Tropical tensor networks for spin glasses

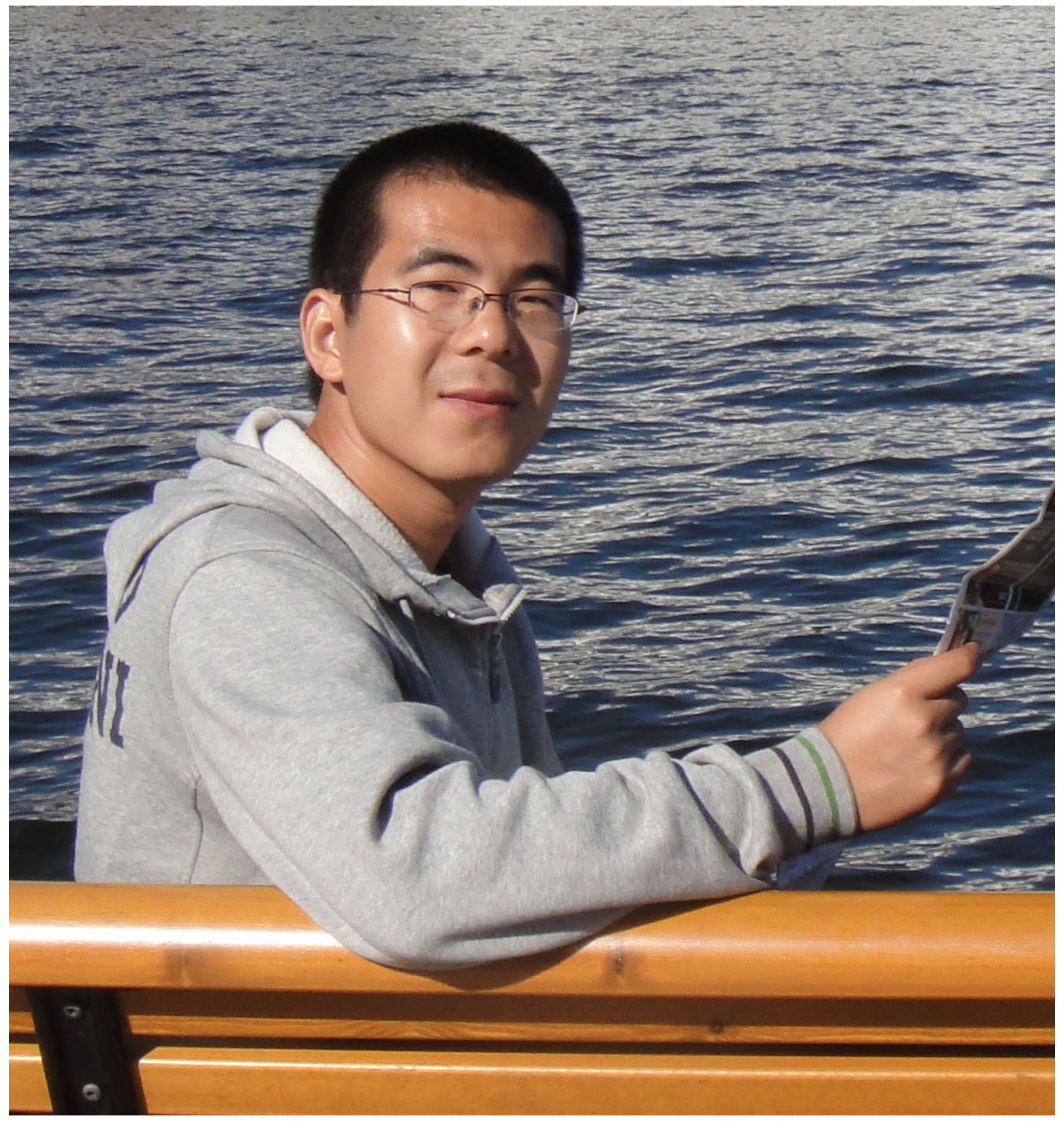
Lei Wang (王磊)
https://wangleiphy.github.io
Institute of Physics, CAS

