ARTICLE OPEN

Check for updates

Exploiting degeneracy in belief propagation decoding of quantum codes

Kao-Yueh Kuo[1] and Ching-Yi Lai[1]✉

Quantum information needs to be protected by quantum error-correcting codes due to imperfect physical devices and operations. One would like to have an efficient and high-performance decoding procedure for the class of quantum stabilizer codes. A potential candidate is Gallager's sum-product algorithm, also known as Pearl's belief propagation (BP), but its performance suffers from the many short cycles inherent in a quantum stabilizer code, especially highly-degenerate codes. A general impression exists that BP is not effective for topological codes. In this paper, we propose a decoding algorithm for quantum codes based on quaternary BP with additional memory effects (called MBP). This MBP is like a recursive neural network with inhibitions between neurons (edges with negative weights), which enhance the perception capability of a network. Moreover, MBP exploits the degeneracy of a quantum code so that the most probable error or its degenerate errors can be found with high probability. The decoding performance is significantly improved over the conventional BP for various quantum codes, including quantum bicycle, hypergraph-product, surface and toric codes. For MBP on the surface and toric codes over depolarizing errors, we observe error thresholds of 16% and 17.5%, respectively.

npj Quantum Information (2022)8:111 ; https://doi.org/10.1038/s41534-022-00623-2

INTRODUCTION

To demonstrate an interesting quantum algorithm, such as Shor's factoring algorithm[1], a quantum computer needs to implement more than 10^{10} logical operations, which means that the error rate of each logical operation must be much less than 10^{-10} (see ref. [2]). With limited quantum devices and imperfect operations[3,4], quantum information needs to be protected by quantum error-correcting codes to achieve fault-tolerant quantum computation[5]. If a quantum state is encoded in a stabilizer code[6,7], the error syndrome of an occurred error can be measured without disturbing the quantum information of the state. A quantum stabilizer code constructed from a sparse graph is favorable since it affords a two-dimensional layout or simple quantum error-correction procedures. This includes the families of surface and toric codes[8], color codes[9], random bicycle codes[10], and generalized hypergraph-product (GHP) codes[11,12].

For a general stabilizer code, the decoding problem of finding the most probable coset of degenerate errors with a given error syndrome is hard[13,14], and an efficient decoding procedure with good performance is desired. The complexity of a decoding algorithm is usually a function of code length N. Edmonds' minimum-weight perfect matching (MWPM)[15] can be used to decode a surface or toric code[16–19], and the complexity of MWPM is $O(N^3)$, which can be reduced to $O(N^2)$ if local matching is used with minor performance loss[19–21]. Duclos-Cianci and Poulin proposed a renormalization group (RG) decoder, which uses a strategy analogous to the decoding of a concatenated code, to decode a toric (or surface) code with complexity proportional to $N\log(\sqrt{N})$[22]. Both MWPM and RG can be generalized for color codes [23–27].

On the other hand, most sparse quantum codes can be decoded by belief propagation (BP)[10,28–30] or its variants with additional processes[31–35]. BP is an iterative algorithm and the decoding complexity per iteration is $O(Nj)$[30,36], where j is the mean column-weight of the check matrix of a quantum code. In general, an average number of iterations proportional to $\log\log N$ is sufficient for BP decoding[37,38]. In practice, a maximum number of iterations $T_{\max}$ proportional to $\log\log N$ up to a large enough constant will be chosen. So the overall decoding complexity of BP is $O(NjT_{\max})$ or $O(Nj\log\log N)$.

Although BP seems to have the lowest complexity, the long-standing problem is that BP does not perform well on quantum codes with high degeneracy unless additional complex processes are included[31,33]. (We say that a code has high degeneracy or is highly degenerate if it has many stabilizers of weight lower than its minimum distance.) The Tanner graph of a stabilizer code inevitably contains many short cycles, which deteriorate the message-passing process in BP[10,31], especially for codes with high degeneracy[33,34,39]. Any message-passing or neural network decoder may suffer from this issue. One may consider variants of BP with additional efforts in pre-training by neural networks[40–43] or post-processing[33,34] such as ordered statistics decoding (OSD)[44], but these methods may not be practical for large codes. In this paper, we will address this long-standing BP problem by devising an efficient quaternary BP decoding algorithm with memory effects (c.f. Eq. (10)), abbreviated MBP, so that the degeneracy of quantum codes can be exploited. Moreover, many known decoders in the literature treat Pauli X and Z errors separately as binary errors, which may incur additional computation overhead or performance loss. MBP directly handles the quaternary errors.

The problem of hard-decision decoding of a classical code is like an energy-minimization problem in a neural network[45], where an energy function measures the parity-check satisfaction (denoted by J_S).

It is known that BP has been used for energy minimization in statistical physics[31,38,46]. Moreover, an iterative decoder based on the gradient descent optimization of the energy function has been proposed[47]. These motivate us to consider a soft-decision generalization of the energy function with variables that are log-

[1]Institute of Communications Engineering, National Yang Ming Chiao Tung University, Hsinchu 300093, Taiwan. ✉email: cylai@nycu.edu.tw

likelihood ratios (LLRs) of Pauli errors and make connections between BP and the gradient descent algorithm. We define an energy function with an additional term J_D that measures the distance between a recovery operator and the initial channel statistics. Then we show that BP in the log domain is like a gradient descent optimization for this generalized energy function but with more elegant step updates[48]. This explains why the conventional BP may work well on a nondegenerate quantum code[10], since this is similar to the classical case [47].

For a highly-degenerate quantum code, it has many low-weight stabilizers corresponding to local minimums in the energy topology so that the conventional BP easily gets trapped in these local minimums near the origin. This suggests that we should use a larger step (which can be controlled by message normalization)[30]. However, this is simply not enough since the energy-minimization process may not converge if large steps are made. An observation from neural networks is that inhibitions (edges with negative weights) between neurons can enhance the perception capability of a network and improve the pattern-recognition accuracy[49–53]. MBP is mathematically formulated to have this inhibition functionality, which helps to resist wrong beliefs passing on the Tanner graph (due to short cycles[37]) or to effectively accelerate the search in a gradient descent optimization. An important feature of MBP is that no additional computation is required, and thus, the complexity of MBP remains the same as the conventional BP with message normalization.

The performance of MBP can be further improved by choosing an appropriate step-size for each error syndrome. However, it is difficult to precisely determine the step-size. If the step-size is too large, MBP may return incorrect solutions or diverge. We propose to choose the step-size using an ε-net so the step-size can be determined adaptively. This adaptive scheme will be called AMBP. The overall complexity of AMBP is still $O(Nj\log\log N)$ since the chosen ε is independent of N.

Another technique adopted in MBP is to use fixed initialization[54,55]. The energy function and energy topology are defined according to the channel statistics. If MBP performs well on a certain channel statistics (say, at a certain depolarizing rate ϵ_0), it means that MBP can correctly determine most syndrome-and-error pairs on that topology. Thus it is better to decode using this energy topology, regardless of the true channel statistics. This technique works for any quantum code.

Computer simulations of MBP on various quantum codes are performed. Note that MBP naturally extends to a more complicated error model of simultaneous data and measurement errors[56]; however, perfect syndrome measurements are assumed in this paper since we focus on the algorithm and performance of BP on degenerate quantum codes. In RESULTS, we demonstrate the decoding of quantum bicycle codes[10], a highly-degenerate GHP code[33], and the (rotated) surface or toric codes[57,58]. Our simulation results show that MBP performs significantly better than the conventional BP. In particular, any degenerate error of the target error up to a stabilizer can also be the target and MBP is able to locate such errors. (See more discussions and examples about energy minimization and the memory effects of MBP in ref. [48]).

RESULTS

Computer simulations

Our main results are based on an efficient decoder for quantum codes, the quaternary MBP (MBP$_4$) (see Algorithm 1). MBP$_4$ has a configurable step-size, which is scaled by a positive constant α^{-1}. For comparison, the conventional quaternary BP (BP$_4$) is similar to MBP$_4$ with $\alpha = 1$ but without additional memory effects. MBP$_4$ can be extended as AMBP$_4$ (Algorithm 2). We simulate the decoding performances of various quantum codes by MBP$_4$ and AMBP$_4$ in the following. The message-update schedule will be denoted by a prefix parallel/serial[28].

