Tensor network renormalization for q-state clock model

Jun-Han Huang,[1] Zai-Zhou Xin,[1] and Jia-Hao Liu[1]

[1] *Department of Physics, Fudan University, Shanghai 200433, China*

(Dated: December 25, 2022)

The q-state clock model is an excellent platform to study the BKT transitions. With the recently developed tensor network approaches, studying the q-state clock model becomes possible. In this paper, we perform different tensor network algorithms on the q-state clock model to study the critical behaviours, such as the critical points, central charge and scaling dimension. The results are consistent with previous works.

I. INTRODUCTION

Understanding emergent phenomena in condensed matter systems lies at the heart of physics, such as phase transitions and critical phenomena. Two typical types of phase transitions include the Ginzburg-Landau phase transition driven by order parameters with symmetry breaking and the Berezinskii-Kosterlitz-Thouless (BKT) transitions driven by topological defects [1]. The BKT transition is usually presented in the XY model, providing an example for beyond Landau's symmetry breaking paradigm. Recently, scientists found that BKT transitions can occur in the discrete XY model, also known as the q-state clock model or vector Potts model. It's also pointed out that this model undergoes an intermediate state with two critical points when the possible orientations $q \geq 5$ [2]. However, accurate determination of the two critical temperatures for the cases $q \geq 5$ also remains a challenging problem [3, 4]. Since the critical points are hard to determine, especially with large q, by using usual Monte-Carlo methods [5], other numerical approaches must be used to study this system.

Recently, tensor network has become a powerful theoretical and numerical tool for studying condensed matter systems [6, 7]. In statistical physics, the partition function can be expressed in terms of tensor networks, thus if we can calculate the tensor network, it's quite simple to obtain other physical quantities. Hence, the problem is reduced to the contraction of multidimensional tensor network, and many algorithms have been proposed to implement tensor contraction approximations [8, 9]. However, when applying tensor network methods to the q-state clock model especially for large q, high accuracy tensor network algorithms are badly needed, since this model is strongly entangled in short-range, which is irrelevant when studying critical properties. Moreover, truncation error must be reduced in order to accurately determine the critical points.

In this paper, we perform the tensor network algorithms on the q-state clock model to study the critical behaviours, such as the critical point, scaling dimension and central charge. We use different tensor network algorithms (LN-TNR, TEFR, EV-TNR and Loop-TNR) to study the model and compare the results.

II. Q-STATE CLOCK MODEL

A. Partition Function and Transfer Matrix

The Hamiltonian of the q-state clock model is given by

$$H = -J \sum_{\langle ij \rangle} \cos\left(\theta_i - \theta_j\right) \tag{1}$$

where $\theta_i = 2\pi i/q$, and $i \in \{1, 2, \cdots, q\}$. Let $\theta_{ij} = \theta_i - \theta_j$, we have the partition function

$$Z = \sum_{\{\theta\}} \exp\left(\beta J \sum_{\langle ij \rangle} \theta_{ij}\right) \tag{2}$$

For $q = 2$, the model is reduced to the Ising model. In the limit $q \to \infty$, the q-state clock model is equivalent to the XY model. We note that, the partition function can be rewritten into the form of tensor:

$$Z = \mathrm{tTr} \otimes T \tag{3}$$

Consider 2-dimension square lattice, we can construct the tensor from the interaction between four adjacent spins (seen in Fig. 1). Thus the 4-rank tensor T_{ijkl}:

$$T_{ijkl} = \exp \beta J \left(\cos\theta_i j + \cos\theta_j k + \cos\theta_k l + \cos\theta_l i\right) \tag{4}$$

where $i, j, k, l \in \{1, 2, \cdots, q\}$. And tensor T can be calculated by summing over the whole tensor network. In this way, we have the partition function

$$Z = \mathrm{tTr} \otimes T = \sum_{ijkl\ldots.} T_{jfei} T_{hgjk} T_{qklr} T_{lits} \cdots \tag{5}$$

B. Tensor Network Renormalization

Calculating the tTr is a hard problem in high dimensions, because there are so many tensors in the network.A simple idea to accelerates this calculation is illustrated in Fig. 2. According to the approximate method first introduced by Levin and Nave[11], we find a "bigger" tensor T' to replace the original "smaller" tensors. As a result, we can then express the trace as

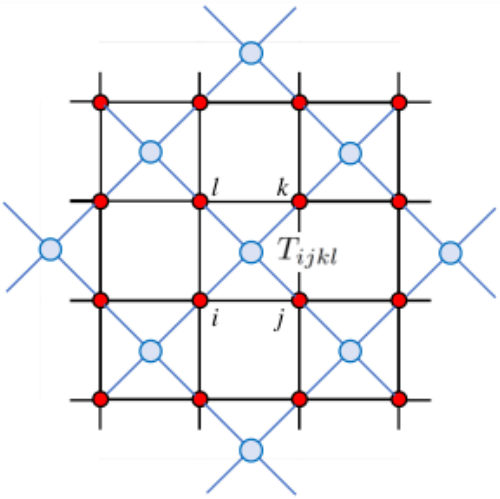

FIG. 1: Tensor network representation of q-state clock model on square lattice[10]

tTr$[T \otimes T \dots] \approx tTr[T'' \otimes T'' \dots]$,where we only need to deal with less tensors contained in the new network. The cost of reducing tensors is that each has a higher dimension, but it's difficult to store and call the huge tensors in the high-order iterations if we don't take any approximation. In the TRG algorithm, we use the low-rank approximation to reduce the dimension of the new tensor to the desired value.