2008.06888

https://github.com/TensorBFS/TropicalTensors.jl

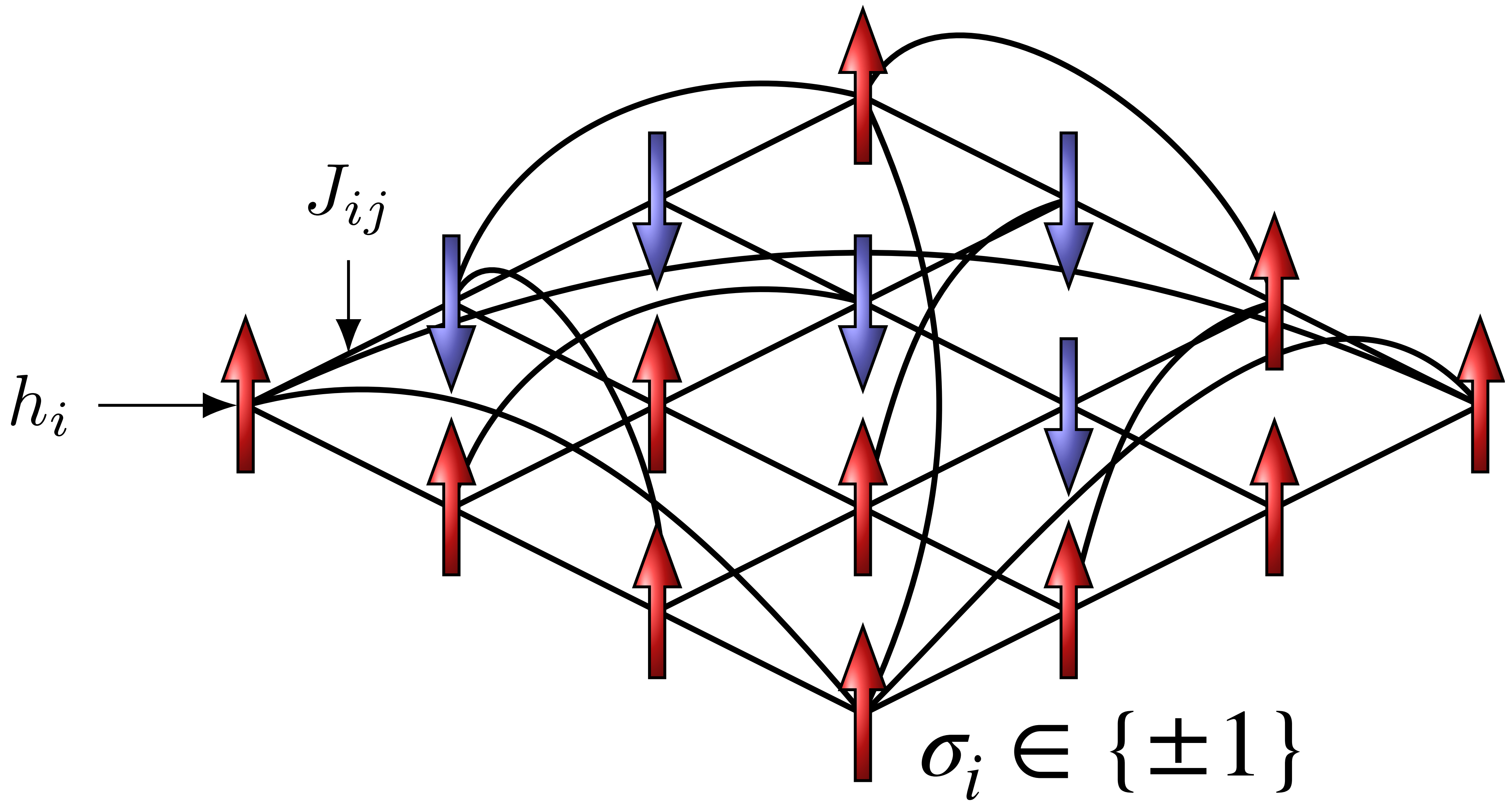
Thanks to my collaborators

刘金国, QuEra & Harvard



张潘, ITP CAS

Spin glasses

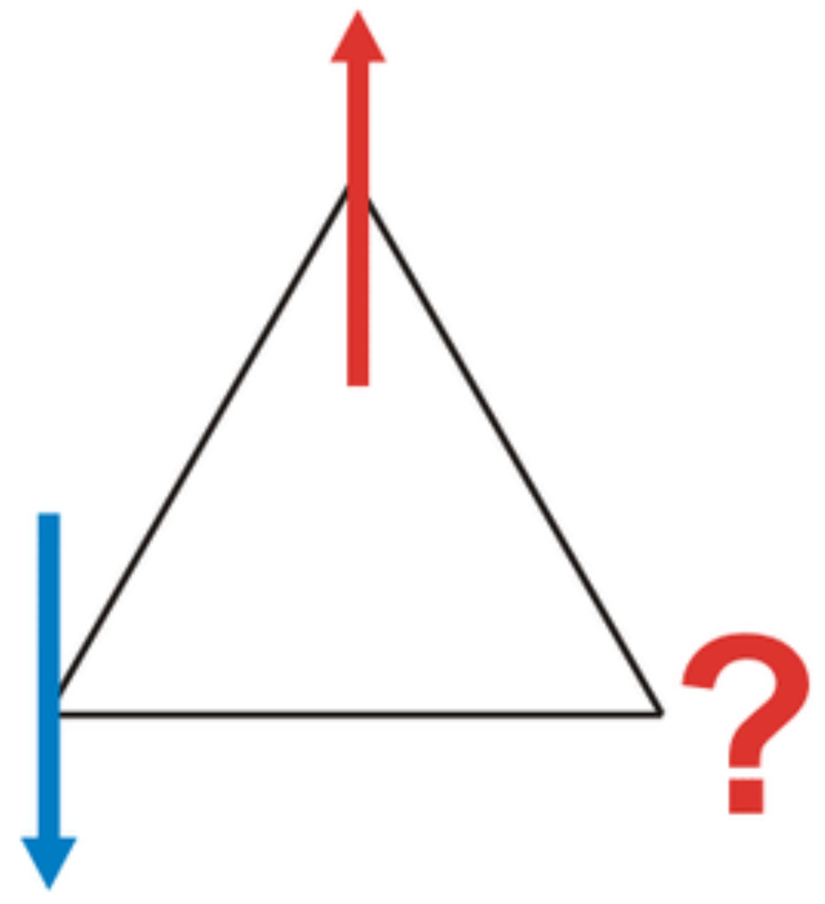


