# Quantum-Inspired Tensor Network Methods for Quadratic Unconstrained Binary Optimization

**Iago Leal de Freitas**
Davidson School of Chemical Engineering
Purdue University
iago@iagoleal.com

**João Victor Paim de Cerqueira Melo Souza**
Davidson School of Chemical Engineering
Purdue University
jpaimdec@purdue.edu

**David E. Bernal Neira**
Davidson School of Chemical Engineering
Purdue University
dbernaln@purdue.edu

## Abstract

Conventional approaches for integer optimization show little benefit from GPU acceleration. On the other hand, the main computational problem in quantum many-body physics is the computation of eigenvectors of exponentially large, but structured, matrices. To address such problems, small dense matrix operations are employed, which are amenable to parallel computing. Thus, efficient methods that run in parallel on both CPUs and GPUs have been devised, such as the Density Matrix Renormalization Group (DMRG) algorithm. Embedding binary optimization programs into quantum systems reformulates minimization as an eigenvalue problem for which scalable, parallelizable algorithms exist. Hence, we represent these systems using matrix product operators (MPO), a tensor network. MPOs exactly represent QUBO problems in $\mathcal{O}(N^3)$ space, avoiding the exponential cost of a direct tensor representation. Our algorithm constructs the MPO for a QUBO problem and applies DMRG, which approximates the smallest eigenvalue within a certain truncation tolerance in polynomial time. We provide a methodological description of the algorithm and numerical results from our Julia library `TenSolver.jl`.

## 1 Introduction

In recent years, numerical linear algebra has seen a surge in scalable algorithms that exploit multi-core and GPU computation for matrix decompositions and eigenvalue problems [22, 43, 12]. These advances enable efficient solution of continuous optimization problems on GPUs, accelerating applications such as machine learning model training [24]. In contrast, classical methods for integer optimization, including Quadratic Unconstrained Binary Optimization (QUBO), have struggled to exploit these accelerations due to a mismatch between standard algorithms and GPU architectures [35, 6].

QUBO is a class of mathematical programming problems with applications in logistics, physics, finance, biology, and engineering [36]. QUBO is NP-hard [36] and closely related to the Ising model [30]. Because of this relation, QUBO has been a central target for quantum computing (QC), both on current devices running quantum annealing [21] and hybrid classical–quantum algorithms such as the quantum approximate optimization algorithm [13], and on future fault-tolerant quantum computers [38]. Despite promising results [9, 13], quantum computers still lack the scalability needed

39th Conference on Neural Information Processing Systems (NeurIPS 2025) Workshop: ScaleOPT: GPU-Accelerated and Scalable Optimization.

for large real-world problems [1]. However, using QC concepts to design algorithms for classical hardware yields *quantum-inspired algorithms* with strong practical performance.

Tensor networks (TN) [5] are a computational tool for representing large-scale linear algebra systems and are well suited to parallel hardware such as multicore clusters or GPUs. They have been successful in computational chemistry [46], fluid dynamics [18], and machine learning [40]. Tensor networks are also widely used to simulate quantum systems on classical hardware, including in challenges to quantum supremacy claims [42]. Variational algorithms based on Matrix Product States (MPS) and Operators (MPO) [45, 46, 51, 39] efficiently simulate many quantum many-body systems.

There is growing interest in using TNs for combinatorial optimization. The solver in [3] constructs tensor networks whose topology mirrors the original problem and that, when contracted, produce the optimal solution. To encode linear constraints, [2, 28] uses $U(1)$-symmetric MPS as part of generative models for binary optimization, while [20] represents constraints via projection operators acting as linking nodes in the network. For QUBO, [4] adapts these methods to build a polynomial-time solver for tridiagonal QUBO, and [27] uses contractions of tropical tensor networks to find ground states of the Ising model, which is equivalent to QUBO [30].

Contracting tensor networks with complex topologies is NP-hard [44]. Here, we instead convert QUBO into an eigenvalue problem for Matrix Product Operators (MPO), a class of tensor networks with linear topology and efficient contraction [33]. For these networks, powerful algorithms from computational physics [45, 46] find approximate eigenvectors in polynomial time. One such algorithm, the *Density Matrix Renormalization Group* (DMRG), relies on repeated applications of small dense matrices and is suitable for parallelization. Our implementation, `TenSolver.jl`, builds upon the Julia ecosystem [7], using `ITensors.jl` [16, 15] for tensor network manipulation and `QUBO.jl` [49] for integration with the `JuMP` [29] modeling language.

