SoftwareX 31 (2025) 102257

Contents lists available at ScienceDirect

SoftwareX

journal homepage: www.elsevier.com/locate/softx

Original software publication

SpinGlassPEPS.jl: Tensor-network package for Ising-like optimization on quasi-two-dimensional graphs

Tomasz Śmierzchalski [a], Anna M. Dziubyna [b,c], Konrad Jałowiecki [a], Zakaria Mzaouali [a,d], Łukasz Pawela [a,*], Bartłomiej Gardas [a], Marek M. Rams [b]

[a] Institute of Theoretical and Applied Informatics, Polish Academy of Sciences, Bałtycka 5, 44-100 Gliwice, Poland
[b] Jagiellonian University, Institute of Theoretical Physics, Łojasiewicza 11, 30-348 Kraków, Poland
[c] Jagiellonian University, Doctoral School of Exact and Natural Sciences, Łojasiewicza 11, 30-348, Kraków, Poland
[d] Institut für Theoretische Physik, Universität Tübingen, Auf der Morgenstelle 14, 72076, Tübingen, Germany

ARTICLE INFO

Keywords:
Spin-glass problems
Tensor network contractions
QUBO
Ising model
Random Markov field

ABSTRACT

This work introduces `SpinGlassPEPS.jl`, a software package implemented in Julia, designed to find low-energy configurations of generalized Potts models, including Ising and QUBO problems, utilizing heuristic tensor network contraction algorithms on quasi-2D geometries. In particular, the package employs the Projected Entangled-Pairs States to approximate the Boltzmann distribution corresponding to the model's cost function. This enables an efficient branch-and-bound search (within the probability space) that exploits the locality of the underlying problem's topology. As a result, our software enables the discovery of low-energy configurations for problems on quasi-2D graphs, particularly those relevant to modern quantum annealing devices. The modular architecture of `SpinGlassPEPS.jl` supports various contraction schemes and hardware acceleration.

Code metadata

Current code version	v1.4.1
Permanent link to code/repository used for this code version	https://github.com/ElsevierSoftwareX/SOFTX-D-25-00078
Legal Code License	Apache 2.0
Code versioning system used	git
Software code languages, tools, and services used	Julia
Compilation requirements, operating environments & dependencies	`Julia 1.11`, `CUDA.jl v5`, `TensorOperations.jl`, `Memoize.jl`
If available, Link to developer documentation/manual	https://euro-hpc-pl.github.io/SpinGlassPEPS.jl/dev/
Support email for questions	lpawela@iitis.pl

1. Motivation and significance

`SpinGlassPEPS.jl` provides a robust software package designed to find low-energy configurations for generalized Potts models [1], including Ising [2] or, equivalently, Quadratic Unconstrained Binary Optimization (QUBO) problems. By leveraging heuristic tensor network algorithms, specifically, Projected Entangled-Pairs States (PEPS) [3–6], the package enables efficient exploration of low-energy solutions within complex problem topologies. This capability is particularly significant in the context of current quantum and classical annealing devices [7–9], where efficient and scalable solutions are essential.

The significance of `SpinGlassPEPS.jl` lies in its ability to lower the entry barrier to applying advanced tensor network (TN) methods in classical optimization by offering clean, efficient (GPU-accelerated) and modular software written in Julia. Our package functions both as a standalone Ising solver and as a source of independent tools for developing physics-inspired algorithms. It offers a modular architecture that supports various contraction schemes and hardware acceleration, making it adaptable to a wide range of problem instances. `SpinGlassPEPS.jl` is specifically tailored to solve Ising and QUBO problems on the topologies of near-term quantum annealers, such as

* Corresponding author.
E-mail address: lpawela@iitis.pl (Łu. Pawela).

https://doi.org/10.1016/j.softx.2025.102257
Received 4 February 2025; Received in revised form 25 May 2025; Accepted 30 June 2025
Available online 31 July 2025

Pegasus and Zephyr graphs [10,11]. The software has been extensively benchmarked [12] on problems defined on these geometries.

2. Software description

`SpinGlassPEPS.jl` is a collection of Julia [13] packages implementing heuristic tensor-network based algorithm to find low-energy states and their corresponding energies (i.e., the spectrum) of generalized Potts model,

$$E(\boldsymbol{x}) = \sum_{\langle m,n\rangle \in \mathcal{F}} E_{m,n}(x_m, x_n) + \sum_{n \in \mathcal{W}} E_n(x_n), \tag{1}$$

defined on a graph $\mathcal{G} = (\mathcal{W}, \mathcal{F})$ specified by its edges, $\mathcal{F}$, and vertices, $\mathcal{W}$. The method tackles a family of sparse two-dimensional graphs called king's graphs [14], see Fig. 1(b). A particular problem instance is defined by real-valued functions, $E_n(x_n)$ and $E_{m,n}(x_m, x_n)$, where n, $m \in \mathcal{W}$.

In particular, this includes the Ising model,

$$E(\boldsymbol{s}) = \sum_{\langle i,j\rangle \in \mathcal{E}} J_{ij} s_i s_j + \sum_{i \in \mathcal{V}} h_i s_i, \tag{2}$$

defined on $\mathcal{G}' = (\mathcal{V}, \mathcal{E})$, where $i, j \in \mathcal{V}$, $J_{ij}, h_i \in \mathbb{R}$ and $s_i \in \{-1, 1\}$. The Ising model can be considered a special case of the Potts model, and we extend this by allowing clusters of spin variables to be grouped into effective Potts degrees of freedom with higher dimensions. This approach enables the package to manage more complex quasi-2D graph geometries, including those relevant to current quantum and classical annealing device architectures.