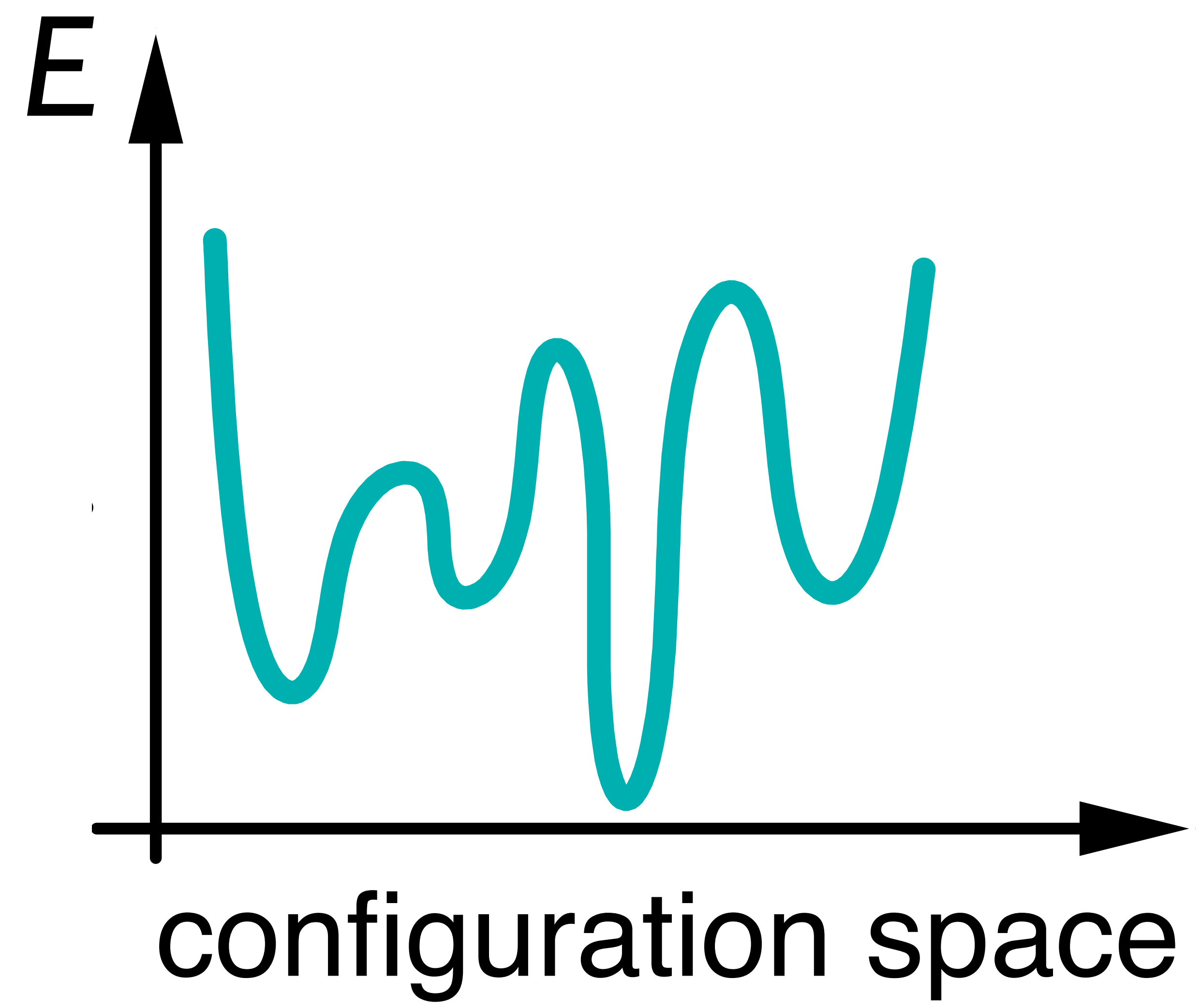
$$E(\{\sigma\}) = \sum_{i<j} J_{ij}\sigma_i\sigma_j + \sum_i h_i\sigma_i$$

$$E^* = \min_{\sigma} E(\{\sigma\}) \quad ???$$

- Same as QUBO, has broad applications in statistical physics, machine learning, coding theory…
- Finding ground state is NP-hard: Barahona 1982
- Ising formulation for many NP problems: Lucas 1302.5834
(including Karp's 21 NP-complete problems)