## 2 Problem description

A *Quadratic Unconstrained Binary Optimization* (QUBO) problem consists of solving

$$\min_{x\in\mathbb{B}^N} \quad x^\top Q x \quad = \quad \min_{x\in\mathbb{B}^N} \quad \sum_{i\geq j} Q_{ij} x_i x_j \tag{1}$$

where $Q \in \mathbb{R}^{N\times N}$ is a real matrix, $x$ is the binary decision variables vector, and we write $\mathbb{B} = \{0,\, 1\}$.

Notice that the inner product $x^\top Q x$ is symmetric, hence we assume $Q$ upper-triangular w.l.o.g. Also, since the decision variables are binary, $x_i = x_i^2$, and linear terms, i.e., $x_i^2 q_i = x_i q_i$ appear on the diagonal of $Q$. Despite its simple formulation, QUBO is a known NP-hard problem [36] and is equivalent to the Ising spin Model in condensed matter physics [30].

### 2.1 Tensor reformulation

In this section, we show how to reformulate a QUBO as an eigenvector problem, making it more suitable for parallelization. We convert a QUBO as in (1) from a discrete problem in $N$ binary variables into a $N$ particles quantum system, with 2-dimensional state spaces each. For this, consider the complex space $\mathbb{C}^2$ with orthonormal basis $\{|0\rangle\,,\, |1\rangle\}$. Its elements are called *qubits*. The tensor product of $N$ qubit spaces, i.e., $\bigotimes_{i=1}^N \mathbb{C}^2$, has dimension $2^N$ and an orthonormal basis indexed by all length $N$ binary strings, $|x\rangle \coloneqq |x_1\rangle \otimes \ldots \otimes |x_N\rangle$ where $x \in \mathbb{B}^N$.

We then build an operator $H\colon \bigotimes_{i=1}^N \mathbb{C}^2 \to \bigotimes_{i=1}^N \mathbb{C}^2$ whose least eigenvalue is the solution of (1). Start by defining the $2 \times 2$ Hermitian matrix $D = \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix}$ For a multivariable system, it is useful to create a shorthand $D_i$ for the operator that applies $D$ to the $i$-th qubit while keeping all others untouched, called a *single-site operator*,

$$D_i(x_1 \otimes \ldots \otimes x_i \otimes \ldots \otimes x_N) = x_1 \otimes \ldots \otimes (Dx_i) \otimes \ldots \otimes x_N.$$

These are diagonal operators who recover the $i$-th component of a basis vector, i.e., $\langle x|D_i|x\rangle = x_i$ and $\langle x|D_i|y\rangle = 0$ for $x \neq y$.

Now consider a QUBO as in (1) with matrix $Q$. We form an operator $H$ from it by substituting each variable $x_i$ with the respective operator $D_i$,

$$H = \sum\nolimits_{i\geq j} Q_{ij} D_i D_j. \tag{2}$$

Since these operators affect only one element of a tensor product, they commute with each other, i.e., $D_i D_j = D_j D_i$, and there is no ambiguity about the order of multiplication.

The operator $H$ defined in Eq. (2) is a gigantic $2^N \times 2^N$ complex matrix encoding all possible feasible solutions to the QUBO. To see this, consider a binary vector $x \in \mathbb{B}^N$, and notice that

$$\langle x|H|x\rangle = \sum\nolimits_{i\geq j} Q_{ij} \langle x|D_i D_j|x\rangle = \sum\nolimits_{i\geq j} Q_{ij} x_i x_j = x^\top Q x.$$

The solution to the QUBO corresponds to this operator's least eigenvalue, and the associated eigenspace is spanned by its optimal solutions. To show this, note that $H$ is Hermitian; thus, its smallest eigenvalue equals the minimum of its Rayleigh-Ritz quotient [25]. Writing $|\psi\rangle = \sum_{x\in\mathbb{B}^N} \psi_x |x\rangle$,

$$\lambda_1(H) = \min_{\|\psi\|=1} \langle\psi|H|\psi\rangle = \min_{\|\psi\|=1} \sum\nolimits_{x\in\mathbb{B}^N} |\psi_x|^2 \langle x|H|x\rangle = \min_{\|\psi\|=1} \sum\nolimits_{x\in\mathbb{B}^N} |\psi_x|^2 (x^\top Q x).$$

Since all weights $|\psi_x|^2$ are non-negative and must add up to 1, the minimum is achieved by choosing only those $x$ that minimize $x^\top Q x$. Thus, we turn the discrete optimization problem in (1) into the continuous linear algebra problem (2), for which parallel solvers can be employed.

