA to a DPM model for clustering vertices in a network where a seating arrangement of the CRP indicates a network partition.

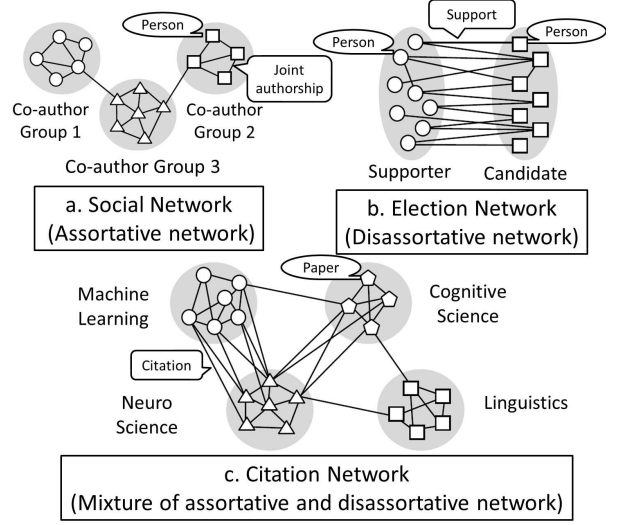


Figure 5: Examples of Network structures.

4.1. Network Model

We used the Newman model (Newman and Leicht, 2007) for network modeling in this work. The Newman model is a probabilistic generative network model. This model is flexible, which enables researchers to analyze observed graph data without specifying the network structure (disassortative or assortative) in advance.

In an assortative network, such as a social network, the members (vertices) of each class are mostly connected to the other members of the same class. The communications between members in three social groups is illustrated in Fig. 5, where one sees that the members generally communicate more with others in the same group than they do with those outside the group. In a disassortative network, the members (vertices) have most of their connections outside their class. An election network of supporters and candidates is illustrated in Fig. 5-b, where a link indicates support for a candidate. The Newman model can model not only these two kinds of networks but also a mixture of them, such as a citation network (see Fig.5-c), but, the user must decide in advance the number of classes. We therefore used the DPM extension of the Newman model as described in Appendix C.

4.2. Dataset

We used three social network datasets, Netscience¹, Citeseer², and Wikivote³. Netscience is a coauthorship network of scientists working on a network that has 1,589 scientists (vertices). Citeseer is a citation network dataset for 2,110 papers (vertices). Wikivote is a bipartite network constructed using administrator elections and vote history data in Wikipedia. Its 7,115 vertices represent Wikipedia users and a directed edge from vertex i to vertex j represents that user i voted for user j . Netscience, Wikivote, and Citeseer respectively correspond to network examples a, b, and c in Fig.5. We used the vertices in these networks to represent customers in the CRP, and we used a negative log-likelihood as an energy function to find the MAP solution.

4.3. Annealing schedule

We tested several β/m schedules using combinations of $\beta_0 = 0.2m$, $0.4m$, and $0.6m$ and $\beta = \beta_0 \log(1+t)$, $\beta_0 \sqrt{t}$, and $\beta_0 t$, where t denotes the t -th iteration. The results we observed in our experiments showed that $\beta_0 = 0.4m$ and $\beta_0 \sqrt{t}$ created a better schedule in SA in terms of the MAP estimation. That is, β increases to $\beta_0 \sqrt{T}$, where T is the total number of iterations. In QA, we use the same β/m schedule we used in SAs. Note that since the new table is easy to sample at very small β (where the probability distribution becomes flat, see Eq. (4)), the SACRP has many tables at small β and converges very slowly. That is, inverse temperatures that are too low do not work well in the CRP.

Since interaction f is a function of Γ and β , in practice we have to consider the schedule of $f(\beta, \Gamma)$. The interaction $f(\beta, \Gamma)$ increases when $\frac{\beta\Gamma}{m}$ decreases. QA is known to work well when $f(\beta, \Gamma)$ starts from zero (i.e., “independent” multiple SAs) and gradually increases. This process of $f(\beta, \Gamma)$ is achieved when $\frac{\beta\Gamma}{m}$ is a decreasing function of t . Therefore, we use

$$\frac{\beta\Gamma}{m} = \Gamma_0 \frac{T}{t}, \quad (20)$$

where Γ_0 is a tuning parameter.