For an [[N, K, D]] quantum code that encodes K logical qubits into N physical qubits with minimum distance D, if any errors of weight smaller or equal to t are correctable, its logical error rate is

$$P_{\mathrm{BDD}}(t) \triangleq 1 - \sum_{i=0}^{t} \binom{N}{i} \epsilon^i (1-\epsilon)^{N-i} \tag{1}$$

at depolarizing rate ϵ, using bounded distance decoding (BDD). Let $r \times$ BDD denote the case that any N-fold Pauli error of weight $\le t = \lfloor \frac{rD-1}{2} \rfloor$ is correctable so that the logical error rate is $P_{\mathrm{BDD}}(\lfloor \frac{rD-1}{2} \rfloor)$. If D is unknown, we may directly specify BDD with some t instead of $r \times$ BDD for comparison. Usually, a good classical decoding procedure has a correction radius between $1 \times$ BDD and $2 \times$ BDD[37]. However, the degeneracy of a quantum code is not considered in BDD; we may have decoding performance much better than $2 \times$ BDD in the quantum case. In addition, the optimal achievable decoding performance is unknown[13,14], so $r \times$ BDD serves as a good benchmark.

The mean weight of the rows in the check matrix of a quantum code is called the row-weight and denoted by k. If the row-weight is small, then the quantum code has many low-weight stabilizers. We say that a quantum code is more degenerate if the row-weight of its check matrix is smaller compared to the minimum distance of the code. We will see that MBP$_4$ improves the conventional BP$_4$ more when the tested code is more degenerate. In our simulations, the normalization factor α for the step-size in MBP$_4$ is chosen to be roughly proportional to k and inversely proportional to the depolarizing rate ϵ. (See the analysis in ref. [48])

A relatively larger step-size may be needed for a highly-degenerate quantum code to decode those errors with a weight larger than the row-weight of the check matrix. We use an ε-net of α to adaptively determine the best value α^* for each error syndrome (Algorithm 2, denoted as AMBP$_4$). Since MBP$_4$ is queried as a subroutine in AMBP$_4$ at most ε^{-1} times, the computation complexity of AMBP$_4$ is higher. If ε is independent of N, then the asymptotic complexity remains the same. To efficiently determine α^* is worth further studying.

We briefly explain how to interpret the simulation results. Let $\mathcal{G}_N$ be the N-fold Pauli group, $\mathcal{S} \subset \mathcal{G}_N$ be the stabilizer group that defines a quantum code, and $\mathcal{S}^\perp \subset \{I, X, Y, Z\}^{\otimes N}$ denote the set of operators with phase $+1$ that commute with $\mathcal{S}$ in $\mathcal{G}_N$. Let n_{tot} be the number of tested error samples for a data point in the simulation of the performance curve of a code. Suppose that $\mathbf{E}^{(i)}$ and $\hat{\mathbf{E}}^{(i)} \in \{I, X, Y, Z\}^{\otimes N}$ are the tested and estimated errors, respectively, for $i = 1, 2, \ldots, n_{\mathrm{tot}}$. Denote

$$n_0 = \#\text{ of pairs } (\mathbf{E}^{(i)}, \hat{\mathbf{E}}^{(i)}) : \hat{\mathbf{E}}^{(i)} \ne \mathbf{E}^{(i)}, \tag{2}$$

$$n_e = \#\text{ of pairs } (\mathbf{E}^{(i)}, \hat{\mathbf{E}}^{(i)}) : \hat{\mathbf{E}}^{(i)} \notin \pm \mathbf{E}^{(i)} \mathcal{S}, \tag{3}$$

$$n_u = \#\text{ of pairs } (\mathbf{E}^{(i)}, \hat{\mathbf{E}}^{(i)}) : \hat{\mathbf{E}}^{(i)} \mathbf{E}^{(i)} \in \mathcal{S}^\perp \setminus \pm \mathcal{S}. \tag{4}$$

Empirically, we have the classical block error rate $P(\hat{\mathbf{E}} \ne \mathbf{E}) = n_0/n_{\mathrm{tot}}$, the quantum logical error rate $P(\hat{\mathbf{E}} \notin \pm \mathbf{E}\mathcal{S}) = n_e/n_{\mathrm{tot}}$, and the undetected error rate $P(\hat{\mathbf{E}}\mathbf{E} \in \mathcal{S}^\perp \setminus \pm \mathcal{S}) = n_u/n_{\mathrm{tot}}$.

Since $(\hat{\mathbf{E}} \notin \pm \mathbf{E}\mathcal{S}) \subseteq (\hat{\mathbf{E}} \ne \mathbf{E})$, by Bayes' rule, we have

$$\begin{aligned} P(\hat{\mathbf{E}} \notin \pm \mathbf{E}\mathcal{S}) &= P(\hat{\mathbf{E}} \notin \pm \mathbf{E}\mathcal{S}, \hat{\mathbf{E}} \ne \mathbf{E}) \\ &= P(\hat{\mathbf{E}} \ne \mathbf{E}) \times P(\hat{\mathbf{E}} \notin \pm \mathbf{E}\mathcal{S} | \hat{\mathbf{E}} \ne \mathbf{E}) \\ &= \frac{n_0}{n_{\mathrm{tot}}} \times \frac{n_e}{n_0}. \end{aligned} \tag{5}$$

Usually, a classical decoding strategy is to lower n_0/n_{tot}, which means that the target error needs to be accurately located from a given syndrome. Such a strategy has a limit in performance due to short cycles or strong degeneracy of the code. If a decoder converges to anyone of the degenerate errors, the decoding succeeds. A better strategy has to also lower the ratio n_e/n_0, which will be called the error suppression ratio, by exploiting the degeneracy.

In the simulations, $\mathbf{E}^{(i)}$ is drawn from a memoryless depolarizing error model and then decoded as $\hat{\mathbf{E}}^{(i)}$. The pairs $(\mathbf{E}^{(i)}, \hat{\mathbf{E}}^{(i)})$ are collected until we have 100 logical error events for a data point. Otherwise, an error bar between two crosses is used to show the 95% confidence interval (1.96 times the standard error of the mean). If a maximum number of iteration $T_{\max}$ is reached, but the BP does not converge, the decoding stops, and this error sample is counted as a logical error. $T_{\max}$ is chosen to match the literature for comparison if it was specified. (Empirically, $T_{\max}$ is chosen to be much larger than the average number of iterations τ.) We will see that MBP$_4$ significantly improves BP$_4$ with better convergence speed (smaller τ) and lower logical error rate (with $n_e/n_0 < 1$).

Bicycle codes

MacKay et al. constructed families of random bicycle codes, which are sparse-graph codes with performance possibly close to the quantum Gilbert–Varshamov rate[10]. To have an [[N, K]] random bicycle code, the number of row-weight k is chosen and two random circulant matrices are generated accordingly to define the check matrix of a quantum code of rate $\frac{K}{N}$ (after proper row-deletion). Since the minimum distance of the code is no larger than k due to the code construction, it may have a high decoding error-floor when k is small. For [[3786, 946]] bicycle codes, MacKay et al. showed that a code of row-weight $k \geq 24$ can have good BP decoding performance. However, the decoding complexity is lower for a check matrix with a smaller k and the syndrome measurements are simpler. Thus we would like to have a good decoder for random bicycle codes of small row-weight.

We first construct bicycle codes with the same parameters as in ref. [10]. Figure 1a shows the conventional BP$_4$ performance on [[3786, 946]] bicycle codes for row-weights $k = 24, 20, 16, 12$. It shows that the code of row-weight 24 is able to achieve the logical error rate of 10^{-4} before hitting the error-floor. Also shown in Fig. 1a are the performance curves from MacKay et al. using binary BP (BP$_2$), which treats Pauli X and Z errors separately. It can be seen that BP$_4$ performs better than BP$_2$, because the correlations between X errors and Z errors are considered[28].

Now we show that the performance is significantly improved with MBP$_4$, as shown in Fig. 1b. The error-floor performance is improved, and the code of row-weight 16 is able to well achieve the logical error rate of 10^{-6}. The minimum distance of a bicycle code is usually unknown, so it is hard to compare its performance with $r \times$ BDD for some r. However, we know that the minimum distance is no larger than k for a bicycle code. Thus the performance of a code with $k = 16$ is at most $P_{\text{BDD}}(\lfloor \frac{16-1}{2} \rfloor)$ for $1 \times$ BDD. On the other hand, the performance of MBP$_4$ on the code of row-weight 16 is close to $P_{\text{BDD}}(t)$ with t between 140 and 200 depending on the logical error rate as shown in Fig. 1b, which is better than $17 \times$ BDD.

The average numbers of iterations are shown in Fig. 1c. The convergence behavior is good since the number of average iterations decreases when the physical error rate decreases. It can be seen that the number of average iterations using MBP$_4$ for $k = 12$ decreases more than in the other three cases of $k = 16$, 20, and 24 since there are more lower-weight stabilizers and hence more low-weight degenerate errors.