Difficulties of spin glasses

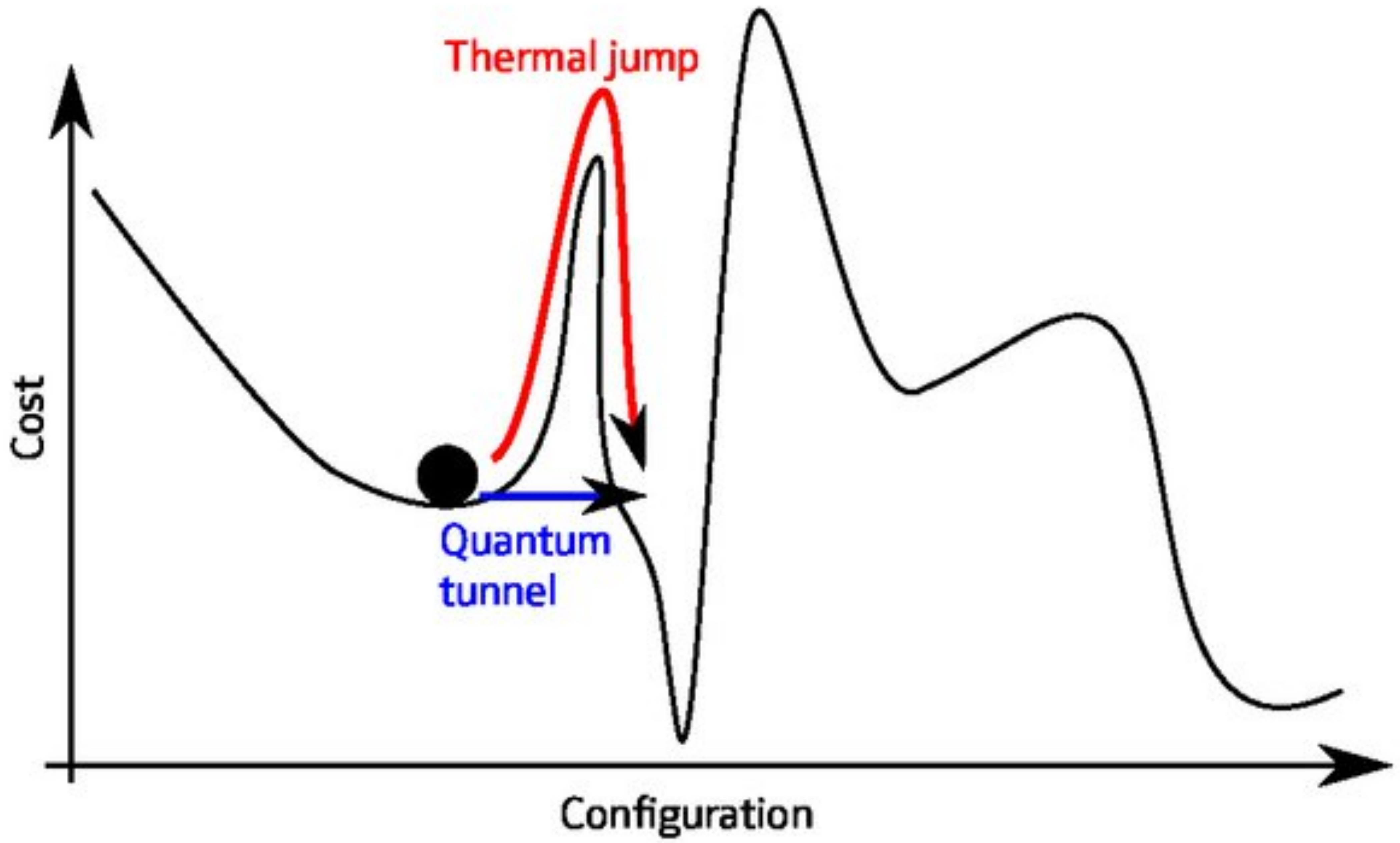




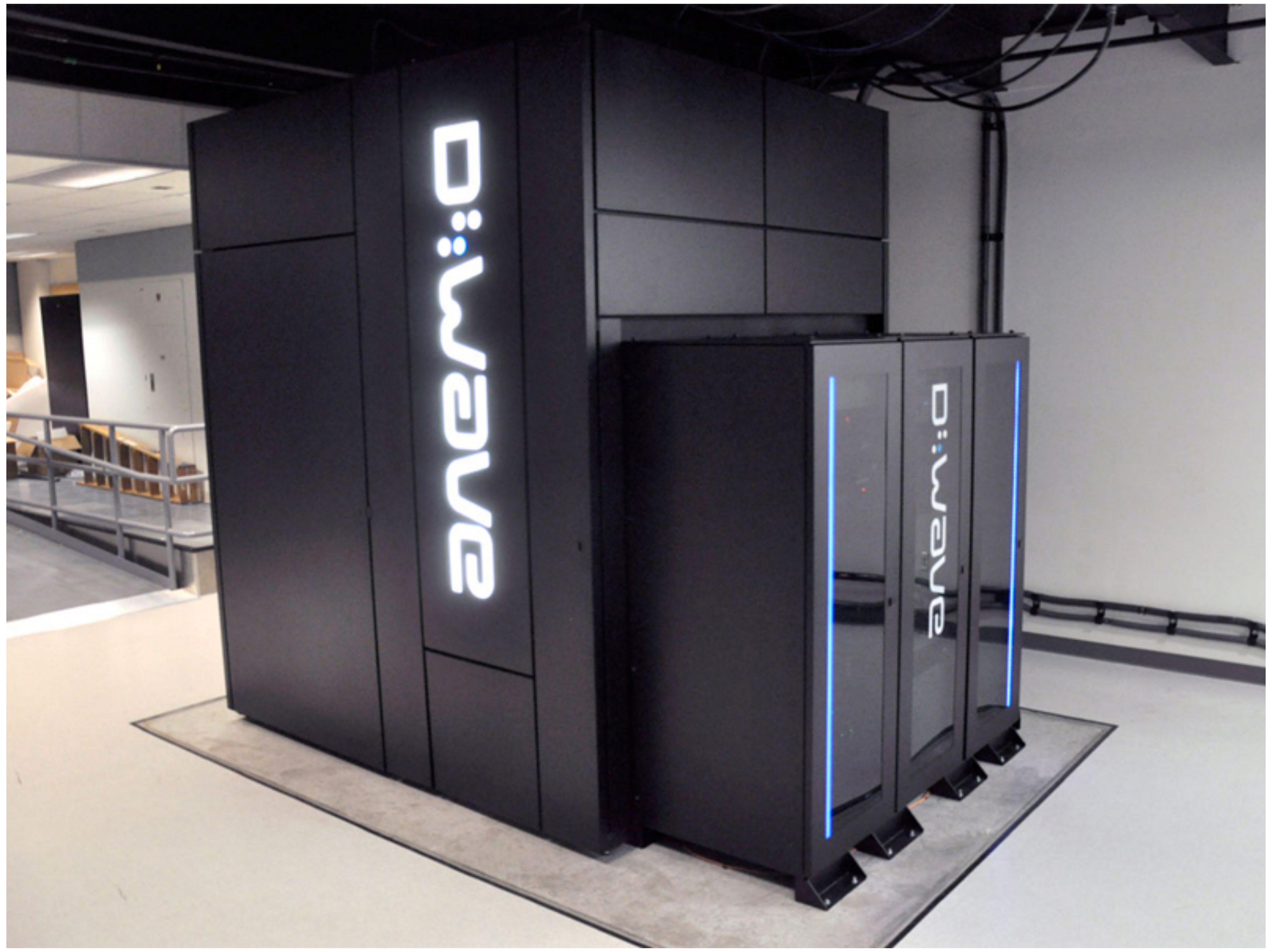
Frustration