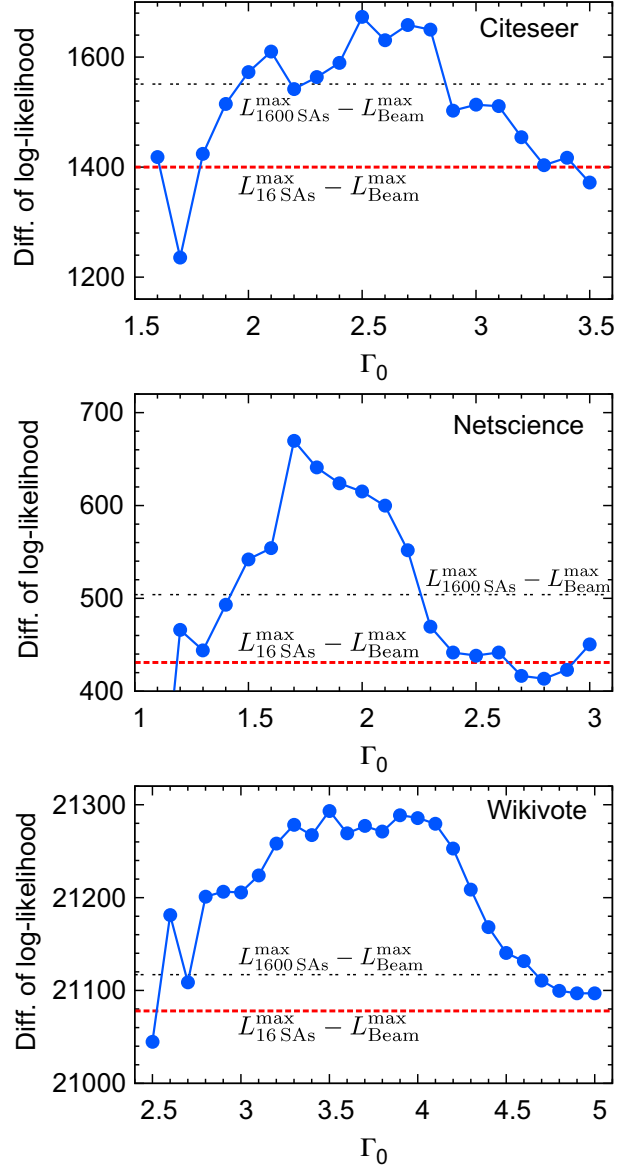


Figure 6: Experimental results for the maximum log-likelihoods (minimum energies) of QA and other algorithms. Vertical axes show the difference of maximum log-likelihoods. A higher value is better (closer to optimum states). We denote L_{QA}^{\max} , $L_{16\ SA_s}^{\max}$ as the maximum log-likelihood of 16 CRPs in QA and SA, $L_{1600\ SA_s}^{\max}$ as the maximum log-likelihood of 1600 CRPs in SA, and L_{Beam}^{\max} as the maximum log-likelihood of the beam search with beam-width = 100. The solid line indicates $L_{QA}^{\max} - L_{Beam}^{\max}$. Lower (red) and upper dotted lines indicate $L_{16\ SA_s}^{\max} - L_{Beam}^{\max}$ and $L_{1600\ SA_s}^{\max} - L_{Beam}^{\max}$. When these lines take positive values, QA and SA outperform the beam search. Whenever the solid line is higher than the dotted lines QA outperforms SAs. The horizontal axes are the Γ_0 for QA.

¹http://www.casos.cs.cmu.edu/computational_tools/datasets/external/netscience/

²<http://www.cs.umd.edu/projects/linqs/projects/lbc/index.html>

³<http://snap.stanford.edu/data/wiki-Vote.html>

4.4. Experimental settings

Our purpose is to search for a better MAP solution to a CRP in a small number of iterations (or short running time). We evaluated optimization algorithms in terms of maximum log-likelihood because we want a state with the highest log-likelihood. We compared QA with SA and the beam search. We used the beam search with an inadmissible score function that achieved the best performance in (DaumeIII, 2007). We set the beam-width to 100. We did not compare the variational inference with QA because the variational inference cannot deal with the Chinese restaurant formulation of the Dirichlet process mixture. That is, it is hard to compare them because their objective functions are different.