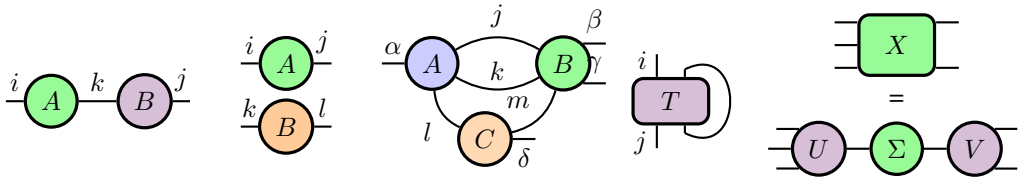
For now, it may be hard to see the benefit of converting a discrete problem into an exponentially larger continuous one. Nevertheless, in Section 3.2, we perform a decomposition of $H$ requiring only $\mathcal{O}(N^3)$ complex coefficients — a representation only a factor of $N$ larger than the $Q$ matrix itself.

### 2.2 Quantum system interpretation

We can also interpret the operator $H$ defined in Eq. (2) as the Hamiltonian of an $N$- particle quantum system with measurable states $|0\rangle$ and $|1\rangle$. In this view, $H$ encodes all possible energies for this system, and finding its ground state recovers the minimum energy for its classical counterpart. Furthermore, the probabilistic interpretation of quantum mechanics lets us use a normalized eigenvector $|x^\star\rangle$ associated with the ground state of $H$ as a probability distribution over optimal binary vectors in the QUBO. In this way, sampling from it yields optimal solutions to the original problem.

## 3 Classical representation of quantum systems as tensor networks

A general state in an $N$-qubit system requires $2^N$ degrees of freedom. For any reasonably large $N$, this is too large, even for the best available hardware. In many cases, however, these states are sparse and do not actually require all such information. To take advantage of this, we can rewrite these sparse tensors in a more amenable representation.




(a) $A_i^k B_k^j$ (b) $A_i^j B_k^l$ (c) $A_\alpha^{jkl} B_{jkm}^{\beta\gamma} C_l^{m\delta}$ (d) $T_{ik}^{jk}$ (e) SVD

Figure 1: Graphical representation of tensor network operations and their equivalent in Einstein summation notation. We add the wire indexes for legibility, but they may be omitted when there is no ambiguity. (a) Contraction amounts to connecting two wires. (b) The tensor product is juxtaposition without connecting any wires. (c) Complicated tensor contractions require more complex network topologies. (d) Partial traces amount to closing wires into loops. (e) A tensor can be decomposed into a network by reshaping it into a matrix and applying a decomposition, such as the SVD.

A *tensor network* is an unevaluated tensor contraction. It consists of a graph in which each degree-$k$ node is a rank-$k$ tensor, and each edge has an associated dimension that represents a contraction between the respective indices. It is beyond the scope of this article to explain the algebra of tensor networks, but we direct the interested reader to [5, 8]. See Figure 1 for examples of tensor contractions and the graphical representation as tensor networks. Usually, a network representation of a tensor

allows for better taking advantage of hidden topological properties that might not be straightforward when looking at it as a single tensor. As an example, large but sparse tensors can often be written as contractions of small tensors, enabling significant reductions in their storage requirements and related computational costs [26].

Even when a tensor is not sparse, it can be approximated by a contraction through its Singular Value Decomposition (SVD). See Figure 1e for a graphical representation of the process. The tensor $T$ is factored into a contraction where the middle matrix $\Sigma$ contains its singular values. By neglecting all singular values below a specific cutoff, this factorization approximates a tensor into a contraction with an arbitrarily small linking dimension.

### 3.1 Matrix product states and operators

In general, for a vector space of $d$ dimensions, a rank $k$ tensor contains $d^k$ degrees of freedom. Using the SVD factorization, it is possible to approximate it by a network containing $\mathcal{O}(kd)$ degrees of freedom. In a quantum setting, this means going from exponential to linear complexity with respect to the number of particles.

A *Matrix Product State* (MPS) is a sequential tensor network, consisting of a linear contraction of rank 3 tensors. Their open indices, called *sites*, are identical to the tensor they represent, but their linking indices are virtual and can have any dimension. Across the edges of an MPS, the maximum dimension for a linking contraction is called *bond dimension* and is an important measure of complexity for MPS algorithms [32]. They are of particular interest because, due to their absence of loops, these tensor networks support algorithms for tensor addition and contraction that are polynomial in the number of sites and bond dimension. Additionally, when viewed as probability distributions, there are efficient algorithms for perfectly sampling from MPS [14], an essential step in Section 4.