Next, we study whether MBP$_4$ improves the error suppression ratio n_e/n_0 defined in Eq. (5). Detailed error counts n_0, n_e, n_u, and n_{tot} of (M)BP$_4$ on the two codes of $k = 16$ and $k = 12$ at depolarizing rate $\epsilon = 0.027$, 0.037, and 0.049 are provided in Table 1. MBP$_4$ has $n_e/n_0 < 1$ for these two codes if $\epsilon \leq 0.049$. Note

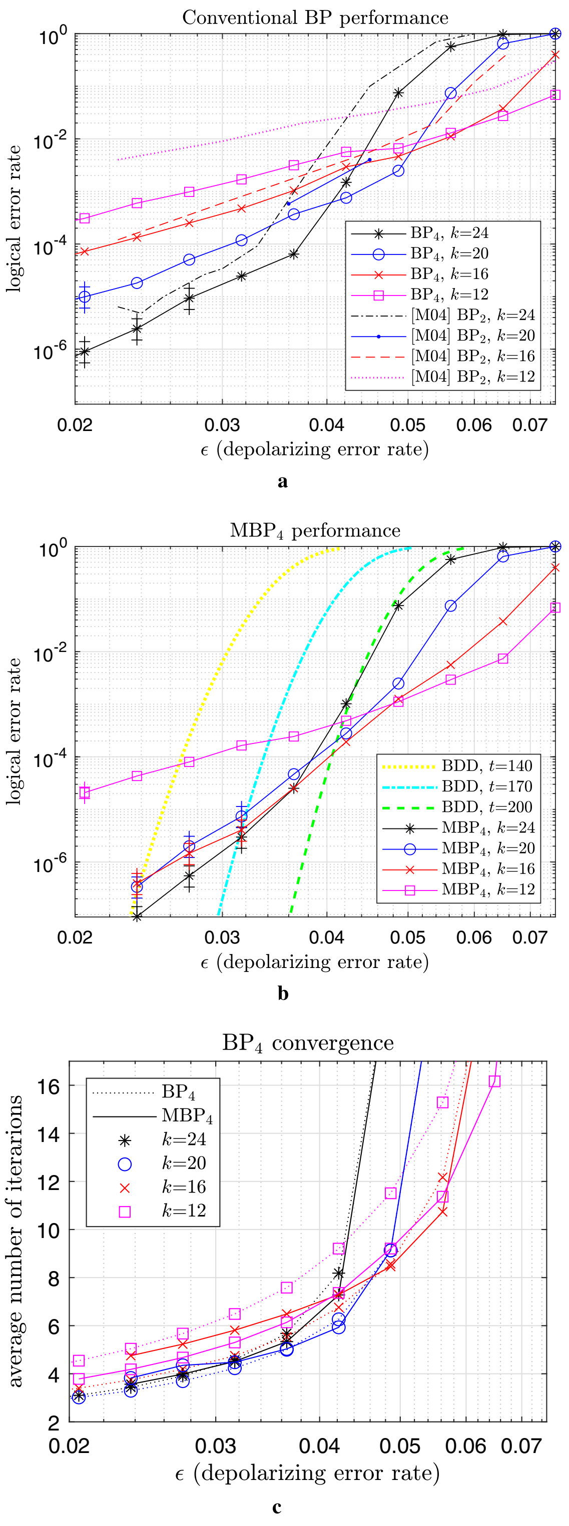


Fig. 1 Performance of parallel BP$_4$ and MBP$_4$ on the [[3786, 946]] bicycle codes of different row-weights (k), based on $T_{\max} = 90$. **a** MBP$_4$ with $\alpha = 1$ (conventional BP$_4$). **b** MBP$_4$ with appropriate $\alpha > 1$ chosen for each ϵ. **c** Average numbers of iterations in **a** (dotted lines) and **b** (solid lines). The [M04] curves in **a** are obtained using a conversion from bit-flip error rate to depolarizing error rate, based on Fig. 6 and Eq. (40) in ref. [10]. In **a** and **b**, 100 logical error events are collected for each data point; or otherwise, an error bar between two crosses indicates the 95% confidence interval.

Table 1. Numbers of various events in the simulations of BP_4 and MBP_4 on the bicycle codes of row-weights 16 and 12.

BP_4 at ϵ:	0.027	0.037	0.049
$k = 16$: n_{tot}	398,936	95,977	21,590
n_0	100	100	103
n_e	100	100	100
n_u	0	0	0
$k = 12$: n_{tot}	101,997	31,765	15,155
n_0	134	154	206
n_e	100	100	100
n_u	1	0	3
MBP_4 at ϵ:	0.027	0.037	0.049
$k = 16$: n_{tot}	12,635,150	3,920,434	79,172
n_0	34	126	102
n_e	20	100	100
n_u	0	0	0
$k = 12$: n_{tot}	1,251,769	410,670	89,821
n_0	518	466	270
n_e	100	100	100
n_u	1	3	2

that n_e/n_0 is small if the decoder finds degenerate errors most of the time. We observe that the decoder exploits the degeneracy more for a code with stabilizers of lower-weight. If the depolarizing rate is smaller, the ratio n_e/n_0 is smaller for MBP_4 on both codes. For $k \le 12$, the minimum distance of a bicycle code would be too small to have a low error-floor.

We remark that the conventional BP_4 has $n_e/n_0 \approx 1$ for most cases when $k \ge 16$. Also listed in Table 1 are the numbers of undetected errors, which are nonzero for $k = 12$. However, the ratio n_u/n_{tot} tends to be small.

To further improve the performance of these bicycle codes, we use $AMBP_4$ with $\alpha^* \in \{2.4, 2.3, \ldots, 0.5\}$. Herein we consider the serial schedule because it accelerates the message update and enlarges the error-correction radius in finite iterations. The performance curves in Fig. 1b are significantly improved, as shown in Fig. 2.

In the case of quantum communication, we may focus on a target logical error rate of 10^{-4} (see ref. [10]), where quantum retransmission is possible if necessary[59]. Consider $\epsilon = \frac{t}{N}$ for large N. The quantum Gilbert–Varshamov rate[6,60] states that there exists a code of rate $\frac{1}{4}$ with the target logical error rate at $\epsilon = 0.063$. One can see from Fig. 2 that the [[3786, 946]] bicycle code with $k = 16$ has this target logical error rate at $\epsilon = 0.057$, which is close to the quantum Gilbert–Varshamov rate.

Generalized hypergraph-product code

Herein we consider the decoding of an [[882, 48, 16]] GHP code constructed in ref. [33]. This code has row-weight 8, which is less than its minimum distance and is thus highly degenerate. The performance of this code under each decoding strategy is shown in Fig. 3. The conventional BP_4, no matter parallel or serial, does not perform good enough. On the other hand, we find that most errors can be decoded by MBP_4 with $\alpha \in [1.2, 1.5]$. The results can be further improved by $AMBP_4$ with $\alpha^* \in \{1.5, 1.49, \ldots, 0.5\}$, for both the parallel and serial schedules.

For reference, we also plot the performance curves in the literature[33] in Fig. 3. The curve "[PK19] BP" is quaternary BP with a

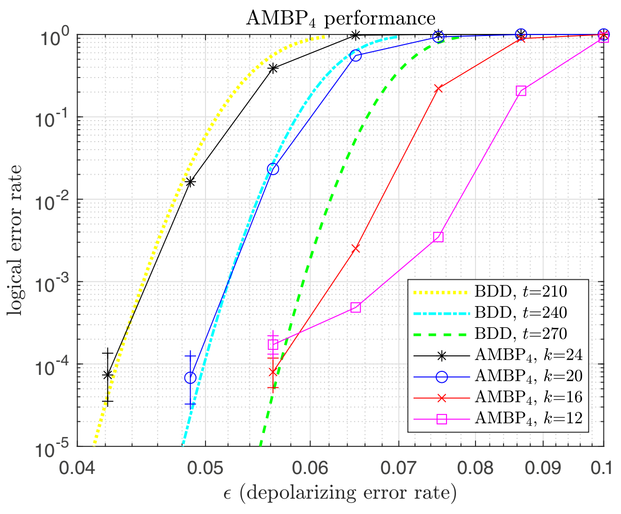


Fig. 2 Performance of serial AMBP4 on [[3786, 946]] bicycle codes. Each [[3786, 946]] bicycle code is constructed with a specific row-weight k. Several BDD performance curves are provided for reference.