Since we used a multi core processor with 16 cores, we set $m = 16$ (i.e., ran one CRP at one core) We set $\alpha = 1$ in the CRP. α is easy to estimate in SAs and QA, but beam search cannot estimate it; therefore, we fixed it in these experiments. The number of iterations, T , for SAs and QA was 30. We generated 16 random seating arrangements $\{\sigma_j^{(\text{random})}\}_{j=1}^{16}$ for the initial settings of QA and 16 SAs, i.e., we use $\sigma_j^{(\text{random})}$ for the same initial setting of the j -th seating arrangement. Moreover, we compared QA ($m = 16$) with 1600 SAs where their MAP solutions are the best one of 1600 simulations with different random initializations. In 1600 SAs, we tried 100 seeds and generated $m = 16$ random seating arrangements for each seed, i.e., we ran the CRPs with $100 \times m (= 16) = 1600$ initial seating arrangements.

4.5. Results and Discussions

Figure 6 shows the experimental results. QA and SAs outperformed the beam search because each line takes a positive value. QA finds a better local optimum than that of 16 and 1600 SAs at some Γ_0 . This means that it is useful to run QA with changing Γ_0 rather than to run multiple SAs.

The effective Γ_0 has a positive correlation with the number of nodes. For example, the effective Γ_0 is around 2 in Netscience and is around 3.5 in Wikivote, which has more nodes than that Netscience does. This is because the quantum effect term depends on $C \cdot f(\beta, \Gamma)$, where C is the number of customers who share tables and thus depends on the number of customers (nodes). This means that the effective parameter range can be inferred from

the number of nodes and we have only to check a few Γ_0 values.

QA needs more time and memory than SA with the linear order of m because QA uses m CRPs. However, when a parallel processing environment can be used and we run multiple SAs in parallel, the scalability of QA is the same as that of SAs. QA ($T = 30, m = 16$) and SA ($T = 30, m = 1$) took about 15 and 13 seconds for Netscience, about 25 and 22 seconds for Wikivote, and about 79 and 76 seconds for Wikivote, where each value was the averaged running time of a single simulation for SA. Because of the multi core processing and the caching of customers sharing tables, the running time of QA was almost the same as that of a single SA. Therefore, QA makes the CRPs converge faster and finds a better seating arrangement than multiple-run SAs. The estimated number of classes achieving the best performance in QA ($m = 16$), 16 SAs, 1600 SAs and the beam search are 26, 22, 65, and 61 for Netscience, 37, 35, 30, and 57 for Citeseer, and 8, 8, 8, and 27 for Wikivote.

We found that a small Γ_0 induces a fast schedule of f , which means $f \gg 0$ at small β . The fast schedules make the convergence of QA too fast; therefore, QA converges at a worse optimum. QA is similar to SAs at large Γ_0 because interaction f remains at almost 0 for a limited number of iterations. Larger Γ_0 makes CRPs in QA more independent, which means the results of QA approach those of SA. We found that interaction f is almost zero when $\Gamma_0 = 5$ and $T = 30$, which means that the performance of QA is similar to that of SA. Therefore, in practice, we only check values of Γ_0 in descending order from a large value of Γ_0 , such as $\Gamma_0 = 5$. That is, the effective value range is easy to infer from some Γ_0 values (in our experimental results, we show some non-effective values in order to provide the negative examples of QA).