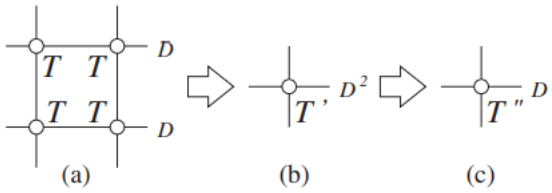

FIG. 2: Renormalization procedure of tensor network[12]

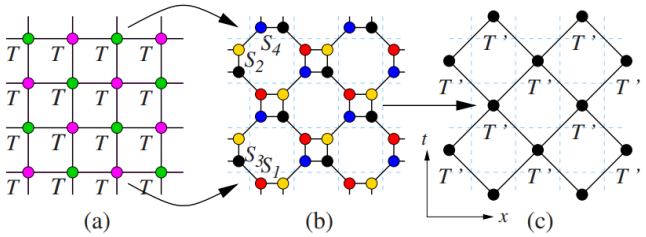

FIG. 3: Implementation on 2D square lattice[12]

The actual implement of the TRG algorithm on a 2d square lattice is a little more complicated.As shown in FIG.3, to construct a new tensor T', We divide the lattice into a purple sublattice and a green sublattice. By means of singular value decomposition, we can decompose the tensor on purple sublattice into two third-order tensors. That is, $T_{ruld} = \sum_{s=1}^{D^2} S_{1uls} S_{3drs}$. And likewise for the green sublattice, we have $T_{ruld} = \sum_{s=1}^{D^2} S_{2lds} S_{4rus}$. We haven't used an approximation here, r, u, l, d all have D values and s has D^2 values.

It is easy to find that after accurate decomposition, the tensor network becomes the form shown in 3??b) and a new fourth order tensor formed from four third order tensors(S) can be regarded as the new tensor T' with a dimension of D^2. The dimension of T' is determined by the form of S. If we choose to keep only the largest first D_{cut} eigen values in SVD, which we think can approximately Keep the trace before and after decomposition unchanged. Then we can make s runs over D_{cut} values, which is to say, the dimension of T' is D_{cut}.

Through the procedure we mentioned above, we can get T^{i+1} from T^i,which is an iteration step of TRG algorithm.By repeating the above process, tensor network renormalization is realized.

C. Critical Properties

We use TRG(LN-TNR) Algorithm to simulate the transition of the 2-dimension q-state clock model. Usually, we use the second derivative of the free energy(the heat capacity) to find the critical point. After renormalization, we can get the partition function Z, and then calculate the free energy F and heat capacity $\partial^2 F/\partial T^2$(See in Fig. 4). Obviously the capacity has a peak at the

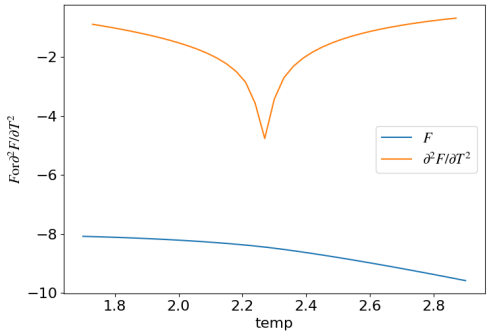

FIG. 4: The free energy and its second derivative for $q = 2$ model

certain temperature. The temperature of the peak is at 2.26 ± 0.02 close to the theoretical value of 2-dimension model is $2/\left(\ln\left(\sqrt{2}+1\right)\right) \approx 2.269$. However, in this article we use the gauge invariant quantity χ introduced in Ref. [12]:

$$\chi = \frac{\left(\sum_{ij} T_{ijij}\right)^2}{\sum_{ijkl} T_{ijkl} T_{klij}} \tag{6}$$

The gauge invariant quantity χ has a good characteristic. It converges to q in ordered phase, and converges to 1 in disordered phase. To help understand this, we take $q = 2$ as an example. For Ising model, the tensor is given by

$$\begin{aligned} &T^{Ising}_{1212} = e^{-4\beta}, \quad T^{Ising}_{2121} = e^{-4\beta}, \\ &T^{Ising}_{1111} = e^{4\beta}, \qquad T^{Ising}_{2222} = e^{4\beta}, \\ &\text{others} = 1. \end{aligned} \tag{7}$$