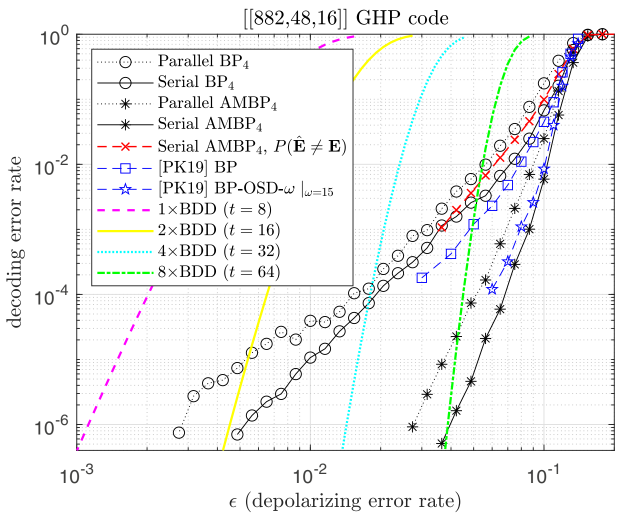


Fig. 3 The performance of the [[822, 48, 16]] GHP code. The maximum number of iteration is $T_{max} = 32$. The decoding error rate for curve $P(\hat{\mathbf{E}} \neq \mathbf{E})$ is the classical block error rate, and for the other (non-BDD) curves, it is the logical error rate. The [PK19] curves are from ref. [33].

layered schedule and the curve "[PK19] BP-OSD-ω" is BP with OSD and additional post-processing. In addition to BP and OSD, BP-OSD-ω has to sort out 2^ω errors in ω unreliable coordinates, so its complexity is high. For this [[882, 48, 16]] GHP code, as shown in Fig. 3, the performance of $AMBP_4$ is better than BP-OSD-ω with $\omega = 15$. The complexity of $AMBP_4$ is low enough, so we simulate to lower logical error rate.

We also plot several $r \times$ BDD performance curves for comparison. Observe that the curve of serial $AMBP_4$ has a slope roughly aligned with $1 \times$ BDD, but its performance is close to $8 \times$ BDD at a logical error rate of 10^{-6}, since more low-weight errors are corrected. We also draw the curve of the classical block error rate $P(\hat{\mathbf{E}} \neq \mathbf{E}) = n_0/n_{tot}$. It becomes the logical error rate after times the ratio n_e/n_0. Figure 3 shows that the improvement by the ratio $n_e/$

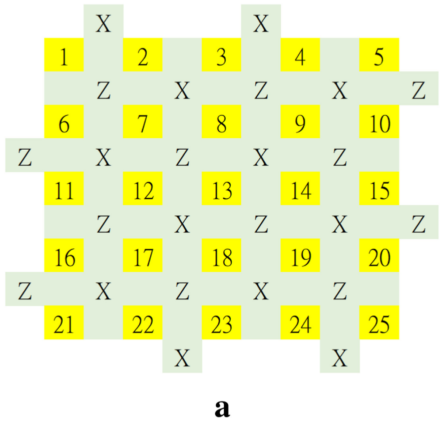
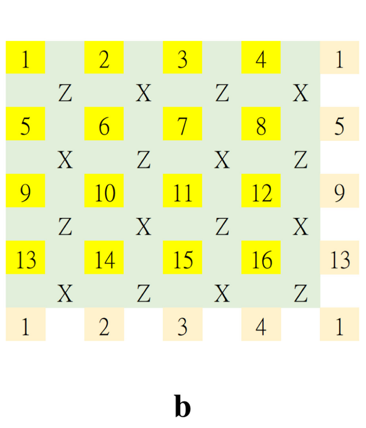


Fig. 4 The lattice representations of (rotated) surface and toric codes. **a** $[[L^2, 1, L]]$ surface code with $L = 5$. **b** $[[L^2, 2, L]]$ toric code with $L = 4$. In both figures, a qubit is represented by a yellow box numbered from 1 to N. Since the toric code is defined on a torus, there are orange boxes on the right and bottom in **b**, each representing the qubit of the same number. An X- or Z-type stabilizer is indicated by a label $W \in \{X, Z\}$ between its neighboring qubits. For example, in **a**, the label X between qubits 1 and 2 is X_1X_2 and the label Z between qubits 1, 2, 6, and 7 is $Z_1Z_2Z_6Z_7$.



n_0 is quite significant, which means that AMBP$_4$ is able to exploit the code degeneracy to have better performance.

Surface and toric codes

In this subsection, we simulate the surface codes with a 45° rotation for lower overhead[57,58]. Our analysis can be applied to rotated toric codes as well.

An $[[L^2, 1, L]]$ surface code for an odd integer L can be defined on an $L \times L$ square lattice. Figure 4a provides an example of $L = 5$. A stabilizer generator of a surface code is of weight 2 or 4, independent of the minimum distance. Consequently, a large surface code is highly-degenerate. As mentioned in the Introduction, the conventional BP cannot handle highly-degenerate quantum codes since there could be many errors of similar likelihood, so BP will hesitate among these errors. The decoding performance curves of the conventional (parallel) BP$_4$ and (serial) MBP$_4$ on several surface codes are shown in Fig. 5. It can be seen that the conventional BP$_4$ does not work well on these surface codes. Moreover, the logical error rate is worse for a surface code with a larger minimum distance.

On the other hand, serial MBP$_4$ is able to decode the surface codes, as shown in Fig. 5. For $L = 17$, the decoding performance of serial MBP$_4$ with $\alpha = 0.65$ is around $1 \times$BDD to $2 \times$BDD, which agrees with Gallager's expectation on BP decoding of classical codes[37].

How MBP$_4$ decodes the surface codes is examined as follows. As previously discussed, n_e/n_0 would be small if a decoder can find degenerate errors of the target, which is indeed the case for MBP$_4$, as shown in Fig. 6a. We also observe undetected error events in the serial MBP$_4$ decoding. For the conventional BP$_4$, we have $n_e/n_0 \approx 1$ and the undetected error rate ≈ 0 for $L > 7$. Thus the improvement of serial MBP$_4$ over BP$_4$ comes at the cost of some undetected error events, as shown in Fig. 6b. (A similar phenomenon was also observed in the neural BP decoder)[42]. This unwanted phenomenon is not surprising, since a large step-size is used so that BP may jump too far, causing logical errors. (We remark that this is not a random search, or otherwise the ratio n_e/n_0 would be as large as 3/4 since there are four logical operators, I, X, Y, and Z, for a logical qubit). However, the undetected error rate is smaller for larger L so this is fine for the purpose of fault-tolerant quantum computation. Figure 6c compares the average numbers of iterations for serial MBP$_4$ and conventional BP$_4$. It can be seen that serial MBP$_4$ uses fewer iterations than the conventional BP$_4$, and yet the performance of serial MBP$_4$ is better. It means that the

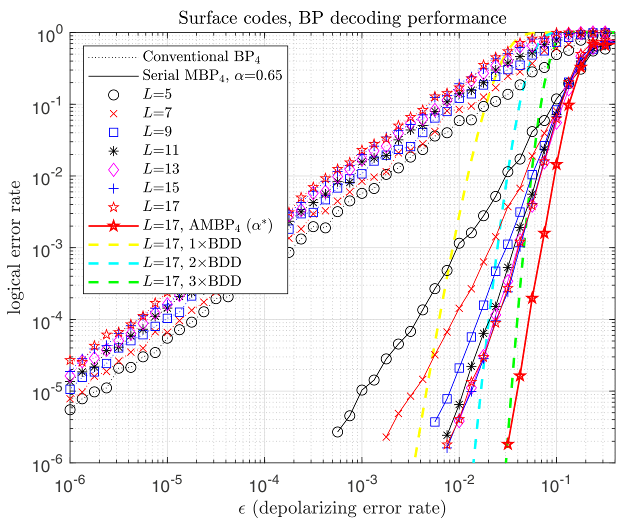


Fig. 5 Performance curves of several surface codes using the conventional BP$_4$ and serial (A)MBP$_4$. The maximum number of iterations is $T_{\max} = 150$. The technique of fixed initialization is used with $\epsilon_0 = 0.013$.

convergence behavior of serial MBP$_4$ is more accurate and the computation is more economic and effective.

Next, we verify that the runtime of MBP$_4$ is $O(Nj)$ per iteration. First, consider the toric codes with mean column-weight $j = 4$. We test serial MBP$_4$ with $\alpha = 0.75$ at depolarizing rate of 0.32 on one core (4.9 GHz) of an Intel i9-9900K machine. The average runtime per iteration is shown in Fig. 7, which is obviously linear in N. Then we consider the surface codes, which have a mean column-weight slightly smaller than 4. As expected, the average runtime per iteration is again linear in N and the slope is smaller than that for the toric codes, as shown in Fig. 7.