5. Conclusion

We proposed a QA algorithm for the DPM models based on the CRP. Our algorithm is different from those of Kurihara et al. (2009) and Sato et al. (2009) in three ways: (i) it can handle an unfixed number of classes in mixture models, (ii) it does not require heuristics such as a purity, and (iii) it uses parallel processing in QA. The proposed algorithm (Eq. (4)) is easy to implement because it is similar to a classical CRP (Eq. (3)). That is, it is easy to apply the proposed algorithm to

other nonparametric models with which it is not easy to apply beam search, such as an infinite relational model (Kemp et al., 2006). The proposed algorithm will therefore be a promising new optimization technique when it is used with rapidly advancing multi core processors. As shown in Eq.(2), our algorithm is regarded as an optimization with a regularized term and its performance depends on parameter f like that other optimization algorithms with a regularized term does (e.g., L1 and L2 regularized optimization algorithms often used in machine learning). For future work, it will be worth analyzing what kind of schedule of f enables QA to work well.

Acknowledgements

The present work was partially supported by the Mitsubishi Foundation, and also by the Next Generation Super Computer Project, Nanoscience Program from MEXT of Japan. The authors are partially supported by Grand-in-Aid for JSPS Fellows (23-7601). Numerical calculations were partly performed on supercomputers at the Institute for Solid State Physics, University of Tokyo. This research was funded in part by the Joint Usage/Research Center for Interdisciplinary Large-scale Information Infrastructures in Japan. This work was supported by a JSPS Grant-in-Aid for Young Scientists (B) 24700135.

Appendix A. Proof of Theorem 3.1

We use the following Trotter product formula (Trotter, 1959) to approximate

$$p_{\text{QA}}(\sigma; \beta, \Gamma) = \frac{1}{Z} \sigma^\top e^{-\beta(\mathcal{H}_c + \mathcal{H}_q)} \sigma. \quad (\text{A.1})$$

If A_1, \dots, A_L are symmetric matrices, we have

$$\exp\left(\sum_{l=1}^L A_l\right) = \left[\prod_{l=1}^L \exp(A_l/m)\right]^m + O\left(\frac{1}{m}\right). \quad (\text{A.2})$$

Applying the Trotter product formula to Eq.(A.1) with finite m , we have

$$\begin{aligned} p_{\text{QA}}(\sigma; \beta, \Gamma) &= \frac{1}{Z} \sigma^\top e^{-\beta(\mathcal{H}_c + \mathcal{H}_q)} \sigma \\ &\approx \frac{1}{Z} \sigma^\top \left(e^{-\frac{\beta}{m}\mathcal{H}_c} e^{-\frac{\beta}{m}\mathcal{H}_q}\right)^m \sigma. \end{aligned} \quad (\text{A.3})$$

We evaluate the residual of this approximation. Since $e^{A_1+A_2} = e^{A_1}e^{A_2}$ does not hold in general⁴, we need to use the Trotter product formula for computation. We rewrite σ as σ_1 and note that $\sigma_1^\top (e^A)^2 \sigma_1 = \sum_{\sigma_2} \sigma_1^\top e^A \sigma_2 \sigma_2^\top e^A \sigma_1$. Hence, we express Eq.(A.3) by marginalizing out auxiliary variables $\{\sigma'_1, \sigma_2, \sigma'_2, \dots, \sigma_m, \sigma'_m\}$,

$$\begin{aligned} &\sigma_1^\top \left(e^{-\frac{\beta}{m}\mathcal{H}_c} e^{-\frac{\beta}{m}\mathcal{H}_q}\right)^m \sigma_1 \\ &= \sum_{\sigma'_1} \sum_{\sigma_2} \dots \sum_{\sigma_m} \sum_{\sigma'_m} \sigma_1^\top e^{-\frac{\beta}{m}\mathcal{H}_c} \sigma'_1 \sigma_1^\top e^{-\frac{\beta}{m}\mathcal{H}_q} \sigma_2 \\ &\quad \dots \sigma_m^\top e^{-\frac{\beta}{m}\mathcal{H}_c} \sigma'_m \sigma_m^\top e^{-\frac{\beta}{m}\mathcal{H}_q} \sigma_{m+1} \\ &= \sum_{\sigma'_1} \sum_{\sigma_2} \dots \sum_{\sigma_m} \sum_{\sigma'_m} \prod_{j=1}^m \sigma_j^\top e^{-\frac{\beta}{m}\mathcal{H}_c} \sigma'_j \sigma_j^\top e^{-\frac{\beta}{m}\mathcal{H}_q} \sigma_{j+1}, \end{aligned} \quad (\text{A.4})$$

where $\sigma_{m+1} = \sigma_1$. To express Eq.(A.4) more particularly, we use the following Lemma Appendix A.1 and Lemma Appendix A.2.