We performed TRG described in II B. After several iterations, the tensor converges to a fixed-point tensor. For high temperature, the fixed-point tensor has the form:

$$T_{1111} = 1, \quad \text{others} = 0. \tag{8}$$

We call tensor like this the trivial tensor T^{TRI}. For low temperature, the fixed-point tensor has the form:

$$T_{1111} = 1, \quad T_{2222} = 1, \text{others} = 0. \tag{9}$$

We can rewrite the tensor in the form $T^{\text{Z}_2} = T^{\text{TRI}} \oplus T^{\text{TRI}}$. This symmetry is because in the ordered phase, it is the same for spins all up and spins all down. And T^{Z_2} happens to be equivalent as the initial tensor in the zero-temperature limit($\beta = \infty$). Thus we can regard the flow of the tensor network as the flow of the temperature $\beta \to \infty$. In this view, we expect there's the same result for high temperature. However, the fixed-point tensor has a form:

$$\widetilde{T}^{\text{TRI}}_{ruld} = \frac{1}{4}, \quad r, u, l, d = 1, 2 \tag{10}$$

which is not the same as T^{TRI} in Eq. 8. However, the two tensor will work out the same tensor trace because of the similarity between them,

$$T'_{r'u'l'd'} = \left(A^{-1}\right)_{l'l} \left(B^{-1}\right)_{u'u} T_{ruld} A_{rr'} B_{dd'} \tag{11}$$

where A and B are orthogonal matrices. And it is easy to check that $A = B = R$

$$R = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \tag{12}$$

To sum up, the gauge invariant quantity χ is a good value to evaluate the phase transition.

After 20 iterations of LN-TNR with $D_{cut} = 24$, we get how χ varies with temperature for $q = 2, 3, 4$(See in Fig.5, 6, 7). We can see for $q = 2, 3$, the simulation critical temperature is intrinsically close to the theoretical value. For $q = 4$, the simulation critical temperature is a little bigger than the theoretical value, which means the LN-TNR algorithm can be so accurate.

The error still remains for $q = 5$ model. There are two critical points for $q = 5$ model. After adjusting the D_{cut} and iteration number, we correctly get the middle phase for $q = 5$(See in Fig. 8). However, the position of the critical point cannot be exactly decided. Thus

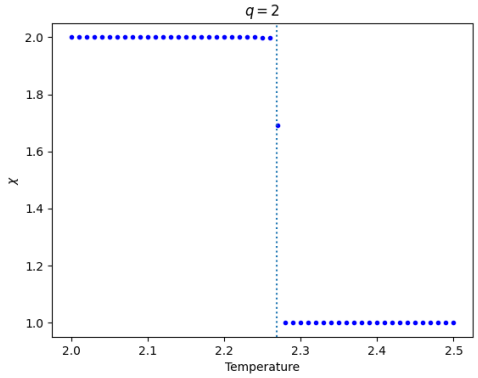

FIG. 5: The invariant quantity χ as a function of temperature for $q = 2$ model. The vertical line is the theoretical value of the critical temperature $2/\left(\ln\left(\sqrt{2}+1\right)\right) \approx 2.2692$

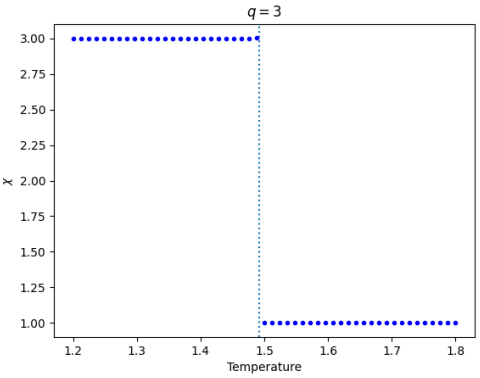

FIG. 6: The invariant quantity χ as a function of temperature for $q = 3$ model. The vertical line is the theoretical value of the critical temperature $3/\left(2\ln\left(\sqrt{3}+1\right)\right) \approx 1.4925$

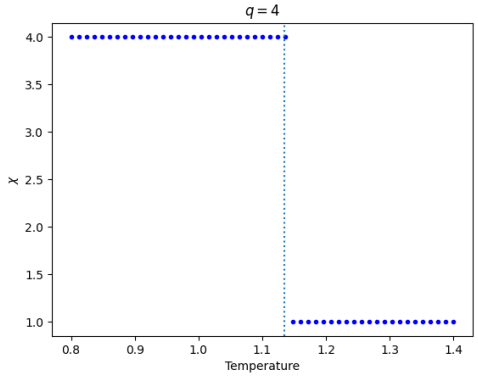

FIG. 7: The invariant quantity χ as a function of temperature for $q = 4$ model. The vertical line is the theoretical value of the critical temperature $2/\left(\ln\left(\sqrt{2}+1\right)\right) \approx 1.1345$

we studied χ as a function of iterations for $q6$. Collect χ during 40 iterations with $D_{cut} = 42$(See in Fig. 9). First, the invariant quantity converged to a stable number about 4.7, which meant this temperature should. As the iteration times increased, χ jumped out of the middle phase into the ordered phase. When iteration times was higher than 30, χ jumped into another unknown state. The jump of χ between states happened because of the truncation error in the SVD decomposition. And the unknown state may predicted some unexpected fixed-point tensor. Thus we need some more precise and optimised algorithm to reduce the unexpected fixed-point and minimize the truncation error. We will talk about them in III.