The algorithm executes a branch-and-bound search [15] within the probability space defined by the Boltzmann distribution at a specific inverse temperature β. This search process involves constructing a predefined number of the most probable local configurations of Potts variables, progressing sequentially from one vertex to the next until the entire system has been explored. The emphasis on king's graphs is incorporated into the algorithm at two distinct levels.

First, in calculating marginal probabilities, we employ a tensor network representation of the classical Boltzmann distribution. For two-dimensional systems, this representation takes the form of PEPS. The marginal and conditional probabilities are derived from the contraction of the tensor network. Although the problem of contracting these networks is formally #P-hard [16], we utilize an established approximate (heuristic) framework to contract two-dimensional PEPS [5,6].

Furthermore, we leverage the locality of interactions within the graph to expand the search space by merging partial trial configurations that are equivalent in terms of the marginal conditional probabilities considered by the algorithm. Specifically, partial configurations with identical values of Potts variables on the boundary adjacent to unexplored regions of the graph are combined. This approach also enables the identification of information regarding local excitations in the system. When two partial configurations share boundary variables, they have a well-defined energy associated with the bulk variables, where one configuration represents a local ground state and the other an excitation localized in the bulk. By collecting such information throughout the search, the algorithm generates a candidate for the ground state along with a set of excited low-energy states, effectively characterizing the low-energy manifold of the optimization problem.

The algorithm workflow is outlined in Fig. 1. The package is designed with a modular architecture that captures the relationships between the high-level concepts employed by the algorithm. This modularity enables the integration of various contraction schemes, leveraging the internal structures of individual tensors within the network, and allows for hardware acceleration, all facilitated by the multiple dispatch capabilities of the Julia language. Detailed explanations of the algorithm's mechanics, along with extensive benchmarks for Pegasus and Zephyr geometries, relevant for D-Wave quantum annealers, are provided in [12].

2.1. Software architecture

`SpinGlassPEPS.jl` is composed of three independent sub-packages, each of which is responsible for the distinct elements of the workflow, see Fig. 2. Namely,

- `SpinGlassEngine.jl` serves as the core package, consisting of routines for executing the branch-and-bound method (with the ability to leverage the problem's locality) for a given Potts instance. It also includes capabilities for reconstructing the low-energy spectrum from identified localized excitations and provides a tensor network constructor.
- `SpinGlassNetworks.jl` facilitates the generation of an Ising graph from a given instance using a set of standard inputs (e.g., instances compatible with the Ocean environment provided by D-Wave) and supports clustering to create effective Potts Hamiltonians.
- `SpinGlassTensors.jl` offers essential tools to create and manipulate tensors that build the PEPS network, with support for CPU and GPU utilization. It manages core operations on tensor networks, including contraction, using the boundary Matrix Product State approach [5]. This package primarily functions as a backend, and users generally do not interact with it directly.

It is worth adding that all of these subpackages can be used as independent tools for developing physics-inspired algorithms or serve as a backend for other software.

2.2. Software functionalities

For Ising/QUBO problems, the package requires instances to be provided in a specific format, similar to the one used in Stanford Gset (reduced from Max-Cut) [17]. In this format, the spins are numbered 1 to N and arranged in rows as i j v, where v represents the coupling value between vertices i and j, or the local magnetic field when $i = j$.

The algorithm produces a low-energy spectrum as its output, detailing the states and their corresponding energies for the given input instance. In addition, the package provides supplementary information, such as the estimated probabilities of these states derived from the approximate contraction of the tensor network.

The package's modular structure offers a range of options to control various aspects of the main algorithm. These include the details and control parameters of the tensor network contraction schemes, the ability to utilize either CPU or GPU for low-level operations, and the choice between constructing the tensor network in a dense or sparse format. The sparse format leverages the internal structures of individual tensors and is particularly essential for handling large clusters, such as those relevant to Pegasus and Zephyr graphs. Detailed information on all required and optional parameters is available in the documentation [18].

2.3. Work division between CPU and GPU

Our package allows the user to use either CPU-only computation or to offload parts of the computation to a dedicated GPU. Currently, only one GPU is utilized. As shown in Fig. 1 the main part of the algorithm is the contraction of the constructed PEPS network. We perform this contraction using the `TensorOperations.jl` [19] package, which allows us to seamlessly switch computational backends (CPU or GPU). When in the CPU mode, the tensor expressions are converted into native Julia code run on Julia's linear algebra backend (either OpenBLAS or MKL).

In the GPU mode, the `TensorOperations.jl` package translates the tensor expressions into calls to Julia's cuTENSOR [20] wrapper. This approach allows us to achieve high levels of code reusability and readability while maintaining high performance in both CPU and GPU computation. Finally, the computations have a relatively small memory footprint. For example, all the examples run on a consumer-grade GPU with 24 GB of vRAM.