Lemma Appendix A.1.

$$\sigma_j^\top e^{-\frac{\beta}{m}\mathcal{H}_c} \sigma'_j = \exp\left[-\frac{\beta}{m}E(\sigma_j)\right] \delta(\sigma_j, \sigma'_j), \quad (\text{A.5})$$

where $\delta(\sigma_j, \sigma'_j) = 1$ if $\sigma_j = \sigma'_j$ and $\delta(\sigma_j, \sigma'_j) = 0$ otherwise.

Proof. By the definition, $e^{-\frac{\beta}{m}\mathcal{H}_c}$ is diagonal with $[e^{-\frac{\beta}{m}\mathcal{H}_c}]_{kk} = E(\sigma^{(k)})$, and σ_j and σ'_j are binary indicator vectors, i.e. only one element in σ_j is one and the others are zero. Thus, the above lemma holds. \square

Lemma Appendix A.2.

$$\begin{aligned} \sigma_j'^\top e^{-\frac{\beta}{m}\mathcal{H}_q} \sigma_{j+1} &= \left[\sinh\left(\frac{\beta}{m}\Gamma\right)\right]^{2N} \\ &\quad \exp\left[\sum_{i=1}^N \sum_{n=1}^N \delta(\tilde{\sigma}'_{j,i,n}, \tilde{\sigma}_{j+1,i,n}) \log \coth\left(\frac{\beta}{m}\Gamma\right)\right]. \end{aligned} \quad (\text{A.6})$$

Proof. Using $(A \otimes B)(C \otimes D) = (AC) \otimes (BD)$ and

⁴If $A_1 A_2 = A_2 A_1$, then $e^{A_1+A_2} = e^{A_1} e^{A_2}$.

$e^{A_1+A_2} = e^{A_1}e^{A_2}$ when $A_1A_2 = A_2A_1$, we find,

$$\begin{aligned}\sigma_j'^\top e^{-\frac{\beta}{m}\mathcal{H}_q}\sigma_{j+1} &= \sigma_j'^\top e^{-\frac{\beta}{m}\{-\Gamma\sum_{i=1}^N\sum_{n=1}^N\sigma_{i,n}^x\}}\sigma_{j+1} \\ &= \sigma_j'^\top \left(\bigotimes_{i=1}^N \bigotimes_{n=1}^N e^{\frac{\beta}{m}\Gamma\sigma_{i,n}^x} \right) \sigma_{j+1} \\ &= \prod_{i=1}^N \prod_{n=1}^N \tilde{\sigma}_{j,i,n}'^\top e^{\frac{\beta}{m}\Gamma\sigma_{j+1,i,n}^x}.\end{aligned}\quad (\text{A.7})$$

Note that $\sigma_j = \bigotimes_{i=1}^N \bigotimes_{n=1}^N \tilde{\sigma}_{j,i,n}$.

$$\begin{aligned}e^{\frac{\beta}{m}\Gamma\sigma^x} &= \sum_{l=0}^{\infty} \frac{1}{l!} \left(\frac{\beta}{m}\Gamma\sigma^x \right)^l \\ &= \left[1 + \frac{1}{2!} \left(\frac{\beta}{m}\Gamma \right)^2 + \dots \right] \mathbb{E} \\ &\quad + \left[\frac{\beta}{m}\Gamma + \frac{1}{3!} \left(\frac{\beta}{m}\Gamma \right)^3 + \dots \right] \sigma^x \\ &= \cosh \left(\frac{\beta}{m}\Gamma \right) \mathbb{E} + \sinh \left(\frac{\beta}{m}\Gamma \right) \sigma^x.\end{aligned}\quad (\text{A.8})$$