Although MPS represent vectors, their analogues for operators are the *Matrix Product Operators* (MPOs). These are sequential tensor networks composed of rank-4 tensors. Besides the MPS operations, this class of tensor networks also has fast polynomial eigensolvers [46]. See Figure 2 for the tensor network representation of such operators.

(a) Matrix Product State. (b) Matrix Product Operator.

Figure 2: Matrix Product States and Operators are tensors represented by linear graphs. When the dimensions of the contractions are bounded, their degrees of freedom are polynomial in the number of open wires.

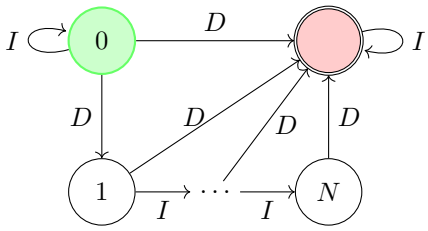
Our particular interest in MPS comes from the *Density Matrix Renormalization Group* (DMRG) algorithm, first introduced in [45, 46]. It is a variational eigensolver based on the methods of Lanczos [23] and Davidson [11], adapted for the MPS structure. It estimates an MPS representing the ground state (least eigenvector) of an operator in MPO form. Its computational complexity and memory requirements scale, respectively, as $\mathcal{O}(\kappa^3 N^4)$ and $\mathcal{O}(\kappa^2 N^2)$, where $\kappa$ is the MPO's maximum bond dimension [31]. DMRG is known to converge to the least eigenvalue when provided with a "good" initial guess. Nevertheless, it is a local optimization method and may get stuck if started too close to other eigenspaces. DMRG can be run on multiple CPUs or GPUs and scales properly until the number of cores reaches the size of the MPO [50].

### 3.2 MPO representation of QUBO

Now we focus our attention on how to represent Eq. (2) as an MPO. The straightforward approach is to construct the tensor $H$ and convert it into an MPO via a sequence of SVDs. As it stands, it is too computationally expensive to write the full tensor. Fortunately, the polynomial nature of the problem allows us to employ the procedures in [34, 10, 37] to convert a sum of local operators into an MPO. With this approach, we obtain an MPO with at most bond dimension $N + 1$.

**Theorem 1.** *There is an MPO representation for the Hamiltonian in Eq.* (2) *with maximum bond dimension* $N + 1$.

The full proof of this theorem is beyond the scope of this paper. A proof sketch is: The procedure in [34] constructs a finite automaton for recognizing only the terms over the alphabet $\{I, D\}$ that appear on the sum (2). It is then possible to appropriately convert this automaton's transition matrix into MPOs, each constituting a tensor with bond dimension equal to the number of states in the automaton. Figure 3 shows the transitions constructing the MPO for a QUBO.




(a) Finite automaton.

$$W^{(i)} = \begin{bmatrix} I & D & 0 & 0 & \dots & 0 & Q_{ii}D \\ 0 & 0 & I & 0 & \dots & 0 & Q_{(i-1)i}D \\ 0 & 0 & 0 & I & \dots & 0 & Q_{(i-2)i}D \\ \vdots & \vdots & \vdots & 0 & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & 0 & \dots & I & Q_{(i-N)i}D \\ 0 & 0 & 0 & 0 & \dots & 0 & I \end{bmatrix}$$

(b) Transition matrix for $i$-th site.

Figure 3: (a) This system starts in the green state and accepts any string that reaches the purple state. These are precisely the length $N$ strings consisting of all $I$ except for either one or two $D$ matrices.(b)The transition matrix for this finite automaton generates the $i$-th tensor for the MPO representing Eq. (2). We view it as a matrix whose components are $2 \times 2$ matrices. The constants $Q$ can be chosen independently from the automaton's structure, and, in order to reproduce the original system, we consider that transitioning from state $k$ to the accepting state produces the term $Q_{(i-k)i}D_{i-k}D_i$. These matrices can be reshaped to rank 4 tensors whose contraction equals $H$.

Note that, as a consequence of Theorem 1, we only need $\mathcal{O}(N^3)$ coefficients to *exactly* represent a QUBO problem as a tensor network.