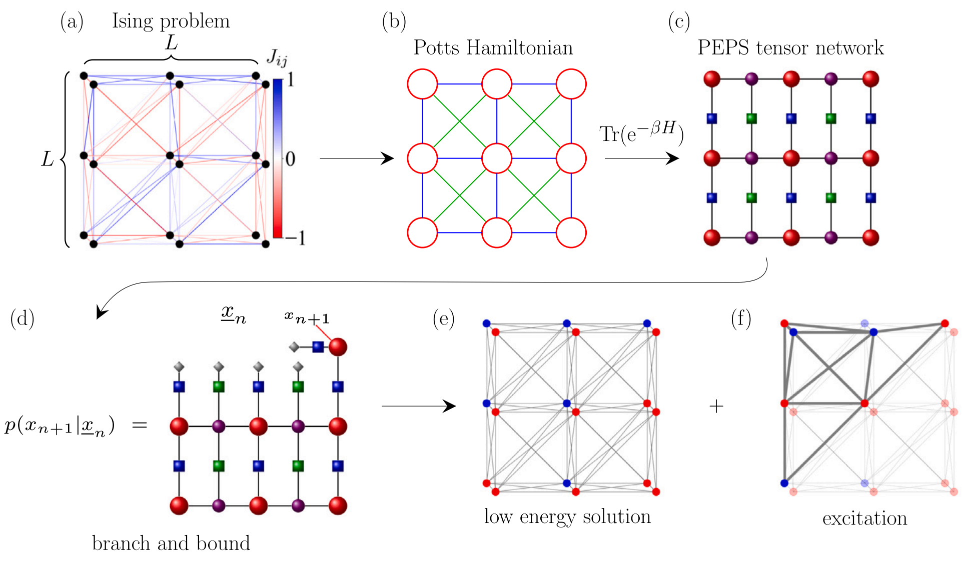

Fig. 1. Execution flow. The Ising problem in (a) is mapped to a Potts Hamiltonian defined on a king's graph in (b). This allows the partition function of that Hamiltonian to be represented as a PEPS tensor network on a square lattice, as in (c). The main algorithm executes the branch and bound search in the probability space, building the most probable configurations by adding one Potts variable at a time. The marginal conditional probabilities follow from an approximate contraction of the corresponding tensor network in (d). The full branch and bound sweep results in a candidate for the most probable (ground state) configuration in (e), together with a set of localized excitations on top of it in (f).

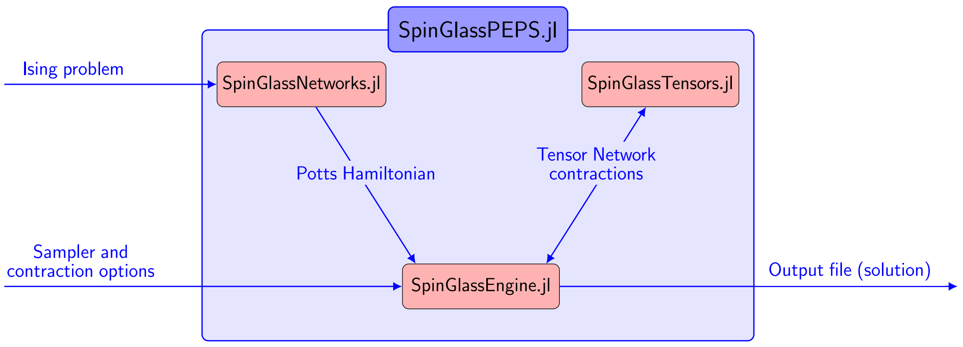

Fig. 2. Interoperability between all `SpinGlassPEPS.jl` packages. The input Ising problem file is processed by `SpinGlassNetworks.jl`, transforming it into a Potts Hamiltonian. The latter is then passed to `SpinGlassEngine.jl`, together with solver's and contraction's parameters. The `SpinGlassEngine.jl` module serves as the core branch and bound solver. It passes the problem of marginal probabilities' calculation to `SpinGlassTensors.jl`, that constructs and approximately contracts the corresponding tensor network. Finally, the solution (ground state, excitations, and their energies) is returned to the user as an output.

3. Illustrative examples

We illustrate the capabilities and modularity of `SpinGlassPEPS.jl` by addressing two distinct problems. First, we solve an Ising problem defined on a quasi-2D graph, see Fig. 1(a). Although this example focuses on a specific topology, the approach is applicable to other structures, including those used in D-Wave systems. The second example tackles an inpainting problem formulated through the Potts model [21–23].

While the Ising model on a king's graph should compute relatively fast, the latter two examples (an inpainting example and a small pegasus-type graph) can take a long time to finish if run on the CPU only. It is recommended to run them using GPU.

3.1. Ising model on a king's graph

In the listing below, we show a complete Julia script to define and solve an Ising problem defined on a graph in Fig. 1(a).

```julia
using SpinGlassPEPS

function get_instance(topology::NTuple{3, Int})
   m, n, t = topology
   "$(@__DIR__)/instances/$(m)x$(n)x$(t).txt"
end
```

```julia
function run_square_diag_bench(::Type{T}; topology::NTuple{3, Int}) where {T}
   m, n, _ = topology
   instance = get_instance(topology)
   lattice = super_square_lattice(topology)

   hamming_dist = 5
   eng = 10

   best_energies = T[]

   potts_h = potts_hamiltonian(
      ising_graph(instance),
      spectrum = full_spectrum,
      cluster_assignment_rule = lattice,
   )

   params = MpsParameters{T}(; bond_dim = 16, num_sweeps = 1)
   search_params = SearchParameters(; max_states = 2^8, cut_off_prob = 1E-4)

   for transform in all_lattice_transformations
      net = PEPSNetwork{KingSingleNode{GaugesEnergy}, Dense, T}(
         m, n, potts_h, transform,
      )

      ctr = MpsContractor(SVDTruncate, net, params;
         onGPU = false, beta = T(2), graduate_truncation = true,
      )

      single = SingleLayerDroplets(eng, hamming_dist, :hamming)
      merge_strategy = merge_branches(
         ctr; merge_type = :nofit, update_droplets = single,
      )

      sol, _ = low_energy_spectrum(ctr, search_params, merge_strategy)
```

```
        push!(best_energies, sol.energies[1])
        clear_memoize_cache()
    end

    ground = best_energies[1]
    @assert isapprox(ground, best_energies)

    println("Best energy found: $(ground)")
end

T = Float64
@time run_square_diag_bench(T; topology = (3, 3, 2))
```