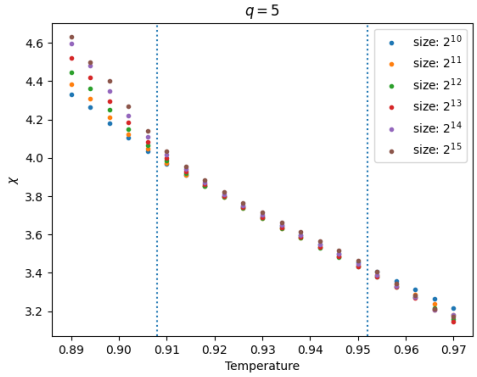

FIG. 8: The invariant quantity χ as a function of temperature for $q = 5$ model

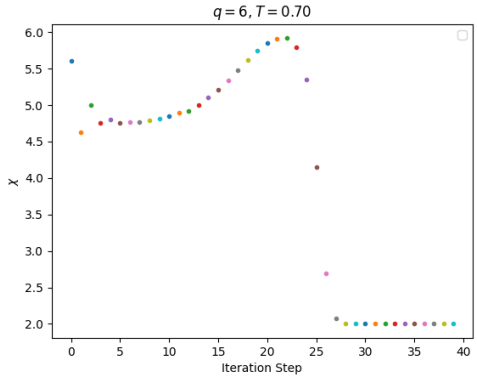

FIG. 9: The invariant quantity χ as a function of iteration times for $q = 6$ model

The principle of conformal invariance is strong at critical point,a single dimensionless number c (central charge of the virasoro algebra) can characterize the universality classes. According to conformal field theory, we can calculate the central charge and scaling dimensions from following formula(see Ref.[12]):

$$[tTr(T_{inv}^{i})^{N_i}] = \sum_{n=0} e^{-2\pi[\delta - c/12]Im(\tau^i)} \tag{13}$$

Among themT_{inv}^{i} is the fixed point tensor,tau is a complex number describes the lattice shape change during rescaling, δ and c are scaling dimensions and central charge.

As for $q = 2$,The theory gives the central charge and the lowest nonzero scaling dimension(at critical point) as 0.5 and 0.125.The results of our calculations with TRG algorithm is shown in FIG.10.We can see that our calculation results are close to the theoretical result as the iteration steps increase at first.However,if the iteration steps are too big, the truncation error in approximate SVD operation will cause the results to deviate. This Phenomena tell us that we need to make appropriate iteration steps choices to make the best results with TRG.This is the limitation of TRG.

It is worth noting that, as central charge is only nonzero when correlation length diverges, plotting central charge as a function of temperature is a good way to discover phase transition. As shown in FIG.11, we use central charge to determining phase transition point which Consistent with results using χ

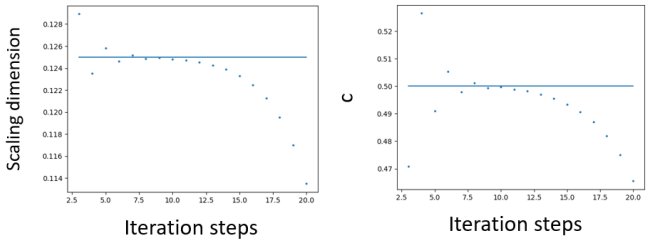

FIG. 10: central charge to determining phase

III. IMPROVED TENSOR NETWORK RENORMALIZATION ALGORITHM

A. TEFR Algorithm

A large class of tensors which can be fixed point of the SVDTRG flow takes the following form:

$$\begin{aligned} T_{ruld}(M^1, M^2, M^3, M^4) &= T_{r_1 r_2, u_2 u_1, l_2 l_1, d_1 d_2} \\ &= M^1_{r_2 u_1} M^2_{u_2 r_1} M^3_{l_2 d_1} M^4_{d_2 r_1} \end{aligned} \tag{14}$$

r_1, r_2 represents the index r,This corner double-line tensor is called CDL tensor. Network constituted by this kind of fixed pointed tensor is shown in FIG. 12 We can see there are lots of disconnected squares in the network, which means the degrees of freedom form cluster and thus interaction between clusters are limited. This makes the

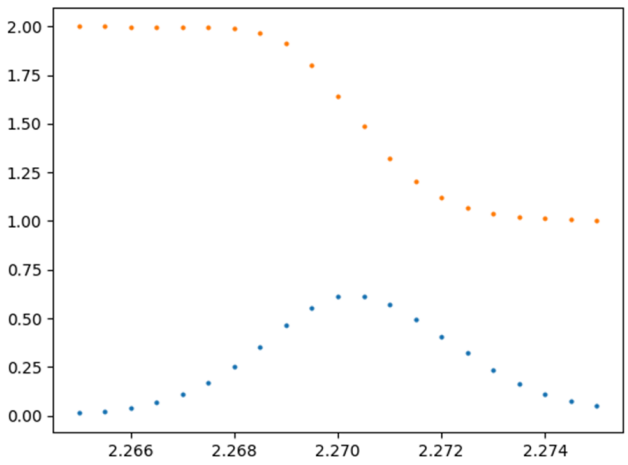

FIG. 11: The central charge and lowest nonzero scaling dimension at critical point as a function of the iteration steps