Although MBP$_4$ succeeds to decode topological codes from our simulations, it is also observed that the performance of MBP$_4$ on surface codes saturates for large L, i.e., the slope of the performance curve does not increase as L increases. For better decoding performance, we use AMBP$_4$ with $\alpha^* \in \{1.0, 0.99, \ldots, 0.5\}$, and the performance for $L = 17$ is greatly improved, as shown in Fig. 5. In Fig. 8, we plot the performance of AMBP$_4$ for each surface code of lattice size $L \in \{3, 5, 7, \ldots, 17\}$ and an error threshold of about 16% is observed. Similarly, a slightly higher error threshold

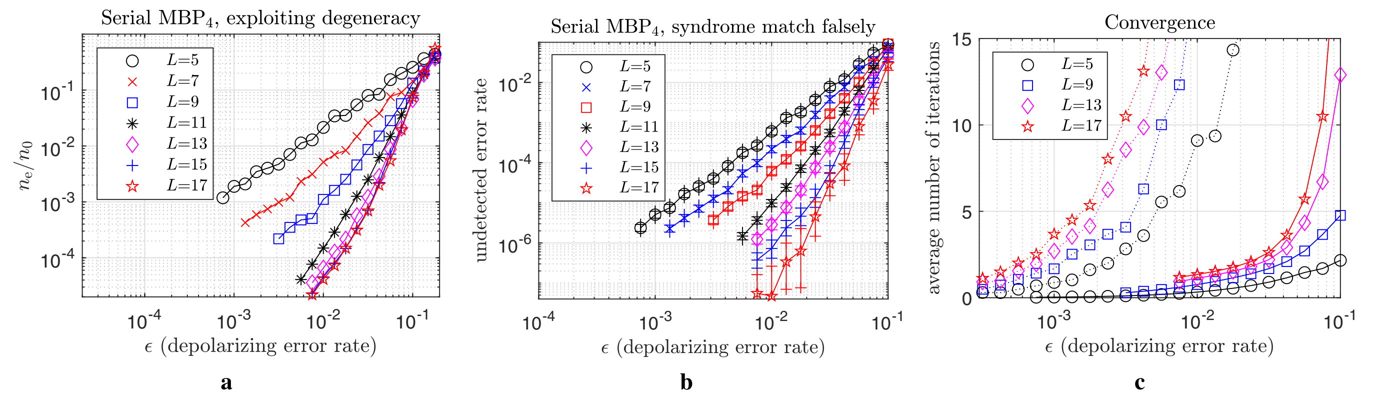


Fig. 6 Some statistical results of serial MBP$_4$ ($\alpha = 0.65$) on surface codes (solid lines). a The error suppression ratio n_e/n_0. **b** Undetected error rate. **c** Average numbers of iterations (solid lines); also shown in **c** are the numbers for the conventional BP$_4$ (dotted lines). In **b**, an error bar between two crosses indicates the 95% confidence interval.

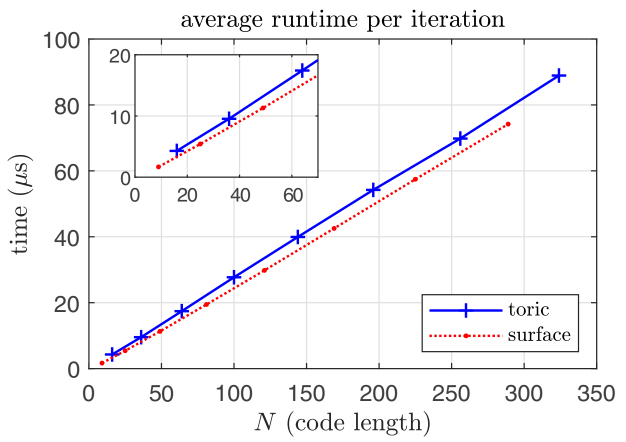


Fig. 7 Almost linear runtime of MBP$_4$. The average runtime of MBP$_4$ on each toric or surface code is plotted.

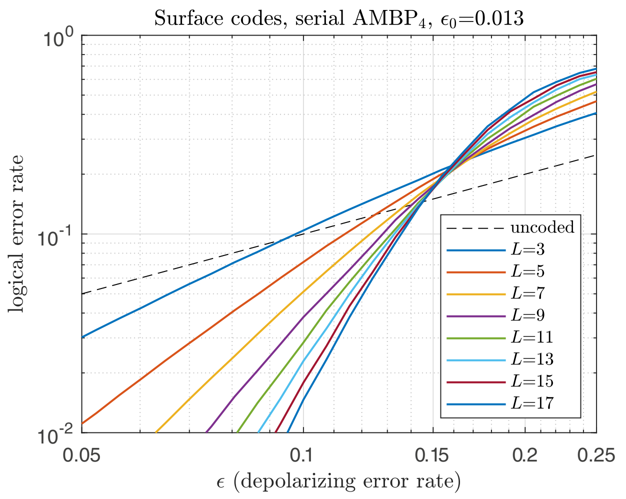


Fig. 8 The threshold performance of serial AMBP$_4$ on the surface codes. The dashed line represents the error rate without error-correction.

of roughly 17.5% can be observed on the toric codes, using AMBP$_4$ decoding[48].

Finally, we compare various polynomial-time decoders in terms of error thresholds and computation complexity in Table 2. Let ϵ_{surf} and ϵ_{toric} denote the error thresholds for the surface and toric codes, respectively. Certain decoders can approach the quantum hashing bound (which is roughly 18.9%[6,60,61]) for the surface or toric codes, but they will not be considered due to high complexities[62–64]. MWPM achieves $\epsilon_{surf} \approx \epsilon_{toric} = 15.5\%$[17,19]. RG combined with BP (RG-BP) achieves $\epsilon_{toric} = 16.4\%$[22]. The matrix product states (MPS) decoder achieves $17\% \leq \epsilon_{surf} \leq 18.5\%$ with complexity $O(N^2)$ specified[65]. Union-find (UF) has complexity almost linear in N, but its decoding performance is slightly worse than MWPM[66]. BP-assisted MWPM (BP-MWPM) has high thresholds for both the surface and toric codes, but its complexity is $O(N^{2.5})$[32]. AMBP$_4$ achieves roughly $\epsilon_{surf} = 16\%$ and $\epsilon_{toric} = 17.5\%$, and its complexity is only $O(N \log\log N)$. Thus AMBP$_4$ is very competing in both decoding performance and computation complexity.

Table 2. The error thresholds and computation complexities of various decoders on the surface codes (ϵ_{surf}) and toric codes (ϵ_{toric}) over depolarizing errors. An entry is denoted – if the value is not provided in the literature.

decoder	ϵ_{surf}	ϵ_{toric}	complexity
MWPM[15]	15.5%[19]	15.5%[a,b][17]	$O(N^2)$[20]
RG-BP[22]	–	16.4%	$O(N \log N)$
MPS[65]	17–18.5%	–	$O(N^2)$
UF[66]	–	14.85%[a]	$O(N)$
BP-MWPM[32]	17.76%[b]	17.76%	$O(N^{2.5})$
AMBP$_4$ (this paper)	16%	17.5%	$O(N \log\log N)$

[a]If the threshold is derived by assuming only bit-flip errors, it will be rescaled by a factor 3/2 (as in Eq. (40) of ref. [10]).
[b]It is generally believed that $\epsilon_{surf} \leq \epsilon_{toric}$. Figure 10 in ref. [19] seems to suggest $\epsilon_{toric} = 15\% < \epsilon_{surf} = 15.5\%$ by the intersection points. However, the MWPM threshold on toric codes over bit-flip errors is 0.1031[17], which, after rescaled by 3/2, provides a higher threshold value of $\epsilon_{toric} = 15.5\%$. Similarly, Figs. 10 and 12 in ref. [32] seem to confusingly suggest $\epsilon_{toric} = 17.76\% < \epsilon_{surf} = 17.84\%$ by the intersection points. Thus only the value of 17.76% is concluded in the literature.

DISCUSSION

We analyzed the energy topology of BP and proposed an efficient BP decoding algorithm for quantum codes called MBP. MBP explores the degeneracy of a quantum code by finding degenerate errors of the target. MBP is competing in both decoding performance and computation complexity. The reader can find a detailed comparison of the thresholds and complexities of MWPM- or BP-based decoders on various topological codes (including color codes and XZZX codes) over depolarizing errors in Table II of ref. [67].