In the example above, `transform` specifies a rotation of the quasi-2D graph. The branch-and-bound search operates with a fixed order of sweeping through local variables, corresponding to the fixed order of tensor network contraction. By rotating the network, the search and contraction can effectively begin from different starting points on the 2D grid, thereby enhancing the stability of the results. This example performs the search across all eight possible transformations of the 2D grid, comparing the best energies obtained for each configuration.

Other control parameters include `Sparsity`, which determines whether dense or sparse tensors should be used. The concept of sparse tensors was introduced to manage large clusters containing $\mathcal{O}(10-20)$ Ising variables (*i.e.*, spins) each, where explicit construction of PEPS tensors, triggered by `Sparsity=Dense`, quickly becomes infeasible. Conversely, `Sparsity=Sparse` circumvents the need for direct construction of individual tensors by performing optimal contractions on small tensor diagrams utilizing internal structure of individual tensors. These diagrams are then combined to efficiently contract the entire network. Finally, `Node = KingSingleNode{GaugesEnergy}` specifies the type of the node used within the tensor networks (e.g., switching between king's graph or a square lattice); `Layout = GaugesEnergy` denotes the division of the PEPS network into boundary Matrix Product States [5,6] used to contract the network.

Low-energy excitations (*i.e.*, droplets) above the best solution can be identified during the optional `merge_branches` step. This option can be provided as an argument to the function that executes the branch-and-bound algorithm, `low_energy_spectrum` (see [12] for extended discussion). To view all droplets found, one may invoke `unpack_droplets(solution)`.

The algorithm searches for diverse excitations within a specified energy range above the ground state. An excitation is accepted only if its Hamming distance from any previously identified excitation exceeds a predefined threshold. This is governed by the parameters `energy_cutoff`, which sets the maximum allowed energy above the ground state, and `hamming_cutoff`, which determines the minimum Hamming distance required between excitations for them to be considered distinct.

We present a selected set of benchmark results in Fig. 3, focusing on two problem sets with 2500 and 5000 spins (a 50×50 grid with 1 and 2 spins per cluster, respectively). The reference results are compared against those obtained from the Simulated Bifurcation Machine (SBM) [8] and CPLEX solvers [24]. The benchmarks were performed utilizing GPU acceleration. We used NVIDIA Titan RTX and GeForce RTX 3090, both of which have 24 GB of VRAM. CPLEX results were obtained on a Intel Core i9-10920X 12 core, 24 thread CPU.

For an instance with 2500 spins, `SpinGlassPEPS.jl` successfully found the ground states for all cases, as certified by CPLEX, outperforming SBM in these tests. This demonstrates that, in certain scenarios (e.g., king's graph), our approach can exceed state-of-the-art methods. For 5000 spins, SBM delivers the best results; however, `SpinGlassPEPS.jl` still performs better than CPLEX, which faces challenges with larger problem sizes.

Both `SpinGlassPEPS.jl` and SBM can output multiple solutions (*i.e.*, low-energy states), whereas CPLEX generates only one. When considering time to solution, our method is typically the most time-intensive, while SBM is the fastest across all tested solvers since it can be GPU-accelerated efficiently.

3.1.1. *Large unit cells*

Our package can target the graphs that can be manufactured by quantum annealing vendors (cf., D-Wave processors depicted in Fig. 4). We provide an extensive benchmark of our approach for those geometries elsewhere [12].

3.2. *Solving Potts model - inpainting problem*

The Potts model defined in Eq. (1) is general enough to describe a wide array of phenomena. One of them is the problem of inpainting in computer vision. It refers to filling in missing or corrupted parts of an image in a way that makes the reconstruction visually plausible [28]. In this example, we use a relatively small benchmarking instance given by [21,23]. It is a discretized triple junction inpainting problem, as shown in Fig. 5. The software accepts instances formatted as in OpenGM Benchmark dataset [23] with only the nearest neighbor and diagonal interactions.

```
instance = "$(@__DIR__)/instances/triplepoint4-plain-ring.h5"
# We add size of picture in pixels
potts_h = potts_hamiltonian(instance, 120, 120)
```

In this case, when searching for droplets, one should set `mode` parameter in `SingleLayerDroplets` to `:RMF`.

```
droplets = SingleLayerDroplets(; max_energy = 100, min_size = 100,
           metric = :hamming, mode=:RMF)
```

The remaining setup is analogous to the king's graph scenario described in the preceding section.

4. Impact and conclusion

Tensor networks offer a powerful suite of tools, most notably employed in quantum many-body simulations. Our package reduces the hurdles for applying these methods to classical optimization [29] by being clean, efficient (by utilizing sparse tensor structures, GPU acceleration, etc.), and modular. Building on the framework introduced in Ref. [30] – originally targeting simpler Chimera-like graphs – `SpinGlassPEPS.jl` provides a flexible software platform for a broader range of problem topologies, such as Pegasus and Zephyr graphs of the near-term quantum annealers. This is particularly relevant given the recent surge in physics-inspired Ising/QUBO solver research (cf. [31–33]). Additionally, We offer features such as droplet discovery [34, 35], Schmidt spectrum calculation [36], and energy level degeneracy analysis [37].