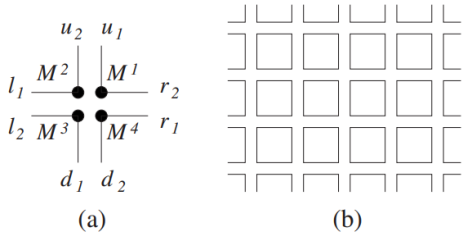

FIG. 12: (a)a CDL tensor.(b)CDL tensors constitute a network

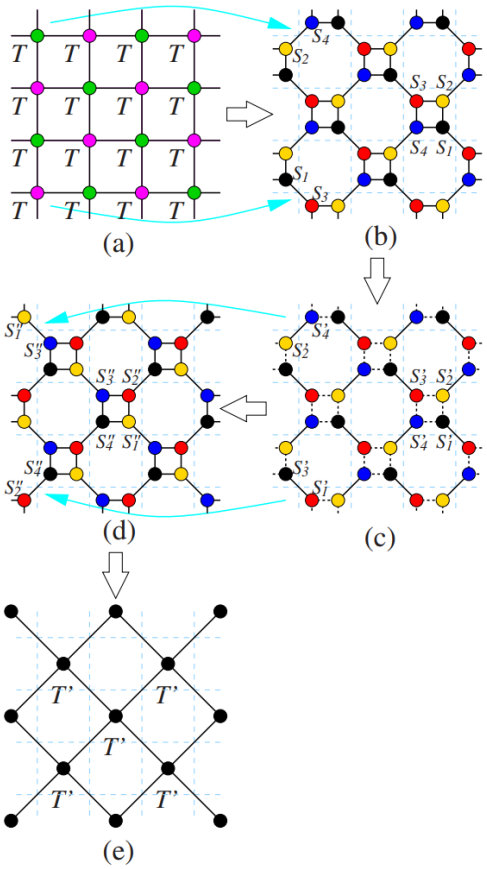

FIG. 13: The graphical representation of TEFR algorithm.

CDL fixed point describe a trivial phase with out long-range entanglement.

To overcome this flaw in the TRG algorithm, there is a improved algorithm called TEFR. As shown in FIG.13, We can see that it is basically consistent with TRG algorithm, except that the process from (c) to (d) is a new content. n this step, the iterative process(details see Ref. [12]) is used to reduce the corresponding degrees of freedom (entanglement reduction) on each edge of the small rectangle(As shown in the FIG. 10, dotted line is changed to a solid line).

It should be noted that we did not try this algorithm because it does not solve the most significant truncation error problem in TRG.

B. EV-TNR Algorithm

Recently Ref. [13] introduced a method to successfully remove short-range correlations from the partition function at each coarse-graining step, even in critical systems. This approach is referred to as Evenbly-Vidal TNR (EV-TNR).

The EV-TNR algorithm begins with an entanglement filtering TNR transformation. The key idea is to insert two projectors (also called disentanglers) u and $u^\dagger$ into the tensor network, with $uu^\dagger = I$. The projectors can reduce short-range correlations and does not change the partition function. However when the insertion combines two indices into a single one, as shown in Fig. 14(c), it will introduce a truncation error δ into the tensor network (Fig. 14(d)). By carefully choosing the projectors, we can minimize $\|\delta\|$ and then resulting tensor network will be a good approximation.

The graphical representation of TNR transformation is shown in Fig. 15. In Fig. 15(a), projectors are inserted into the tensor network. The Fig. 15(b) step is contracting indices to generate $B^{(s)}$ and $C^{(s)}$. The Fig. 15(c) is the singular value decomposition. Details of choosing projectors can be found in Ref. [14].

Since the short-range correlations are reduced through the entanglement filtering step, the EV-TNR algorithm can significantly improve the accuracy in the same D_{cut} as LN-TNR. To demonstrate the advantage of EV-TNR algorithm, we calculate the scaling dimension of $q = 2$ clock model (i.e., the Ising model) and compare it to the result of LN-TNR model, as shown in Fig. 16. For

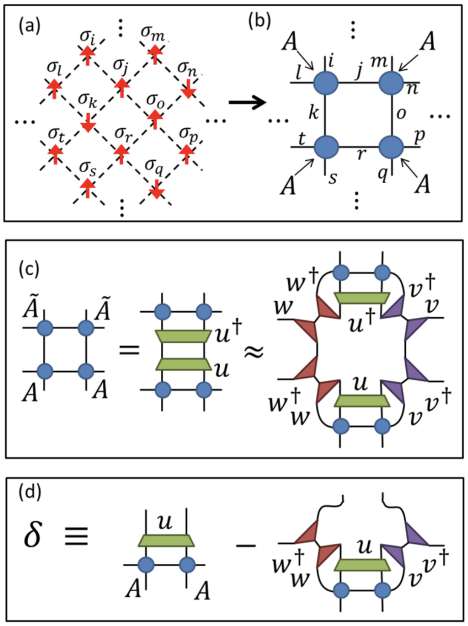

FIG. 14: The entanglement filtering step.