## 4 Algorithm methodology

We propose a tensor network-based algorithm for solving QUBO. It consists of converting a QUBO problem into a quantum system represented by an MPO using the procedure from Section 3.2, and then applying DMRG, which can be parallelized efficiently. The final output is an MPS $\psi$ representing its ground state. If DMRG converges, $\psi$ will be a (quantum) probability distribution over all binary vectors that minimizes the original problem. If the algorithm is stopped before convergence, $\psi$ can still be treated as a probability distribution over feasible solutions that approximate the true minimum. The steps are represented in Algorithm 1.

**Algorithm 1** Tensor network-based QUBO solver.

**Input:** A square matrix $Q$, an initial MPS guess $|\psi_0\rangle$
**Parameters:** SVD cutoff $\epsilon$, maximum bond dimension $\kappa$
$H \leftarrow \mathrm{MPO}(Q)$
$|\psi\rangle \leftarrow |\psi_0\rangle$
**repeat**
  $E, |\psi\rangle \leftarrow \mathrm{DMRG}(H, |\psi\rangle)$ ▷ Better MPS approximation
**until** $H|\psi\rangle \approx E|\psi\rangle$
$x^\star \leftarrow \mathrm{sample}(|\psi\rangle)$
**return** $E$, $x^\star$ ▷ $E \approx (x^\star)^\top Q x^\star$ approximates the minimum

Notice that, while the MPO representation is exact, the variational eigensolver is an approximate method that may require truncation of the solution. Generally, there is a controllable trade-off between accuracy and speed when deciding how much to simplify the original problem. This is expected since QUBO is NP-hard while all steps in here are polynomial.

## 5 Numerical results

In this section, we evaluate the algorithm's performance on multi-core CPUs in a series of QUBO problems and compare it to the commercial mixed-integer solver Gurobi [19] version 12.0. We bench-

mark the solver using QUBO instances from the QPLib dataset [17], obtained via QUBOLib [48]. The results are shown in Fig. 4.

All runs were performed using the Anvil supercomputer [41]. The CPU benchmarks used 32 cores of an AMD EPYC™ 7763 CPU, while the GPU benchmarks used a single NVIDIA A100 GPU. The proposed algorithm is implemented as the package `TenSolver.jl`[1] in Julia [7] and builds upon `ITensors.jl` [16, 15] and `QUBO.jl` [49].

The runs had no fixed iterations for convergence. All applied SVDs had a singular value cut-off of $10^{-7}$. We start with a maximum bond dimension of $40$ for the solution, but allow it to increase linearly with the iterations. Following [47], a small amount of noise is applied to the density matrices throughout the calculations. We choose to start with $10^{-5}$ and decrease it exponentially with each iteration. A $10^{-4}$ threshold was used to assess solver convergence. As the initial state for DMRG, we use a random MPS of bond dimension $40$. [2]

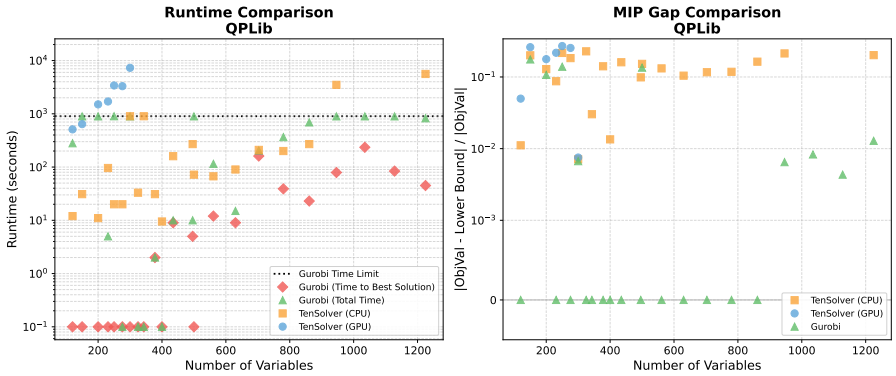



Figure 4: Comparison between Gurobi and TenSolver. (Left) Runtime across devices and variable counts. (Right) MIP Gap computed with respect to the Gurobi lower bound.

From the results in Fig. 4, we observe that the results were comparable to those of the commercial solver Gurobi running on CPU.

## 6 Conclusions and Future Work

We present a quantum-inspired algorithm for approximately solving Quadratic Unconstrained Binary Optimization (QUBO) problems using Tensor Networks (TNs). The method maps QUBOs to a Matrix Product Operator (MPO) and solves them with the Density Matrix Renormalization Group (DMRG), where the MPO bond dimension grows only polynomially with system size, ensuring scalability. Reformulating the QUBO as a Hamiltonian enables parallelism.