As such, `SpinGlassPEPS.jl` acts as a practical reference tool to develop and work with quantum and classical annealing technologies. The examples presented show that the tensor network approach can achieve high-quality solutions comparable to those of CPLEX, a state-of-the-art solver in operational research [24]. Notably, for certain king's graphs [14], our solutions surpass those from Toshiba's Simulated Bifurcation Machines (SBM [8]), which is an unexpected result that merits further investigation. This highlights a potential weakness in the SBM family of algorithms, demonstrating the practical value of our software.

Future development of `SpinGlassPEPS.jl` will focus on three main directions.

The first one is to expand available features. This includes implementing alternative contraction schemes for the PEPS network, e.g., Corner Transfer Matrix (CTM [38,39]), or expanding the available geometries of the Potts Hamiltonian, e.g., the Lechner-Hauke-Zoller architecture [40]. Also features that focus on aiding physical research, such as calculating approximations of some thermodynamical properties (e.g., free energy) [41].

Another direction is adding multi-GPU support. This part will require a careful re-examination of the underlying tensor network algorithm [12] in order to identify the most efficient way to achieve this

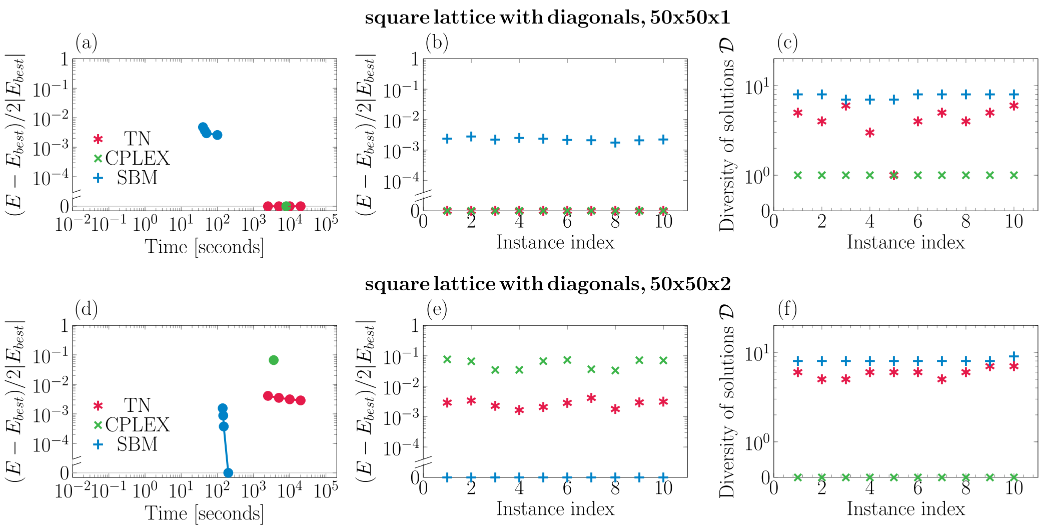

Fig. 3. Benchmarking SpinGlassPEPS.jl for two sets of Ising problems defined on graphs from Fig. 1(a) with $N = 50 \times 50 \times d$ spins, for $d = 1$ (top row) and $d = 2$ (bottom row). As reference solvers, We employ a Simulated Bifurcation machine (SBM) [25] and CPLEX. The results were obtained using GPU acceleration, where it was available. Panels (a) and (d) show time to solution to the best energy for a median instance (E_{best} is the best results among the considered solvers), and in (b) and (e), we show instance-wise results for 10 instances. In (c) and (f), we show the diversity of obtained solutions, *i.e.*, the number of solutions within approximation ratio $a_r = 0.01$ ($E - E_{\text{best}} < a_r \cdot 2 \cdot E_{\text{best}}$), where each pair has a Hamming distance greater than $N/2$.

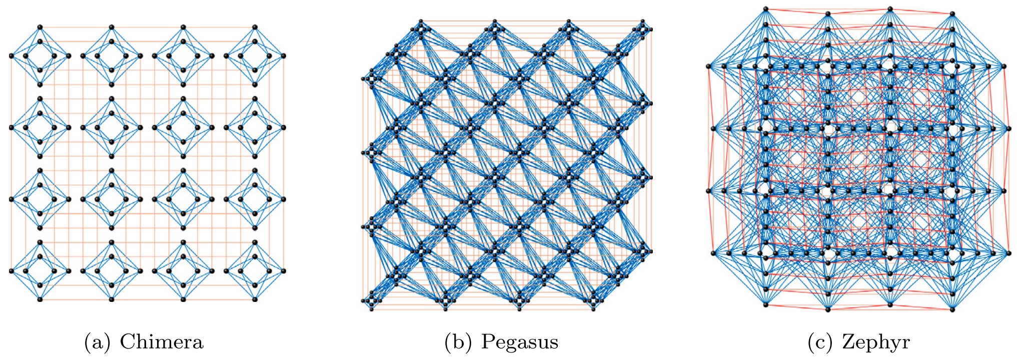

Fig. 4. Problems defined on Chimera, Pegasus, and Zephyr graphs [10,11,26,27], employed in the past and current D-Wave quantum annealers, can be mapped to the Potts Hamiltonian on king's graph upon grouping 8, 24, and 16 spins, respectively. Due to the large unit cell size of the latter two graphs, they require further processing. In particular, sparse connectivity structures between unit cells and GPU acceleration. Both are supported by SpinGlassPEPS.jl package.