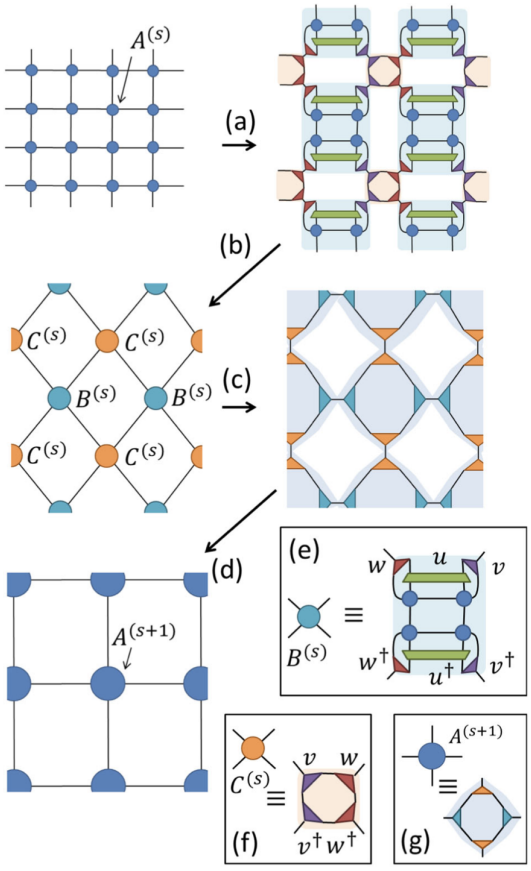

FIG. 15: The graphical representation of TNR transformation.

small system size, the two algorithms give the same result, while for large system size, the LN-TNR results deviate from the theoretical values.

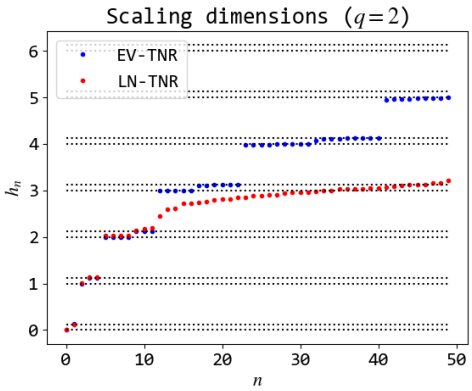

FIG. 16: Scaling dimension calculated by EV-TNR and LN-TNR algorithm, respectively.

C. Loop-TNR Algorithm

We spent a lot of time on the Loop-TNR algorithm [15] but failed. Here we will introduce Loop-TNR algorithm and discuss about our problems.

The Loop-TNR algorithm contains three main steps. The first one is a entanglement filtering step. This idea was first introduced in Ref. [12], and was showed that it can reduce corner double line (CDL) tensors and generate a canonical gauge. This step is exact by inserting projectors between the tensors. See Appendix A for details of entanglement filtering.

The next step is to optimize the tensors on a loop. To reduce the bond dimensions when deforming the square lattice, they defined the following cost function shown in Fig. 17 which is a quadratic function for parameters associated with one tensor $\mathbf{S}_i$

$$\begin{aligned} f &= \|\mathbf{T}_1\cdot\mathbf{T}_2\cdot\mathbf{T}_3\cdot\mathbf{T}_4 - \mathbf{S}_1\cdot\mathbf{S}_2\cdot\mathbf{S}_3\cdot\mathbf{S}_4\cdot\mathbf{S}_5\cdot\mathbf{S}_6\cdot\mathbf{S}_7\cdot\mathbf{S}_8\|^2 \\ &= \|\,|\Psi_A\rangle - |\Psi_B\rangle\,\|^2 \\ &= \langle\Psi_A|\Psi_A\rangle + \langle\Psi_B|\Psi_B\rangle - \langle\Psi_A|\Psi_B\rangle - \langle\Psi_B|\Psi_A\rangle \\ &= \mathcal{C} + \mathbf{S}_i^\dagger\mathcal{N}_i\mathbf{S}_i - \mathcal{W}_i^\dagger\mathbf{S}_i - \mathbf{S}_i^\dagger\mathcal{W}_i \end{aligned} \tag{15}$$

which is quite different from the LN-TNR algorithm or EV-TNR alorithm, where $f_1 = \|\mathbf{T}_1 - \mathbf{S}_1\cdot\mathbf{S}_2\|^2$ and $f_2 = \|\mathbf{T}_2 - \mathbf{S}_3\cdot\mathbf{S}_4\|^2$. For each $\mathbf{S}_i$ the minimum of $f(\mathbf{S}_i)$ can be found by solving the following equation

$$\mathcal{N}_i\mathbf{S}_i = \mathcal{W}_i \tag{16}$$

and we can optimize $\mathbf{S}_i$ according to Eq. 16. By sweeping back and forth we can find a minimum of f. The final step is the same coarse-graining procedure as LN-TNR algorithm[11].