We implement the approach in the open-source package `TenSolver.jl` and demonstrate it on a benchmark QUBO instances, achieving competitive performance with a commercial MIP solver on multi-core CPUs. The physics-inspired representation naturally supports efficient parallelization in CPU and GPU, addressing a major challenge in discrete optimization.

Future work will focus on improving the algorithm's suitability for GPU environments and on further exploring the solver's parallelization potential. Additionally, pre-processing techniques are planned to enhance convergence rates.

## References

[1] Amira Abbas, Andris Ambainis, Brandon Augustino, Andreas Bärtschi, Harry Buhrman, Carleton Coffrin, Giorgio Cortiana, Vedran Dunjko, Daniel J Egger, Bruce G Elmegreen, et al.

[1] Available in `https://github.com/SECQUOIA/TenSolver.jl`

[2] The instances and results presented herein are available in `https://github.com/SECQUOIA/qplib_tensolver`.

Challenges and opportunities in quantum optimization. *Nature Reviews Physics*, pages 1–18, 2024.

[2] Javier Alcazar, Mohammad Ghazi Vakili, Can B. Kalayci, and Alejandro Perdomo-Ortiz. GEO: Enhancing Combinatorial Optimization with Classical and Quantum Generative Models, June 2022. arXiv:2101.06250 [quant-ph].

[3] Alejandro Mata Ali. Explicit Solution Equation for Every Combinatorial Problem via Tensor Networks: MeLoCoToN, February 2025. arXiv:2502.05981 [cs].

[4] Alejandro Mata Ali, Iñigo Perez Delgado, Marina Ristol Roura, and Aitor Moreno Fdez de Leceta. Polynomial-time Solver of Tridiagonal QUBO and QUDO problems with Tensor Networks, April 2024. arXiv:2309.10509 [quant-ph].

[5] Aleksandr Berezutskii, Atithi Acharya, Roman Ellerbrock, Johnnie Gray, Reza Haghshenas, Zichang He, Abid Khan, Viacheslav Kuzmin, Minzhao Liu, Dmitry Lyakh, Danylo Lykov, Salvatore Mandrà, Christopher Mansell, Alexey Melnikov, Artem Melnikov, Vladimir Mironov, Dmitry Morozov, Florian Neukart, Alberto Nocera, Michael A. Perlin, Michael Perelshtein, Ruslan Shaydulin, Benjamin Villalonga, Markus Pflitsch, Marco Pistoia, Valerii Vinokur, and Yuri Alexeev. Tensor networks for quantum computing, March 2025. arXiv:2503.08626 [quant-ph].

[6] David E Bernal Neira, Carl D Laird, Laurens R Lueg, Stuart M Harwood, Dimitar Trenev, and Davide Venturelli. Utilizing modern computer architectures to solve mathematical optimization problems: A survey. *Computers & Chemical Engineering*, 184:108627, 2024.

[7] Jeff Bezanson, Alan Edelman, Stefan Karpinski, and Viral B Shah. Julia: A fresh approach to numerical computing. *SIAM review*, 59(1):65–98, 2017.

[8] Jacob Biamonte and Ville Bergholm. Tensor Networks in a Nutshell, July 2017. arXiv:1708.00006 [cond-mat, physics:gr-qc, physics:hep-th, physics:math-ph, physics:quant-ph].

[9] Sergio Boixo, Troels F Rønnow, Sergei V Isakov, Zhihui Wang, David Wecker, Daniel A Lidar, John M Martinis, and Matthias Troyer. Evidence for quantum annealing with more than one hundred qubits. *Nature physics*, 10(3):218–224, 2014.

[10] Benjamin Corbett and Akimasa Miyake. Scaling up the transcorrelated density matrix renormalization group, 2025.

[11] Ernest R. Davidson. The iterative calculation of a few of the lowest eigenvalues and corresponding eigenvectors of large real-symmetric matrices. *Journal of Computational Physics*, 17(1):87–94, January 1975.

[12] Jack Dongarra, Mark Gates, Azzam Haidar, Jakub Kurzak, Piotr Luszczek, Stanimire Tomov, and Ichitaro Yamazaki. Accelerating numerical dense linear algebra calculations with gpus. In *Numerical computations with GPUs*, pages 3–28. Springer, 2014.

[13] Edward Farhi, Jeffrey Goldstone, and Sam Gutmann. A quantum approximate optimization algorithm. *arXiv preprint arXiv:1411.4028*, 2014.