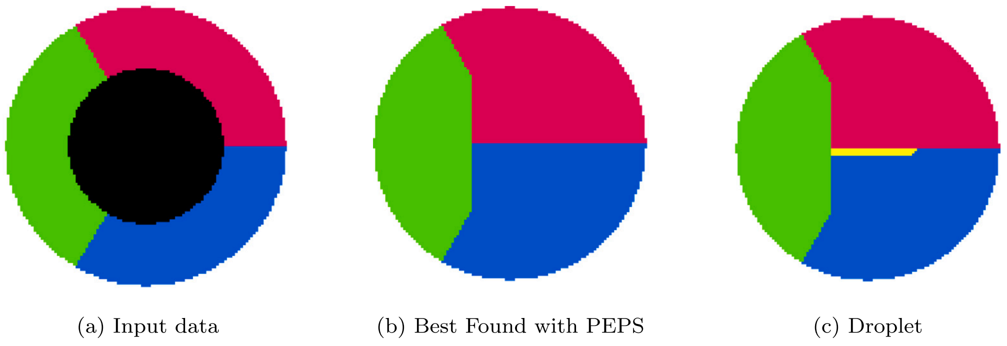

Fig. 5. The picture in panel (a) shows the used inpainting problem. Data is given on the circular boundary, and the solution should fill in the black region. It is part of the OpenGM2 Benchmark dataset [23]. The picture in panel (b) shows the ground state obtained by SpinGlassPEPS.jl. Due to the introduced anisotropy by the grid used in discretization, the results show a bias toward axis-parallel edges [21]. Yellow region in (c) shows a low-energy excitation, *i.e.*, a group of variables that should be collectively filliped to obtain another low-energy solution.

feature. This will involve researching whether multi-GPU parallelism can be achieved (i.e. processing various parts of the contraction in parallel, or can we process larger tensors sequentially on multiple GPUs?)

Lastly, a possible future feature is an implementation of the auto-embedding of instances into the relevant Potts Hamiltonian [42,43]. Currently, the user has to choose the embedding. An automatic selection of the embedding would require the development of a heuristic algorithm for choosing an appropriate embedding. Aside from coding the new functionality, this feature would require a significant amount of research to develop such heuristics.

CRediT authorship contribution statement

Tomasz Śmierzchalski: Writing – review & editing, Writing – original draft, Visualization, Validation, Software, Conceptualization. **Anna M. Dziubyna:** Writing – review & editing, Writing – original draft, Visualization, Validation, Software. **Konrad Jałowiecki:** Writing – review & editing, Writing – original draft, Software, Methodology, Investigation. **Zakaria Mzaouali:** Writing – original draft, Software, Investigation. **Łukasz Pawela:** Writing – review & editing, Writing – original draft, Validation, Supervision, Software, Resources, Project administration, Methodology, Investigation, Funding acquisition, Formal analysis, Conceptualization. **Bartłomiej Gardas:** Writing – review & editing, Writing – original draft, Supervision, Software, Resources, Project administration, Methodology, Investigation, Funding acquisition, Formal analysis, Conceptualization. **Marek M. Rams:** Writing – review & editing, Writing – original draft, Supervision, Software, Resources, Project administration, Methodology, Investigation, Funding acquisition, Formal analysis, Conceptualization.

Declaration of competing interest

The authors declare the following financial interests/personal relationships which may be considered as potential competing interests: Lukasz Pawela reports financial support was provided by Foundation for Polish Science. Bartlomiej Gardas reports financial support was provided by Foundation for Polish Science. Tomasz Smierzchalski reports financial support was provided by National Science Centre Poland. Zakaria Mzaouali reports financial support was provided by National Science Centre Poland. Marek M. Rams reports financial support was provided by National Science Centre Poland. Anna M Dziubyna reports financial support was provided by National Science Centre Poland. If there are other authors,they declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This project was supported by the National Science Center (NCN), Poland, under Projects: Sonata Bis 10, No. 2020/38/E/ST3/00269 (T.S., Z.M.) and 2020/38/E/ST3/00150 (A.D., M.R.) and Foundation for Polish Science (grant no POIR.04.04.00-00-14DE/18-00 carried out within the Team-Net program co-financed by the European Union under the European Regional Development Fund) (B.G., Ł.P.).

Appendix. Instalation and examples

The `SpinGlassPEPS.jl` can be installed using the Julia package manager for Julia v1.11. In Julia REPL type] to enter Pkg REPL, then type:

```
pkg> add SpinGlassPEPS
```

The code for all the examples presented in the paper can be found in the main repository [44] folder "examples". To run them, first, clone the repository. Then, run Julia inside the "examples" folder. The environment needed to execute the provided code is given by the `Project.toml` file. It can be activated by typing] in Julia REPL and then:

```
pkg> activate .

pkg> instantiate
```

next in Julia REPL type:

```
julia> include("ising_model_on_a_kings_graph.jl")
```

Alternatively, one can run the examples by typing:

```
$ julia --project=. -e "using Pkg; Pkg.instantiate()"

$ julia --project=. ising_model_on_a_kings_graph.jl
```

in the preferred shell.