It is known that BP can be treated as a recurrent neural network (RNN)[68]. Similarly, our MBP induces an RNN with inhibition without the pre-training process. This may provide an explanation why RNN decoders can work on degenerate codes[42]. Thus, one may consider an MBP-based neural network decoder, which naturally generalizes the BP-based neural networks[42,68]. One would have an adjustable parameter $a_{mn,i}$ for each edge (m, n) at iteration i.

In AMBP$_4$, one has to find a proper value for α^*. An efficient strategy to select α^* is desired. A clue is that α^* should be related to the properties of the error syndrome. For example, a syndrome vector of high weight usually corresponds to an error of high weight and a smaller value of α should be chosen.

Our decoder can be extended for fault-tolerant quantum computation with imperfect quantum gates, following the initial study of BP decoding for both data and syndrome errors[56]. This is our ongoing work.

METHODS

BP with additional memory effects (MBP)

Decoding an $[[N, K]]$ quantum code subject to an (unknown) error $\mathbf{E} \in \mathcal{G}_N$ is to estimate an $\hat{\mathbf{E}} \in \pm\mathbf{E}\mathcal{S}$, given a check matrix $\mathbf{S} \in \{I, X, Y, Z\}^{M\times N}$ (where $M \geq N - K$), a syndrome $\mathbf{z} \in \{0, 1\}^M$, a real $\alpha > 0$, and initial LLRs $\{\mathbf{\Lambda}_n = (\mathbf{\Lambda}_n^X, \mathbf{\Lambda}_n^Y, \mathbf{\Lambda}_n^Z) \in \mathbb{R}^3\}_{n=1}^N$ of the error rate at each qubit (see ref. [30]). The error syndrome $\mathbf{z} \in \{0, 1\}^M$ is defined by

$$\mathbf{z}_m \triangleq \begin{cases} 0, & \text{if } \mathbf{E} \text{ and } \mathbf{S}_m \text{ commute;} \\ 1, & \text{if } \mathbf{E} \text{ and } \mathbf{S}_m \text{ anticommute;} \end{cases}$$

where $\mathbf{S}_m$ is the m-th row of $\mathbf{S}$. For simplicity, an $\mathbf{E} \in \{I, X, Y, Z\}^{\otimes N}$ is represented by $\mathbf{E} = (\mathbf{E}_1, \mathbf{E}_2, \ldots, \mathbf{E}_N) \in \{I, X, Y, Z\}^N$.

The LLR value $\mathbf{\Lambda}_n^W \triangleq \mathbf{p}_n^I/\mathbf{p}_n^W$ for $W \in \{X, Y, Z\}$ is initialized by a distribution vector $\mathbf{p}_n = (\mathbf{p}_n^I, \mathbf{p}_n^X, \mathbf{p}_n^Y, \mathbf{p}_n^Z) = (1 - \epsilon_0, \frac{\epsilon_0}{3}, \frac{\epsilon_0}{3}, \frac{\epsilon_0}{3})$ for independent depolarizing errors. The value of ϵ_0 can be the channel error rate ϵ or an independent fixed point $\in [0, 1]$. When the LLR is initialized by a fixed point, this is referred to as fixed initialization.

We denote $\mathcal{N}(m) = \{n : \mathbf{S}_{mn} \neq I\}$ and $\mathcal{M}(n) = \{m : \mathbf{S}_{mn} \neq I\}$. Define functions $\lambda_W : \mathbb{R}^3 \to \mathbb{R}$

$$\lambda_W(\gamma^X, \gamma^Y, \gamma^Z) \triangleq \ln \frac{1 + e^{-\gamma^W}}{e^{-\gamma^X} + e^{-\gamma^Y} + e^{-\gamma^Z} - e^{-\gamma^W}} \tag{6}$$

for $W \in \{X, Y, Z\}$. Also define an operation $\boxplus$: for a set of k real scalars $a_1, a_2, \ldots, a_k \in \mathbb{R}$,

$$\boxplus_{n=1}^{k} a_n \triangleq 2\tanh^{-1}\left(\prod_{n=1}^{k} \tanh \frac{a_n}{2}\right). \tag{7}$$

We may simplify a notation $\mathcal{M}(n) \setminus \{m\}$ as $\mathcal{M}(n) \setminus m$.

Algorithm 1:. Quaternary MBP (MBP$_4$)

Input: $\mathbf{S} \in \{I, X, Y, Z\}^{M\times N}$, $\mathbf{z} \in \{0, 1\}^M$, $T_{\max} \in \mathbb{Z}_+$, a real $\alpha > 0$, and initial LLRs $\{(\mathbf{\Lambda}_n^X, \mathbf{\Lambda}_n^Y, \mathbf{\Lambda}_n^Z) \in \mathbb{R}^3\}_{n=1}^N$.

Initialization. For $n = 1$ to N and $m \in \mathcal{M}(n)$, let

$$\mathbf{\Gamma}_{n\to m}^W = \mathbf{\Lambda}_n^W, \ \ W \in \{X, Y, Z\}.$$

Horizontal Step. For $m = 1$ to M and $n \in \mathcal{N}(m)$, compute

$$\mathbf{\Delta}_{m\to n} = (-1)^{\mathbf{z}_m} \boxplus_{n' \in \mathcal{N}(m)\setminus n} \lambda_{\mathbf{S}_{mn'}}(\mathbf{\Gamma}_{n'\to m}). \tag{8}$$

Vertical Step. For $n = 1$ to N and $W \in \{X, Y, Z\}$, compute

$$\mathbf{\Gamma}_n^W = \mathbf{\Lambda}_n^W + \frac{1}{\alpha} \sum_{\substack{m \in \mathcal{M}(n) \\ \langle W, \mathbf{S}_{mn}\rangle = 1}} \mathbf{\Delta}_{m\to n}. \tag{9}$$

- (**Hard Decision**.) Let $\hat{\mathbf{E}} = (\hat{\mathbf{E}}_1, \hat{\mathbf{E}}_2, \ldots, \hat{\mathbf{E}}_N)$, where $\hat{\mathbf{E}}_n = I$ if $\mathbf{\Gamma}_n^W > 0$ for all $W \in \{X, Y, Z\}$, and $\hat{\mathbf{E}}_n = \arg\min_{W\in\{X,Y,Z\}} \mathbf{\Gamma}_n^W$, otherwise.
- If $\langle \hat{\mathbf{E}}, \mathbf{S}_m \rangle = \mathbf{z}_m \, \forall m$, halt and return "CONVERGE";
- Otherwise, if the maximum number of iterations $T_{\max}$ is reached, halt and return "FAIL";
- (**Fixed Inhibition**.) Otherwise, for $n = 1$ to N, $m \in \mathcal{M}(n)$, and $W \in \{X, Y, Z\}$, compute

$$\mathbf{\Gamma}_{n\to m}^W = \mathbf{\Gamma}_n^W - \langle W, \mathbf{S}_{mn}\rangle \mathbf{\Delta}_{m\to n}. \tag{10}$$

- Repeat from the horizontal step.

Motivated by the energy topology of a degenerate quantum code and gradient decent energy optimization, we propose MBP$_4$ in Algorithm 1. MBP$_4$ has variable-to-check messages $\lambda_{\mathbf{S}_{mn}}(\mathbf{\Gamma}_{n\to m})$ and check-to-variable messages $\Delta_{m\to n}$, similarly to the log-BP in ref. [30]; however, the message $\Delta_{m\to n}$ is used differently when generating $\mathbf{\Gamma}_{n\to m}^W$ in Eq. (10). Especially we can rewrite Eq. (10) as

$$\mathbf{\Gamma}_{n\to m}^W = \mathbf{\Lambda}_n^W + \left(\frac{1}{\alpha} \sum_{\substack{m' \in \mathcal{M}(n) \\ \langle W, \mathbf{S}_{m'n}\rangle = 1}} \mathbf{\Delta}_{m'\to n}\right) - \langle W, \mathbf{S}_{mn}\rangle \mathbf{\Delta}_{m\to n}. \tag{11}$$

The term $-\langle W, \mathbf{S}_{mn}\rangle \Delta_{m\to n}$ is called inhibition, which provides adequate strength to resist the wrong belief looped in the short cycles. Unlike refs. [28,30], where the corresponding inhibition is also scaled by $1/\alpha$, we suggest to keep this inhibition strength (Eq. (11)) since this part is the belief inherited in check node m, and it must remain unchanged when we update the belief in variable n to make the decoding less affected by the short cycles. Consequently, these introduce additional memory effects in MBP. How to choose the factor α is intriguing. Please see ref. [48] for more discussions; for reference, MBP$_4$ can also be defined in the linear domain.