[14] Andrew J. Ferris and Guifre Vidal. Perfect sampling with unitary tensor networks. *Physical Review B*, 85(16):165146, April 2012.

[15] Matthew Fishman, Steven R. White, and E. Miles Stoudenmire. Codebase release 0.3 for ITensor. *SciPost Phys. Codebases*, pages 4–r0.3, 2022.

[16] Matthew Fishman, Steven R. White, and E. Miles Stoudenmire. The ITensor Software Library for Tensor Network Calculations. *SciPost Phys. Codebases*, page 4, 2022.

[17] Fabio Furini, Emiliano Traversi, Pietro Belotti, Antonio Frangioni, Ambros Gleixner, Nick Gould, Leo Liberti, Andrea Lodi, Ruth Misener, Hans Mittelmann, Nikolaos Sahinidis, Stefan Vigerske, and Angelika Wiegele. QPLIB: A library of quadratic programming instances. *Mathematical Programming Computation*, 2018.

[18] Nikita Gourianov, Peyman Givi, Dieter Jaksch, and Stephen B Pope. Tensor networks enable the calculation of turbulence probability distributions. *Science Advances*, 11(5):eads5990, 2025.

[19] Gurobi Optimization, LLC. Gurobi Optimizer Reference Manual, 2024.

[20] Tianyi Hao, Xuxin Huang, Chunjing Jia, and Cheng Peng. A Quantum-Inspired Tensor Network Algorithm for Constrained Combinatorial Optimization Problems. *Frontiers in Physics*, 10:906590, July 2022.

[21] Tadashi Kadowaki and Hidetoshi Nishimori. Quantum annealing in the transverse ising model. *Physical Review E*, 58(5):5355, 1998.

[22] Jens Krüger and Rüdiger Westermann. Linear algebra operators for gpu implementation of numerical algorithms. In *ACM SIGGRAPH 2005 Courses*, pages 234–es. Association for Computing Machinery, 2005.

[23] C. Lanczos. An iteration method for the solution of the eigenvalue problem of linear differential and integral operators. *Journal of Research of the National Bureau of Standards*, 45(4):255, October 1950.

[24] Sören Laue, Mark Blacher, and Joachim Giesen. Optimization for classical machine learning problems on the gpu. In *Proceedings of the AAAI conference on artificial intelligence*, volume 36, pages 7300–7308, 2022.

[25] Peter D. Lax. *Linear algebra and its applications*. Pure and applied mathematics. Wiley-Interscience, Hoboken, N.J, 2nd ed edition, 2007. OCLC: ocn138342469.

[26] Jiawen Liu, Jie Ren, Roberto Gioiosa, Dong Li, and Jiajia Li. Sparta: High-performance, element-wise sparse tensor contraction on heterogeneous memory. In *Proceedings of the 26th ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming*, pages 318–333, 2021.

[27] Jin-Guo Liu, Lei Wang, and Pan Zhang. Tropical Tensor Network for Ground States of Spin Glasses. *Physical Review Letters*, 126(9):090506, March 2021. arXiv:2008.06888 [cond-mat, physics:quant-ph, stat].

[28] Javier Lopez-Piqueres, Jing Chen, and Alejandro Perdomo-Ortiz. Symmetric Tensor Networks for Generative Modeling and Constrained Combinatorial Optimization, July 2023. arXiv:2211.09121 [quant-ph].

[29] Miles Lubin, Oscar Dowson, Joaquim Dias Garcia, Joey Huchette, Benoît Legat, and Juan Pablo Vielma. JuMP 1.0: Recent improvements to a modeling language for mathematical optimization. *Mathematical Programming Computation*, 2023.

[30] Andrew Lucas. Ising formulations of many NP problems. *Frontiers in Physics*, 2, 2014.

[31] Andor Menczer, Maarten van Damme, Alan Rask, Lee Huntington, Jeff Hammond, Sotiris S. Xantheas, Martin Ganahl, and Örs Legeza. Parallel implementation of the density matrix renormalization group method achieving a quarter petaflops performance on a single dgx-h100 gpu node. *Journal of Chemical Theory and Computation*, 20(19):8397–8404, 2024. PMID: 39297788.

[32] Román Orús. A practical introduction to tensor networks: Matrix product states and projected entangled pair states. *Annals of physics*, 349:117–158, 2014.