Substituting Eq.(A.8) into Eq.(A.7), we have Eq.(A.6) because

$$\begin{aligned}\tilde{\sigma}_{j,i,n}'^\top &\left[\cosh \left(\frac{\beta}{m}\Gamma \right) \mathbb{E} + \sinh \left(\frac{\beta}{m}\Gamma \right) \sigma^x \right] \tilde{\sigma}_{j+1,i,n} \\ &= \cosh \left(\frac{\beta}{m}\Gamma \right) \delta(\tilde{\sigma}_{j,i,n}', \tilde{\sigma}_{j+1,i,n}) \\ &\quad + \sinh \left(\frac{\beta}{m}\Gamma \right) (1 - \delta(\tilde{\sigma}_{j,i,n}', \tilde{\sigma}_{j+1,i,n}))\end{aligned}\quad (\text{A.9})$$

□

Using Lemma Appendix A.1 and Lemma Appendix A.2 into Eq.(A.4),

$$\begin{aligned}\sigma_1^\top &\left(e^{-\frac{\beta}{m}\mathcal{H}_c} e^{-\frac{\beta}{m}\mathcal{H}_q} \right)^m \sigma_1 \\ &= \sum_{\sigma_2} \dots \sum_{\sigma_m} \prod_{j=1}^m \exp \left[-\frac{\beta}{m}E(\sigma_j) \right] \left[\sinh \left(\frac{\beta}{m}\Gamma \right) \right]^{2N} \\ &\quad \exp \left[\sum_{i=1}^N \sum_{n=1}^N \delta(\tilde{\sigma}_{j,i,n}', \tilde{\sigma}_{j+1,i,n}) \log \coth \left(\frac{\beta}{m}\Gamma \right) \right].\end{aligned}\quad (\text{A.10})$$

and from Eq.(A.3), we have shown,

$$p_{\text{QA}}(\sigma_1; \beta, \Gamma) \approx \sum_{\sigma_2} \dots \sum_{\sigma_m} p_{\text{QA-ST}}(\sigma_1, \sigma_2, \dots, \sigma_m; \beta, \Gamma). \quad (\text{A.11})$$

The same relation holds for $\sigma_2, \sigma_3, \dots, \sigma_m$.

Appendix B. Derivation of Eq.(4)

Since

$$\begin{aligned}p_{\text{QA}}(\tilde{\sigma}_{j,i} | \{\sigma_d\}_{d=1}^m \setminus \{\tilde{\sigma}_{j,i}\}; \beta, \Gamma) \\ \propto e^{-\frac{\beta}{m}E(\sigma_j)} e^{f(\beta, \Gamma)(s(\sigma_{j-1}, \sigma_j) + s(\sigma_j, \sigma_{j+1}))},\end{aligned}\quad (\text{B.1})$$

we have

$$\begin{aligned}p_{\text{QA}}(\tilde{\sigma}_{j,i} | \{\sigma_d\}_{d=1}^m \setminus \{\tilde{\sigma}_{j,i}\}; \beta, \Gamma) &\propto \\ &\begin{cases} \left(\frac{\sum_{n=1, n \neq i}^N \delta(\tilde{\sigma}_{j,i,n}, (1, 0)^\top)}{N-1+\alpha} \right)^{\frac{\beta}{m}} e^{\tilde{s}(\tilde{\sigma}_{j-1,i}, \tilde{\sigma}_{j,i}, \tilde{\sigma}_{j+1,i})f(\beta, \Gamma)}, & \text{if } \tilde{\sigma}_{j,i} \in \tilde{\Sigma}_{j,i} \setminus \{\rho_{2N-1}^{(i)}\}, \\ \left(\frac{\alpha}{N-1+\alpha} \right)^{\frac{\beta}{m}}, & \text{if } \tilde{\sigma}_{j,i} = \rho_{2N-1}^{(i)}, \\ 0 & \text{if } \tilde{\sigma}_{j,i} \notin \tilde{\Sigma}_{j,i}, \end{cases}\end{aligned}\quad (\text{B.2})$$

$$\tilde{s}(\tilde{\sigma}_{j-1,i}, \tilde{\sigma}_{j,i}, \tilde{\sigma}_{j+1,i})$$

$$= \sum_{n=1, n \neq i}^N \delta(\tilde{\sigma}_{j-1,i,n}, \tilde{\sigma}_{j,i,n}) + \sum_{n=1, n \neq i}^N \delta(\tilde{\sigma}_{j,i,n}, \tilde{\sigma}_{j+1,i,n}). \quad (\text{B.3})$$