References

[1] Kinzel W, Domany E. Critical properties of random Potts models. Phys Rev B 1981;23:3421–34. http://dx.doi.org/10.1103/PHYSREVB.23.3421.
[2] Lucas A. Ising formulations of many NP problems. Front Phys 2014;2(5). http://dx.doi.org/10.3389/fphy.2014.00005.
[3] Nishino T, Hieida Y, Okunishi K, Maeshima N, Akutsu Y, Gendiar A. Two-dimensional tensor product variational formulation. Progr Theoret Phys 2001;105(3):409–17. http://dx.doi.org/10.1143/PTP.105.409.
[4] Verstraete F, Wolf MM, Perez-Garcia D, Cirac JI. Criticality, the area law, and the computational power of projected entangled pair states. Phys Rev Lett 2006;96(22):220601. http://dx.doi.org/10.1103/PhysRevLett.96.220601.
[5] Verstraete F, Cirac JI, Murg V. Matrix Product States, Projected Entangled Pair States, and variational renormalization group methods for quantum spin systems. Adv Phys 2008;57(2):143–224. http://dx.doi.org/10.1080/14789940801912366.
[6] Orús R. A practical introduction to tensor networks: Matrix product states and projected entangled pair states. Ann Physics 2014;349:117–58. http://dx.doi.org/10.1016/j.aop.2014.06.013.
[7] King AD, et al. Quantum critical dynamics in a 5,000-qubit programmable spin glass. Nature 2023;617(7959):61–6. http://dx.doi.org/10.1038/s41586-023-05867-2.
[8] Goto H, Endo K, Suzuki M, Sakai Y, Kanao T, Hamakawa Y, et al. High-performance combinatorial optimization based on classical mechanics. Sci Adv 2021;7(6):eabe7953. http://dx.doi.org/10.1126/sciadv.abe7953.
[9] Wang J, Ebler D, Wong KYM, Hui DSW, Sun J. Bifurcation behaviors shape how continuous physical dynamics solves discrete Ising optimization. Nat Commun 2023;14(1):2510. http://dx.doi.org/10.1038/s41467-023-37695-3.
[10] Boothby K, King D, Raymond J. Zephyr topology of D-wave quantum processors. Tech. rep. 14-1056A-A, D-Wave; 2021, https://www.dwavesys.com/media/2uznec4s/14-1056a-a_zephyr_topology_of_d-wave_quantum_processors.pdf. [Accessed 31 January 2024].
[11] Dattani N, Szalay S, Chancellor N. Pegasus: The second connectivity graph for large-scale quantum annealing hardware. 2019, http://dx.doi.org/10.48550/arXiv.1901.07636, arXiv:1901.07636.
[12] Dziubyna AM, Śmierzchalski T, Gardas B, Rams MM, Mohseni M. Limitations of tensor-network approaches for optimization and sampling: A comparison to quantum and classical Ising machines. Phys. Rev. Appl. 2025;23(5):054049. http://dx.doi.org/10.1103/PhysRevApplied.23.054049.
[13] Bezanson J, Edelman A, Karpinski S, Shah VB. Julia: A fresh approach to numerical computing. SIAM Rev 2017;59(1):65–98. http://dx.doi.org/10.1137/141000671.
[14] Chang GJ. Algorithmic aspects of domination in graphs. In: Pardalos PM, Du D-Z, Graham RL, editors. Handbook of combinatorial optimization. New York, NY: Springer New York; 2013, p. 221–82. http://dx.doi.org/10.1007/978-1-4419-7997-1_26.
[15] Morrison DR, Jacobson SH, Sauppe JJ, Sewell EC. Branch-and-bound algorithms: A survey of recent advances in searching, branching, and pruning. Discrete Optim 2016;19:79–102. http://dx.doi.org/10.1016/j.disopt.2016.01.005.
[16] Schuch N, Wolf MM, Verstraete F, Cirac JI. Computational complexity of Projected Entangled Pair States. Phys Rev Lett 2007;98(14):140506. http://dx.doi.org/10.1103/PhysRevLett.98.140506.
[17] Standford University. Index of /~yyye/yyye/Gset. 2003, https://web.stanford.edu/~yyye/yyye/Gset/. [Accessed 19 May 2024].

[18] Śmiechrzalski T, Dziubyna AM, Pawela Ł, omiej Gardas B, Rams MM. SpinGlassPEPS.jl documentation. 2025, https://euro-hpc-pl.github.io/SpinGlassPEPS.jl. [Accessed 30 January 2025].

[19] Lukas Devos JH, contributors. TensorOperations.jl. 2023, http://dx.doi.org/10.5281/zenodo.3245496, URL https://github.com/Jutho/TensorOperations.jl.

[20] Besard T, Foket C, De Sutter B. Effective extensible programming: Unleashing Julia on GPUs. IEEE Trans Parallel Distrib Syst 2018. http://dx.doi.org/10.1109/TPDS.2018.2872064, arXiv:1712.03112.

[21] Lellmann J, Schnörr C. Continuous multiclass labeling approaches and algorithms. SIAM J Imaging Sci 2011;4(4):1049–96. http://dx.doi.org/10.1137/100805844.

[22] Kappes JH, Speth M, Reinelt G, Schnörr C. Towards efficient and exact MAP-Inference for large scale discrete computer vision problems via combinatorial optimization. In: 2013 IEEE conference on computer vision and pattern recognition. 2013, p. 1752–8. http://dx.doi.org/10.1109/CVPR.2013.229.

[23] Kappes JH, Andres B, Hamprecht FA, Schnörr C, Nowozin S, Batra D, et al. A comparative study of modern inference techniques for structured discrete energy minimization problems. Int J Comput Vis 2015;1–30. http://dx.doi.org/10.1007/s11263-015-0809-x.