Remark 1. In Algorithm 1, one can verify that

$$\lambda_{\mathbf{S}_{mn}}(\mathbf{\Gamma}_{n\to m}) = \lambda_{\mathbf{S}_{mn}}(\mathbf{\Gamma}_n) - \mathbf{\Delta}_{m\to n}. \tag{12}$$

It is more efficient to update $\lambda_{\mathbf{S}_{mn}}(\mathbf{\Gamma}_{n\to m})$ in this way so that for each n, computing $\lambda_{\mathbf{S}_{mn}}(\mathbf{\Gamma}_n)$ needs at most three computations of $\lambda_{\mathbf{S}_{mn}}(\cdot)$ for $\mathbf{S}_{mn} \in \{X, Y, Z\}$; otherwise, directly computing $\lambda_{\mathbf{S}_{mn}}(\mathbf{\Gamma}_{n\to m})$ needs $|\mathcal{M}(n)|$ (usually ≥3) computations of $\lambda_{\mathbf{S}_{mn}}(\cdot)$. On the other hand, the computation in the horizontal step can be simplified as in Remarks 1 and 4 of ref. [30]. Then the MBP$_4$ complexity is proportional to the number of edges Nj per iteration, and thus the overall complexity is $O(NjT_{\max})$ or $O(Nj \log\log N)$.

Adaptive MBP

Herein we propose a variation of MBP$_4$ with α chosen adaptively, as shown in Algorithm 2. The value of α controls the search radius of MBP$_4$. Typically, a fixed α is chosen so that BP focuses on an error-correction region between $1 \times$ BDD and $2 \times$ BDD. For highly-degenerate codes, we intend to correct errors in a much wider region, and thus we need to consider variations in α. More specifically, α should be chosen according to the given syndrome vector. Precisely determining a required value of α helps to achieve the desired performance, but in general, it is difficult to do so. Generating a solution by referring to multiple instances of the decoder is an important technique in Monte Carlo sampling methods (cf. parallel tempering in ref. [62]), as well as in neural networks (cf. Fig. 4 of ref. [68]). This is like using an ε-net of α. Thus we conduct multiple instances of MBP$_4$ with different values of α, and choose a valid (syndrome-matched) solution with the largest (the most conservative) value of α. This value of α is adaptively chosen and denoted by α^*, so Algorithm 2 is referred to as AMBP$_4$. For this algorithm, the prefix parallel/serial is used to indicate the schedule type of the oracle function MBP$_4$.

Note that in Algorithm 2 each value of α_i is tested in a sequential manner; these α_i's can be tested in parallel if the physical resources for implementation are available.

```
Algorithm 2:. Adaptive MBP$_4$ (AMBP$_4$)
    Input: $\mathbf{S} \in \{I, X, Y, Z\}^{M \times N}$, $\mathbf{z} \in \{0, 1\}^M$, $T_{\max} \in \mathbb{Z}_+$, $\{\mathbf{\Lambda}_n = (\mathbf{\Lambda}_n^X, \mathbf{\Lambda}_n^Y, \mathbf{\Lambda}_n^Z) \in \mathbb{R}^3\}_{n=1}^N$, a sequence of real values $\alpha_1 > \alpha_2 > \cdots > \alpha_l > 0$, and an oracle function MBP$_4$.
    Initialization: Let $i = 1$.
    MBP Step: Run MBP$_4$($\mathbf{S}$, $\mathbf{z}$, $T_{\max}$, $\alpha_i$, $\{\mathbf{\Lambda}_n\}$), which will return "CONVERGE" or "FAIL" with estimated $\hat{\mathbf{E}} \in \{I, X, Y, Z\}^N$.
    Adaptive Check:
        • If the return indicator is "CONVERGE", then return "SUCCESS" (with valid $\hat{\mathbf{E}}$ and $\alpha^* = \alpha_i$);
        • Let $i \leftarrow i + 1$. If $i > l$, return "FAIL" (with invalid $\hat{\mathbf{E}}$);
        • Otherwise, repeat from the MBP Step.
```

DATA AVAILABILITY

The data that support the findings of this study are available from the corresponding author upon reasonable request.

Received: 23 May 2021; Accepted: 23 August 2022;
Published online: 14 September 2022