[33] I. V. Oseledets. Tensor-Train Decomposition. *SIAM Journal on Scientific Computing*, 33(5):2295–2317, 2011. _eprint: https://doi.org/10.1137/090752286.

[34] Sebastian Paeckel, Thomas Köhler, and Salvatore R. Manmana. Automated construction of $U(1)$-invariant matrix-product operators from graph representations. *SciPost Physics*, 3(5), November 2017. Publisher: Stichting SciPost.

[35] Kalyan Perumalla and Maksudul Alam. Design considerations for gpu-based mixed integer programming on parallel computing platforms. In *50th International Conference on Parallel Processing Workshop*, pages 1–7, 2021.

[36] Abraham P Punnen. The quadratic unconstrained binary optimization problem. *Springer International Publishing*, 10:978–3, 2022.

[37] Jiajun Ren, Weitang Li, Tong Jiang, and Zhigang Shuai. A general automatic method for optimal construction of matrix product operators using bipartite graph theory. *The Journal of Chemical Physics*, 153(8), August 2020. Publisher: AIP Publishing.

[38] Yuval R Sanders, Dominic W Berry, Pedro CS Costa, Louis W Tessler, Nathan Wiebe, Craig Gidney, Hartmut Neven, and Ryan Babbush. Compilation of fault-tolerant quantum heuristics for combinatorial optimization. *PRX quantum*, 1(2):020312, 2020.

[39] Ulrich Schollwoeck. The density-matrix renormalization group in the age of matrix product states. *Annals of Physics*, 326(1):96–192, January 2011. arXiv:1008.3477 [cond-mat].

[40] Richik Sengupta, Soumik Adhikary, Ivan Oseledets, and Jacob Biamonte. Tensor networks in machine learning. *European Mathematical Society Magazine*, (126):4–12, 2022.

[41] X. Carol Song, Preston Smith, Rajesh Kalyanam, Xiao Zhu, Eric Adams, Kevin Colby, Patrick Finnegan, Erik Gough, Elizabett Hillery, Rick Irvine, Amiya Maji, and Jason St. John. Anvil - system architecture and experiences from deployment and early user operations. In *Practice and Experience in Advanced Research Computing 2022: Revolutionary: Computing, Connections, You*, PEARC '22, New York, NY, USA, 2022. Association for Computing Machinery.

[42] Joseph Tindall, Antonio Mello, Matt Fishman, Miles Stoudenmire, and Dries Sels. Dynamics of disordered quantum systems with two- and three-dimensional tensor networks, 2025.

[43] Stanimire Tomov, Rajib Nath, Hatem Ltaief, and Jack Dongarra. Dense linear algebra solvers for multicore with gpu accelerators. In *2010 IEEE International Symposium on Parallel & Distributed Processing, Workshops and Phd Forum (IPDPSW)*, pages 1–8. IEEE, 2010.

[44] F. Verstraete, M. M. Wolf, D. Perez-Garcia, and J. I. Cirac. Criticality, the area law, and the computational power of projected entangled pair states. *Phys. Rev. Lett.*, 96:220601, Jun 2006.

[45] Steven R. White. Density matrix formulation for quantum renormalization groups. *Phys. Rev. Lett.*, 69:2863–2866, Nov 1992.

[46] Steven R. White. Density-matrix algorithms for quantum renormalization groups. *Phys. Rev. B*, 48:10345–10356, Oct 1993.

[47] Steven R. White. Density matrix renormalization group algorithms with a single center site. *Phys. Rev. B*, 72:180403, Nov 2005.

[48] Pedro Maciel Xavier and David E. Bernal Neira. QUBOLib.jl: A Julia library of QUBO instances, 2023. Accessed: 2025-11-21.

[49] Pedro Maciel Xavier, Pedro Ripper, Tiago Andrade, Joaquim Dias Garcia, Nelson Maculan, and David E. Bernal Neira. QUBO.jl: A Julia Ecosystem for Quadratic Unconstrained Binary Optimization, July 2023. arXiv:2307.02577 [quant-ph].

[50] Chunyang Xiang, Weile Jia, Wei-Hai Fang, and Zhendong Li. Distributed multi-gpu ab initio density matrix renormalization group algorithm with applications to the p-cluster of nitrogenase. *Journal of Chemical Theory and Computation*, 20(2):775–786, 2024.

[51] Mingru Yang and Steven R. White. Time Dependent Variational Principle with Ancillary Krylov Subspace. *Physical Review B*, 102(9):094315, September 2020. arXiv:2005.06104 [cond-mat, physics:quant-ph].