[24] CPLEX, IBM ILOG. Users manual for CPLEX. 2024, URL https://www.ibm.com/docs/en/icos/22.1.1?topic=optimizers-users-manual-cplex.

[25] Goto H, Tatsumura K, Dixon AR. Combinatorial optimization by simulating adiabatic bifurcations in nonlinear Hamiltonian systems. Sci Adv 2019;5(4):eaav2372. http://dx.doi.org/10.1126/sciadv.aav2372.

[26] Boothby K, Bunyk P, Raymond J, Roy A. Next-generation topology of D-Wave quantum processors. Tech. rep. 14-1026A-C, D-Wave; 2019, https://www.dwavesys.com/media/jwwj5z3z/14-1026a-c_next-generation-topology-of-dw-quantum-processors.pdf. [Accessed 31 January 2024].

[27] Lanting T, et al. Entanglement in a quantum annealing processor. Phys Rev X 2014;4(2):021041. http://dx.doi.org/10.1103/PhysRevX.4.021041.

[28] Zeng Y, Lin Z, Yang J, Zhang J, Shechtman E, Lu H. High-resolution image inpainting with iterative confidence feedback and guided upsampling. In: Vedaldi A, Bischof H, Brox T, Frahm J-M, editors. Computer vision – ECCV 2020. Cham: Springer International Publishing; 2020, p. 1–17. http://dx.doi.org/10.1007/978-3-030-58529-7_1.

[29] Cichocki A. Tensor networks for big data analytics and large-scale optimization problems. 2014, http://dx.doi.org/10.48550/arXiv.1407.3124, arXiv:1407.3124.

[30] Rams MM, Mohseni M, Eppens D, Jałowiecki K, Gardas B. Approximate optimization, sampling, and spin-glass droplet discovery with tensor networks. Phys Rev E 2021;104(2):025308. http://dx.doi.org/10.1103/PhysRevE.104.025308.

[31] Schulz S, Willsch D, Michielsen K. Learning-driven annealing with adaptive Hamiltonian modification for solving large-scale problems on quantum devices. 2025, http://dx.doi.org/10.48550/arXiv.2502.21246, arXiv:2502.21246.

[32] Munoz-Bauza H, Lidar D. Scaling advantage in approximate optimization with quantum annealing. Phys Rev Lett 2025;134:160601. http://dx.doi.org/10.1103/PhysRevLett.134.160601.

[33] Pawłowski J, Tuziemski J, Tarasiuk P, Przybysz A, Adamski R, Hendzel K, et al. VeloxQ: A fast and efficient QUBO solver. 2025, http://dx.doi.org/10.48550/arXiv.2501.19221, arXiv:2501.19221.

[34] Newman CM, Stein DL. Critical droplets and replica symmetry breaking. Front Phys 2024;12. http://dx.doi.org/10.3389/fphy.2024.1473378.

[35] Shen M, Ortiz G, Liu Y-Y, Weigel M, Nussinov Z. Universal fragility of spin glass ground states under single bond changes. Phys Rev Lett 2024;132:247101. http://dx.doi.org/10.1103/PhysRevLett.132.247101.

[36] Zhou P-F, Lu Y, Wang J-H, Ran S-J. Tensor network efficiently representing Schmidt decomposition of quantum many-body states. Phys Rev Lett 2023;131:020403. http://dx.doi.org/10.1103/PhysRevLett.131.020403.

[37] Pelofske E. Biased degenerate ground-state sampling of small Ising models with converged QAOA. 2024, http://dx.doi.org/10.48550/arXiv.2411.05294, arXiv:2411.05294.

[38] Lukin IV, Sotnikov AG. Corner transfer matrix renormalization group approach in the zoo of Archimedean lattices. Phys Rev E 2024;109:045305. http://dx.doi.org/10.1103/PhysRevE.109.045305, URL https://link.aps.org/doi/10.1103/PhysRevE.109.045305.

[39] Mangazeev VV, Hagan B, Bazhanov VV. Corner transfer matrix approach to the yang-lee singularity in the two-dimensional Ising model in a magnetic field. Phys Rev E 2023;108:064136. http://dx.doi.org/10.1103/PhysRevE.108.064136, URL https://link.aps.org/doi/10.1103/PhysRevE.108.064136.

[40] Lechner W, Hauke P, Zoller P. A quantum annealing architecture with all-to-all connectivity from local interactions. Sci Adv 2015;1(9):e1500838. http://dx.doi.org/10.1126/sciadv.1500838.

[41] Nishimori H. Instability of the ferromagnetic phase under random fields in an Ising spin glass with correlated disorder. Phys Rev E 2025;111:044109. http://dx.doi.org/10.1103/PhysRevE.111.044109.

[42] Gomez-Tejedor A, Osaba E, Villar-Rodriguez E. Addressing the minor-embedding problem in quantum annealing and evaluating state-of-the-art algorithm performance. 2025, http://dx.doi.org/10.48550/arXiv.2504.13376, arXiv:2504.13376.

[43] Pelofske E. 4-Clique network minor embedding for quantum annealers. Phys Rev Appl 2024;21:034023. http://dx.doi.org/10.1103/PhysRevApplied.21.034023, URL https://link.aps.org/doi/10.1103/PhysRevApplied.21.034023.

[44] Śmiechrzalski T, Dziubyna AM, Pawela Ł, omiej Gardas B, Rams MM. SpinGlassPEPS.jl code. 2025, https://github.com/euro-hpc-pl/SpinGlassPEPS.jl. [Accessed 30 Jan 2025].