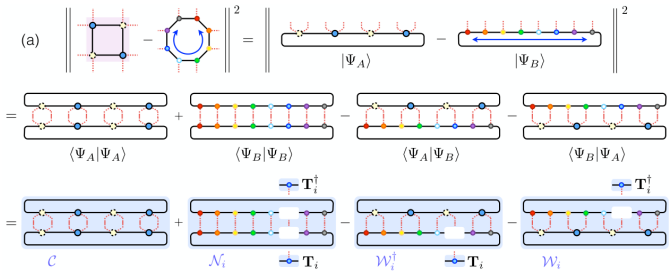

FIG. 17: The cost function [15], corresponds to Eq. 15.

Our problem is mainly with the second step. When calculating $\mathcal{N}_i$ and $\mathcal{W}_i$, we need to calculate the contraction of multiple tensors. However this step is high on memory cost, and is hard to calculate with our personal computer under the condition of $D_{cut} = 36$ as in Ref. [10]. We used the following contraction procedure but still cannot significantly reduce the virtual memory cost when D_{cut} is large. We calculate $S_i \cdot \mathbf{S}_{i+1}$ first, which is a 4-order tensor, as shown in Fig. 18(a). Then calculate $\mathbf{S}_i \cdot \mathbf{S}_{i+1} \cdot \mathbf{S}_i^\dagger \cdot \mathbf{S}_{i+1}^\dagger$ for $\mathcal{N}_i$ (Fig. 18(b)) and $\mathbf{S}_i \cdot \mathbf{S}_{i+1} \cdot \mathbf{T}_i$ for $\mathcal{W}_i$ (Fig. 18(c) and Fig. 18(d)). These tensors are 4 or 5-order tensors. Thus the order of tensor contraction is within 5. Notice that these tensor calculations can be simply performed on GPU, so our future plan contains introducing GPU algorithms, like CUDA.

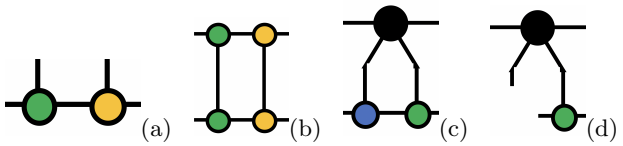

FIG. 18: Rules of tensor contraction when calculating $\mathcal{N}_i$ and $\mathcal{W}_i$.

This problem can also probably be solved by considering the C_4 rotational symmetry, which reduces the cost function into

$$f = \|\mathbf{T}_1 \cdot \mathbf{T}_2 \cdot \mathbf{T}_3 \cdot \mathbf{T}_4 - \mathbf{M} \cdot \mathbf{M} \cdot \mathbf{M} \cdot \mathbf{M}\|^2 \tag{17}$$

where $\mathbf{M} = \mathbf{S}_1 \cdot \mathbf{S}_2$. Thus we can use the conjugate gradient method introduced in Ref. [16] to minimize the cost function.

IV. CONCLUSION

In this paper we studied the Tensor Network Renormalization algorithm and used it to study the phase transition of the q-state model. We calculated the free energy F, heat capacity C around the critical point for Ising model. By introducing the gauge invariant quantity χ, we obtained the critical temperature for $q = 2, 3, 4, 5$ model and explained how this can be done in the flow of the tensor network and studied about some CFT concepts including central charge and scaling dimension. From the simulations above, We found that the TRG(LN-TNR) had a big problem about truncation error and unexpected fixed-point.

Thus we introduced other improved TNR algorithm, TEFR, EV-TNR, and Loop-TNR. On on hand, these algorithm used different ways to reduce the CDL tensor to avoid meeting unexpected fixed-point. On the other hand, they used their own way to optimize the cost function during each TNR-iteration step.

Appendix A: Detail Algorithms of Loop-TNR

In this section we show how to find the projectors $\mathbf{P}_{iR}$ and $\mathbf{P}_{iL}$. Beginning with $\mathbf{L}_i^{[1,i]} = I$, we do such QR decomposition in each iteration (applying periodic boundary condition on the index j)

$$\mathbf{L}_i^{[k,j]} \cdot \mathbf{T}_j = \tilde{\mathbf{T}}_j \cdot \mathbf{L}_i^{[k,j+1]}, \ j = i \to i-1 \tag{A1}$$

and LQ decomposition iteration for $\mathbf{R}_i^{[k,j]}$, i.e., $\mathbf{T}_j \cdot \mathbf{R}_i^{[k,j]} = \mathbf{R}_i^{[k,j+1]} \cdot \tilde{\mathbf{T}}_j$. After obtaining the converged value $\mathbf{L}_i^{[\infty,i]}$ and $\mathbf{R}_i^{[\infty,i]}$, we can calculate the projectors as follow, (applying periodic boundary condition on the index i)

$$\begin{aligned} \mathbf{P}_{(i-1)R} &= \mathbf{R}_{i-1}^{[\infty,i-1]} \mathbf{V}_{i-1,i} \frac{1}{\sqrt{\mathbf{\Lambda}_{i-1,i}}} \\ \mathbf{P}_{iL} &= \frac{1}{\sqrt{\mathbf{\Lambda}_{\mathbf{i-1,i}}}} \mathbf{U}_{i-1,i}^\dagger \mathbf{L}_i^{[\infty,i]} \end{aligned} \tag{A2}$$

where $\mathbf{\Lambda}, \mathbf{U}, \mathbf{V}$ are obtained by the singular value decomposition $\mathbf{L}_i^{[\infty,i]} \cdot \mathbf{R}_{i-1}^{[\infty,i-1]} = \mathbf{U}_{i-1,i} \cdot \sqrt{\mathbf{\Lambda}_{i-1,i}} \cdot \sqrt{\mathbf{\Lambda}_{i-1,i}} \cdot \mathbf{V}_{i-1,i}^\dagger$. The schematic of these iteration steps are shown in Fig. 19.