REFERENCES

1. Shor, P. W. Algorithms for quantum computation: Discrete logarithms and factoring. In *Proc. 35th Annual Symposium on Foundations of Computer Science* (*FOCS*). 124–134 (IEEE, 1994).
2. Suchara, M. et al. QuRE: The quantum resource estimator toolbox. In *Proc. IEEE 31st International Conference on Computer Design* (*ICCD*). 419–426 (IEEE, 2013).
3. Wang, Y. et al. Single-qubit quantum memory exceeding ten-minute coherence time. *Nat. Photonics* **11**, 646–650 (2017).
4. Arute, F. et al. Quantum supremacy using a programmable superconducting processor. *Nature* **574**, 505–510 (2019).
5. Shor, P. W. Fault-tolerant quantum computation. In *Proc. 37th Annual Symposium on Foundations of Computer Science* (*FOCS*). 56–65 (IEEE Computer Society, 1996).
6. Gottesman, D. *Stabilizer Codes and Quantum Error Correction*. Ph.D. thesis, California Institute of Technology (1997).
7. Calderbank, A. R., Rains, E. M., Shor, P. W. & Sloane, N. J. A. Quantum error correction via codes over GF(4). *IEEE Trans. Inf. Theory* **44**, 1369–1387 (1998).
8. Kitaev, A. Y. Fault-tolerant quantum computation by anyons. *Ann. Phys.* **303**, 2–30 (2003).
9. Bombin, H. & Martin-Delgado, M. A. Topological quantum distillation. *Phys. Rev. Lett.* **97**, 180501 (2006).
10. MacKay, D. J. C., Mitchison, G. & McFadden, P. L. Sparse-graph codes for quantum error correction. *IEEE Trans. Inf. Theory* **50**, 2315–2330 (2004).
11. Tillich, J.-P. & Zémor, G. Quantum LDPC codes with positive rate and minimum distance proportional to the square root of the blocklength. *IEEE Trans. Inf. Theory* **60**, 1193–1202 (2014).
12. Kovalev, A. A. & Pryadko, L. P. Quantum Kronecker sum-product low-density parity-check codes with finite rate. *Phys. Rev. A* **88**, 012311 (2013).
13. Kuo, K.-Y. & Lu, C.-C. On the hardnesses of several quantum decoding problems. *Quantum Inf. Process.* **19**, 1–17 (2020).
14. Iyer, P. & Poulin, D. Hardness of decoding quantum stabilizer codes. *IEEE Trans. Inf. Theory* **61**, 5209–5223 (2015).
15. Edmonds, J. Paths, trees, and flowers. *Can. J. Math.* **17**, 449–467 (1965).
16. Dennis, E., Kitaev, A., Landahl, A. & Preskill, J. Topological quantum memory. *J. Math. Phys.* **43**, 4452–4505 (2002).
17. Wang, C., Harrington, J. & Preskill, J. Confinement-Higgs transition in a disordered gauge theory and the accuracy threshold for quantum memory. *Ann. Phys.* **303**, 31–58 (2003).
18. Raussendorf, R., Harrington, J. & Goyal, K. A fault-tolerant one-way quantum computer. *Ann. Phys.* **321**, 2242–2270 (2006).
19. Wang, D. S., Fowler, A. G., Stephens, A. M. & Hollenberg, L. C. L. Threshold error rates for the toric and planar codes. *Quantum Inf. Comput.* **10**, 456–469 (2010).
20. Fowler, A. G., Whiteside, A. C. & Hollenberg, L. C. Towards practical classical processing for the surface code. *Phy. Rev. Lett.* **108**, 180501 (2012).
21. Fowler, A. G. Minimum weight perfect matching of fault-tolerant topological quantum error correction in average *O*(1) parallel time. *Quantum Inf. Comput.* **15**, 145–158 (2015).
22. Duclos-Cianci, G. & Poulin, D. Fast decoders for topological quantum codes. *Phys. Rev. Lett.* **104**, 050504 (2010).
23. Wang, D. S., Fowler, A. G., Hill, C. D. & Hollenberg, L. C. L. Graphical algorithms and threshold error rates for the 2d color code. *Quantum Inf. Comput.* **10**, 780–802 (2010).
24. Bombin, H., Duclos-Cianci, G. & Poulin, D. Universal topological phase of two-dimensional stabilizer codes. *N. J. Phys.* **14**, 073048 (2012).
25. Delfosse, N. Decoding color codes by projection onto surface codes. *Phys. Rev. A* **89**, 012317 (2014).
26. Sarvepalli, P. & Raussendorf, R. Efficient decoding of topological color codes. *Phys. Rev. A* **85**, 022317 (2012).
27. Stephens, A. M. Efficient fault-tolerant decoding of topological color codes. Preprint at https://arxiv.org/abs/1402.3037 (2014).
28. Kuo, K.-Y. & Lai, C.-Y. Refined belief propagation decoding of sparse-graph quantum codes. *IEEE J. Sel. Areas Inf. Theory* **1**, 487–498 (2020).
29. Kuo, K.-Y. & Lai, C.-Y. Refined belief-propagation decoding of quantum codes with scalar messages. In *Proc. IEEE Globecom Workshops* 1–6 (IEEE, 2020).
30. Lai, C.-Y. & Kuo, K.-Y. Log-domain decoding of quantum LDPC codes over binary finite fields. In *IEEE Transactions on Quantum Engineering* 1–15 (IEEE, 2021).
31. Poulin, D. & Chung, Y. On the iterative decoding of sparse quantum codes. *Quantum Inf. Comput.* **8**, 987–1000 (2008).
32. Criger, B. & Ashraf, I. Multi-path summation for decoding 2D topological codes. *Quantum* **2**, 102 (2018).
33. Panteleev, P. & Kalachev, G. Degenerate quantum LDPC codes with good finite length performance. *Quantum* **5**, 585 (2021).
34. Roffe, J., White, D. R., Burton, S. & Campbell, E. T. Decoding across the quantum LDPC code landscape. *Phys. Rev. Res.* **2**, 043423 (2020).
35. Grospellier, A., Grouès, L., Krishna, A. & Leverrier, A. Combining hard and soft decoders for hypergraph product codes. *Quantum* **5**, 432 (2021).
36. Davey, M. & MacKay, D. Low-density parity check codes over GF(q). *IEEE Commun. Lett.* **2**, 165–167 (1998).
37. Gallager, R. G. *Research Monograph Series* (MIT Press, 1963).
38. MacKay, D. J. C. Good error-correcting codes based on very sparse matrices. *IEEE Trans. Inf. Theory* **45**, 399–431 (1999).
39. Raveendran, N. & Vasić, B. Trapping sets of quantum LDPC codes. *Quantum* **5**, 562 (2021).
40. Torlai, G. & Melko, R. G. Neural decoder for topological codes. *Phys. Rev. Lett.* **119**, 030501 (2017).
41. Krastanov, S. & Jiang, L. Deep neural network probabilistic decoder for stabilizer codes. *Sci. Rep.* **7**, 11003 (2017).
42. Liu, Y.-H. & Poulin, D. Neural belief-propagation decoders for quantum error-correcting codes. *Phys. Rev. Lett.* **122**, 200501 (2019).
43. Maskara, N., Kubica, A. & Jochym-O'Connor, T. Advantages of versatile neural-network decoding for topological codes. *Phys. Rev. A* **99**, 052351 (2019).
44. Fossorier, M. & Lin, S. Soft-decision decoding of linear block codes based on ordered statistics. *IEEE Trans. Inf. Theory* **41**, 1379–1396 (1995).
45. Bruck, J. & Blaum, M. Neural networks, error-correcting codes, and polynomials over the binary *n*-cube. *IEEE Trans. Inf. Theory* **35**, 976–987 (1989).
46. Yedidia, J. S., Freeman, W. T. & Weiss, Y. Constructing free-energy approximations and generalized belief propagation algorithms. *IEEE Trans. Inf. Theory* **51**, 2282–2312 (2005).
47. Lucas, R., Bossert, M. & Breitbach, M. On iterative soft-decision decoding of linear binary block codes and product codes. *IEEE J. Sel. Areas Commun.* **16**, 276–296 (1998).
48. Kuo, K.-Y. & Lai, C.-Y. Exploiting degeneracy in belief propagation decoding of quantum codes. Preprint at https://arxiv.org/abs/2104.13659 (2021).
49. Hopfield, J. J. Neurons with graded response have collective computational properties like those of two-state neurons. *Proc. Nat. Acad. Sci. USA* **81**, 3088–3092 (1984).
50. Hopfield, J. J. & Tank, D. W. "neural" computation of decisions in optimization problems. *Biol. Cybern.* **52**, 141–152 (1985).
51. Hopfield, J. J. & Tank, D. W. Computing with neural circuits: a model. *Science* **233**, 625–633 (1986).
52. Van den Bout, D. E. & Miller, T. K. Improving the performance of the Hopfield-Tank neural network through normalization and annealing. *Biol. Cybern.* **62**, 129–139 (1989).
53. Marcus, C. M., Waugh, F. R. & Westervelt, R. M. Nonlinear dynamics and stability of analog neural networks. *Phys. D.* **51**, 234–247 (1991).
54. Hagiwara, M., Fossorier, M. P. C. & Imai, H. Fixed initialization decoding of LDPC codes over a binary symmetric channel. *IEEE Trans. Inf. Theory* **58**, 2321–2329 (2012).
55. Sutskever, I., Martens, J., Dahl, G. & Hinton, G. On the importance of initialization and momentum in deep learning. In *Proc. 30th International Conference on Machine Learning* (*ICML*) 1139–1147 (PMLR, 2013).

56. Kuo, K.-Y., Chern, I.-C. & Lai, C.-Y. Decoding of quantum data-syndrome codes via belief propagation. In *Proc. IEEE International Symposium on Information Theory* (*ISIT*) 1552–1557 (IEEE, 2021).
57. Bombin, H. & Martin-Delgado, M. A. Optimal resources for topological two-dimensional stabilizer codes: Comparative study. *Phys. Rev. A* **76**, 012305 (2007).
58. Horsman, C., Fowler, A. G., Devitt, S. & Van Meter, R. Surface code quantum computing by lattice surgery. *N. J. Phys.* **14**, 123011 (2012).
59. Yu, N., Lai, C.-Y. & Zhou, L. Protocols for packet quantum network intercommunication. In *IEEE Transactions on Quantum Eng*ineering (2021).
60. Ekert, A. & Macchiavello, C. Quantum error correction for communication. *Phys. Rev. Lett.* **77**, 2585 (1996).
61. Bennett, C. H., DiVincenzo, D. P., Smolin, J. A. & Wootters, W. K. Mixed-state entanglement and quantum error correction. *Phys. Rev. A* **54**, 3824 (1996).
62. Wootton, J. R. & Loss, D. High threshold error correction for the surface code. *Phys. Rev. Lett.* **109**, 160503 (2012).
63. Bombin, H., Andrist, R. S., Ohzeki, M., Katzgraber, H. G. & Martín-Delgado, M. A. Strong resilience of topological codes to depolarization. *Phys. Rev. X* **2**, 021004 (2012).
64. Ohzeki, M. Error threshold estimates for surface code with loss of qubits. *Phys. Rev. A* **85**, 060301 (2012).
65. Bravyi, S., Suchara, M. & Vargo, A. Efficient algorithms for maximum likelihood decoding in the surface code. *Phys. Rev. A* **90**, 032326 (2014).
66. Delfosse, N. & Nickerson, N. H. Almost-linear time decoding algorithm for topological codes. *Quantum* **5**, 595 (2021).
67. Kuo, K.-Y. & Lai, C.-Y. Comparison of 2D topological codes and their decoding performances. In *Proc. IEEE International Symposium on Information Theory* (*ISIT*) 186–191 (IEEE, 2022).
68. Nachmani, E. et al. Deep learning methods for improved decoding of linear codes. *IEEE J. Sel. Top. Signal Process.* **12**, 119–131 (2018).

ACKNOWLEDGEMENTS
CYL was supported by the National Science and Technology Council in Taiwan under Grant MOST110-2628-E-A49-007, MOST111-2628-E-A49-024-MY2, MOST111-2119-M-A49-004, and MOST111-2119-M-001-002.

AUTHOR CONTRIBUTIONS
C.-Y.L. and K.-Y.K. formulated the initial idea and developed the theory. K.-Y.K. performed the simulations. All co-authors contributed to the preparation of the manuscript.

COMPETING INTERESTS
The authors declare no competing interests.

ADDITIONAL INFORMATION
Correspondence and requests for materials should be addressed to Ching-Yi Lai.

Reprints and permission information is available at http://www.nature.com/reprints

Publisher's note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.