This is easy to understand when you consider the meaning of $\tilde{s}(\tilde{\sigma}_{j-1,i}, \tilde{\sigma}_{j,i}, \tilde{\sigma}_{j+1,i})$ in multiple CRPs. $\tilde{s}(\tilde{\sigma}_{j-1,i}, \tilde{\sigma}_{j,i}, \tilde{\sigma}_{j+1,i})$ indicates the number of customers who share tables with the i -th customer in the $j-1$ -th and $j+1$ -th CRPs. Therefore, Eq.(4) is derived as another formulation of Eq.(B.2). Note that Eq.(4) is the approximation of Eq.(13).

Appendix C. Details of Network Model

In this section, we explain the Newman network model. \mathcal{V} is the vertex set. v is a vertex; i.e., $v \in \mathcal{V}$. V is the number of vertices. K is the number of classes. Suppose that the vertices fall into K classes with probability π , where π_k is the probability that a vertex is assigned to class k . Vertex i belongs to class k , indicated by $z_i = k$. Each class has a probability ϕ_{kv} that a link from a particular vertex in class k connects to vertex v . A link from vertex i to vertex v is indicated by $\ell_i = v$. Each vertex links to other vertices in accordance

with ϕ . That is, vertex i links to vertex v in accordance with $\phi_{z_i v}$. The generation process for link ℓ_i is represented by $\ell_i \sim \text{Multi}(\phi_{z_i})$, $z_i \sim \text{Multi}(\pi)$, where $\phi_{z_i} = (\phi_{z_i 1}, \phi_{z_i 2}, \dots, \phi_{z_i V})$, and $\text{Multi}(\cdot)$ is a multinomial distribution.

Suppose that ϕ_t is distributed in accordance with the Dirichlet distribution $H(\tau)$; i.e., $\phi_k \sim H(\tau)$, where τ is a parameter of the Dirichlet distribution. G is a random probability measure over ϕ : $G \sim \text{DP}(\alpha, H(\tau))$, where $\text{DP}(\cdot)$ indicates the Dirichlet process (DP), α is the DP concentration parameter that is equal to the hyper parameter of the CRP, and H is the base measure, which is the Dirichlet distribution here. The generation process for link ℓ_i is represented by $\ell_i \sim \text{Multi}(\phi_{z_i})$, $\phi_{z_i} \sim G$.

Here, we define A as an adjacency matrix with elements $A_{iv} = 1$ if there is an edge from i to v ; otherwise $A_{iv} = 0$. The probability of z_i given $z_{-i} = \{z\} \setminus z_i$ and adjacency matrix A is

$$p(z_i = k | A, z_{-i}; \alpha) \propto p(A_i | z_i = k, A_{-i}, z_{-i}) p(z_i = k | z_{-i}; \alpha), \quad (\text{C.1})$$

where $A_i = (A_{i1}, A_{i2}, \dots, A_{iV})$, and $A_{-i} = A \setminus A_i$.

We can calculate the probability of Eq.(C.1) as follows.

$$p(A_i | z_i = k, A_{-i}, z_{-i}) = \frac{g(V\tau + \sum_{u \neq i} \sum_v A_{uv} z_u^k)}{g(V\tau + \sum_u \sum_v A_{uv} z_u^k)} \prod_v \frac{g(\tau + \sum_u A_{uv} z_u^k)}{g(\tau + \sum_{u \neq j} A_{uv} z_u^k)}, \quad (\text{C.2})$$

where $g(\cdot)$ is the gamma function, and z_i^k indicates 1 if $z_i = k$; otherwise, it indicates 0.

$$p(z_i = k | z_{-i}; \alpha) = \begin{cases} \frac{N_k^{-i}}{V - \frac{1}{\alpha} + \alpha} & (\text{if } k \text{ previously used.}) \\ \frac{1}{V - 1 + \alpha} & (\text{if } k \text{ is new.}) \end{cases} \quad (\text{C.3})$$

where N_k^{-i} is the number of vertices except vertex i assigned to class k ; i.e., $N_k^{-i} = \sum_{v \neq i} z_v^k$. We can adapt a Gibbs sampler for estimating z_i by using Eq. (C.1).