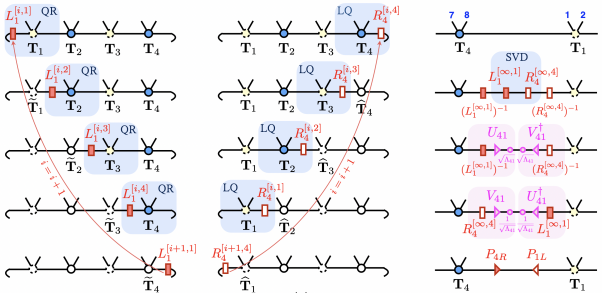

FIG. 19: Entanglement filtering step.

Appendix B: Personal Contribution

1. **Jun-Han Huang (19307110193)** wrote the code of LN-TNR and Loop-TNR algorithms, analyzed and explained the data and results.

2. **Zai-Zhou Xin (19307110240)** studied the LN-TNR, GW-TNR and Loop-TNR algorithm, coded the TRG(LN-TNR) program and TEFR(GW-TNR) program and applied online EV-TNR code to compare the optimization between different algorithm.

3. **Jia-Hao Liu (19307110255)** studied the LN-TNR, GW-TNR and Loop-TNR algorithm, participated in discussions and analyze data, code calculation of central charge and scaling dimensions

---

[1] Z.-Q. Li, L.-P. Yang, Z. Y. Xie, H.-H. Tu, H.-J. Liao, and T. Xiang, Critical properties of the two-dimensional q-state clock model, Phys. Rev. E **101**, 060105 (2020).

[2] S. Elitzur, R. B. Pearson, and J. Shigemitsu, Phase structure of discrete abelian spin and gauge systems, Phys. Rev. D **19**, 3698 (1979).

[3] S. K. Baek, P. Minnhagen, and B. J. Kim, True and quasi-long-range order in the generalized q-state clock model, Phys. Rev. E **80**, 060101 (2009).

[4] Y. Kumano, K. Hukushima, Y. Tomita, and M. Oshikawa, Response to a twist in systems with Z_p symmetry: The two-dimensional p-state clock model, Phys. Rev. B **88**, 104427 (2013).

[5] Jian-Bo Zhang and Da-Ren Ji, Monte carlo simulation of the q-state clock model on a two-dimensional random lattice, Physics Letters A **151**, 469 (1990).

[6] Z.-C. Gu and X.-G. Wen, Tensor-entanglement-filtering renormalization approach and symmetry-protected topological order, Phys. Rev. B **80**, 155131 (2009).

[7] H. N. Phien, J. A. Bengua, H. D. Tuan, P. Corboz, and R. Orús, Infinite projected entangled pair states algorithm improved: Fast full update and gauge fixing, Phys. Rev. B **92**, 035142 (2015).

[8] R. Orús and G. Vidal, Simulation of two-dimensional quantum systems on an infinite lattice revisited: Corner transfer matrix for tensor contraction, Phys. Rev. B **80**, 094403 (2009).

[9] P. Corboz, J. Jordan, and G. Vidal, Simulation of fermionic lattice models in two dimensions with projected entangled-pair states: Next-nearest neighbor hamiltonians, Phys. Rev. B **82**, 245119 (2010).

[10] G. Li, K. H. Pai, and Z.-C. Gu, Accurate simulation of q-state clock model, arXiv: Statistical Mechanics 10.48550/arXiv.2009.10695 (2020).

[11] M. Levin and C. P. Nave, Tensor renormalization group approach to two-dimensional classical lattice models, Phys. Rev. Lett. **99**, 120601 (2007).

[12] Z.-C. Gu and X.-G. Wen, Tensor-entanglement-filtering renormalization approach and symmetry-protected topological order, Phys. Rev. B **80**, 155131 (2009).

[13] G. Evenbly and G. Vidal, Tensor network renormalization, Phys. Rev. Lett. **115**, 180405 (2015).

[14] G. Evenbly, Algorithms for tensor network renormalization, Phys. Rev. B **95**, 045117 (2017).

[15] S. Yang, Z.-C. Gu, and X.-G. Wen, Loop optimization for tensor network renormalization, Phys. Rev. Lett. **118**, 110504 (2017).

[16] B. Pirvu, F. Verstraete, and G. Vidal, Exploiting translational invariance in matrix product state simulations of spin chains with periodic boundary conditions, Phys. Rev. B **83**, 125104 (2011).