References

Aldous, D., 1985. Exchangeability and related topic. *Ecole d'Ete de Probabilités de Saint-Flour XIII-1983*, 1–198.
 Antoniak, C., 1974. Mixtures of Dirichlet processes with applications to bayesian nonparametric problems. *The Annals of Statistics* 2, 1152–1174.

Blei, D.M., Jordan, M.I., 2005. Variational inference for dirichlet process mixtures. *Bayesian Analysis* 1, 121–144.
 DaumeIII, H., 2007. Fast search for dirichlet process mixture models, in: *Proceedings of Artificial Intelligence and Statistics*.
 Dempster, A., Laird, N., Rubin, D., 1977. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society series B* 39, 1–38.
 Farhi, E., Goldstone, J., Gutmann, S., J. Lapan, A.L., Preda, D., 2001. A quantum adiabatic evolution algorithm applied to random instances of an np-complete problem. *Science* 292, 472–476.
 Frey, B.J., Dueck, D., 2007. Clustering by passing messages between data points. *Science* 315, 972–976.
 Geman, S., Geman, D., 1984. Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images. *IEEE Pattern Analysis and Machine Intelligence*, 6, 721–741.
 Kadowaki, T., Nishimori, H., 1998. Quantum annealing in the transverse ising model. *Physical Review E* 58, 5355–5363.
 Kemp, C., Tenenbaum, J.B., Griffiths, T.L., Yamada, T., Ueda, N., 2006. Learning systems of concepts with an infinite relational model, in: *AAAI*.
 Kirkpatrick, S., Gelatt, C.D., Vecchi, M.P., 1983. Optimization by simulated annealing. *Science* 220, 671–680.
 Kurihara, K., Tanaka, S., Miyashita, S., 2009. Quantum annealing for clustering, in: *Proceedings of the 25th Conference on Uncertainty in Artificial Intelligence*.
 Kurihara, K., Welling, M., Teh, Y.W., 2007. Collapsed variational dirichlet process mixture models, in: *IJCAI*, pp. 2796–2801.
 L. Xu, A.K., Oja, E., 1993. Rival penalized competitive learning for clustering analysis, rbf net, and curve detection. *IEEE Transaction of Neural Network* 4, 636–649.
 Lloyd, S., 1996. Universal quantum simulators. *Science* 273, 1073–1078.
 Neal, R.M., 2000. Markov chain sampling methods for dirichlet process mixture models. *Journal of Computational and Graphical Statistics* 9, 249–265.
 Newman, M.E.J., Leicht, E.A., 2007. Mixture models and exploratory analysis in networks, in: *Proceedings of National Academy of Sciences of the United States of America*.
 Nielsen, M., Chuang, I., 2000. *Quantum Computation and Quantum Information*. Cambridge University Press.
 Santoro, G.E., Martoňák, R., Tosatti, E., Car, R., 2002. Theory of quantum annealing of an Ising spin glass. *Science* 295, 2427–2430.
 Sato, I., Kurihara, K., Tanaka, S., Nakagawa, H., Miyashita, S., 2009. Quantum annealing for variational bayes inference, in: *Proceedings of the 25th Conference on Uncertainty in Artificial Intelligence*.
 Suzuki, M., 1976. Relationship between d -dimensional quantum spin systems and $(d+1)$ -dimensional Ising systems – equivalence, critical exponents and systematic approximations of the partition function and spin correlations –. *Progress of Theoretical Physics* 56, 1454–1469.
 Trotter, H.F., 1959. On the product of semi-groups of operators. *Proceedings of the American Mathematical Society* 10, 545–551.
 Wang, C., Lai, J., 2011. Energy based competitive learning. *Neurocomputing* 74, 2265–2275.