Complex landscape

Approaches to spin glasses



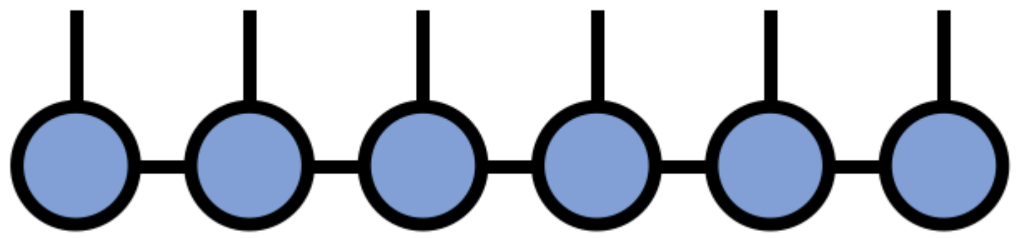
Simulated annealing: thermal fluctuation
Quantum annealing: quantum fluctuation



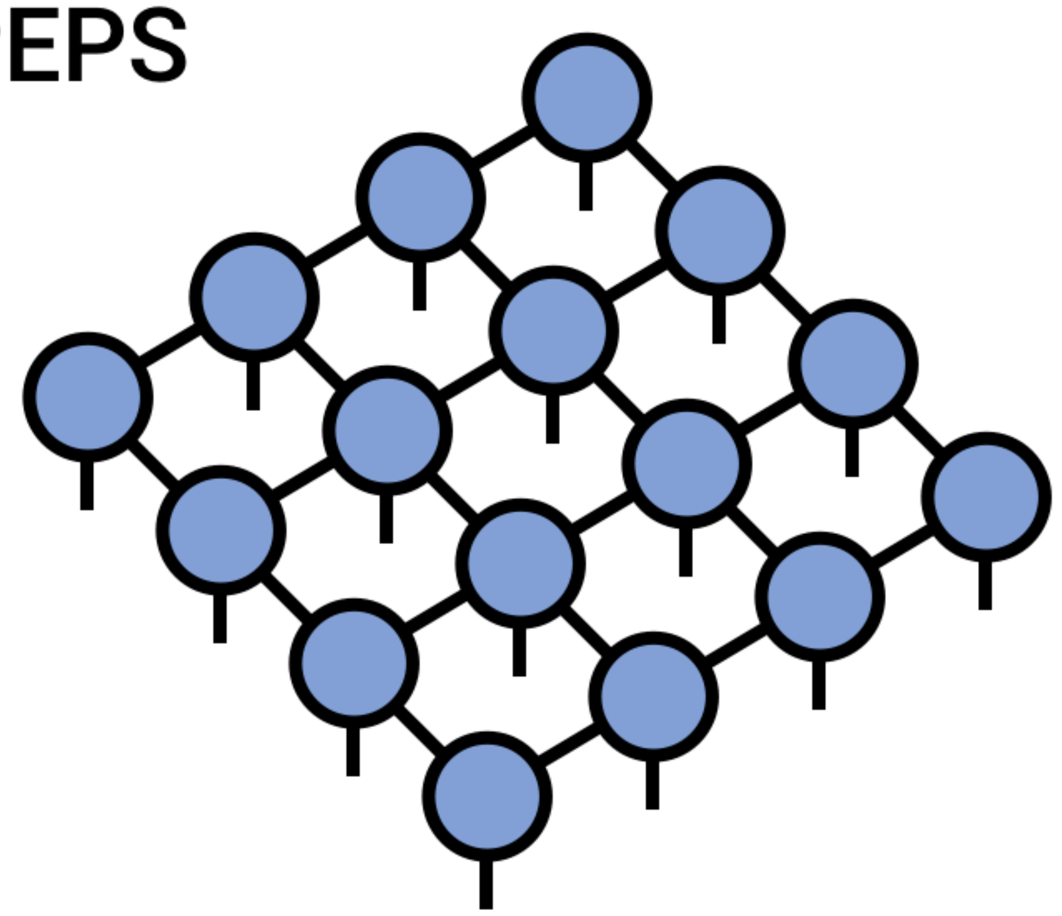
$15 million “quantum” annealer

Tensor Networks

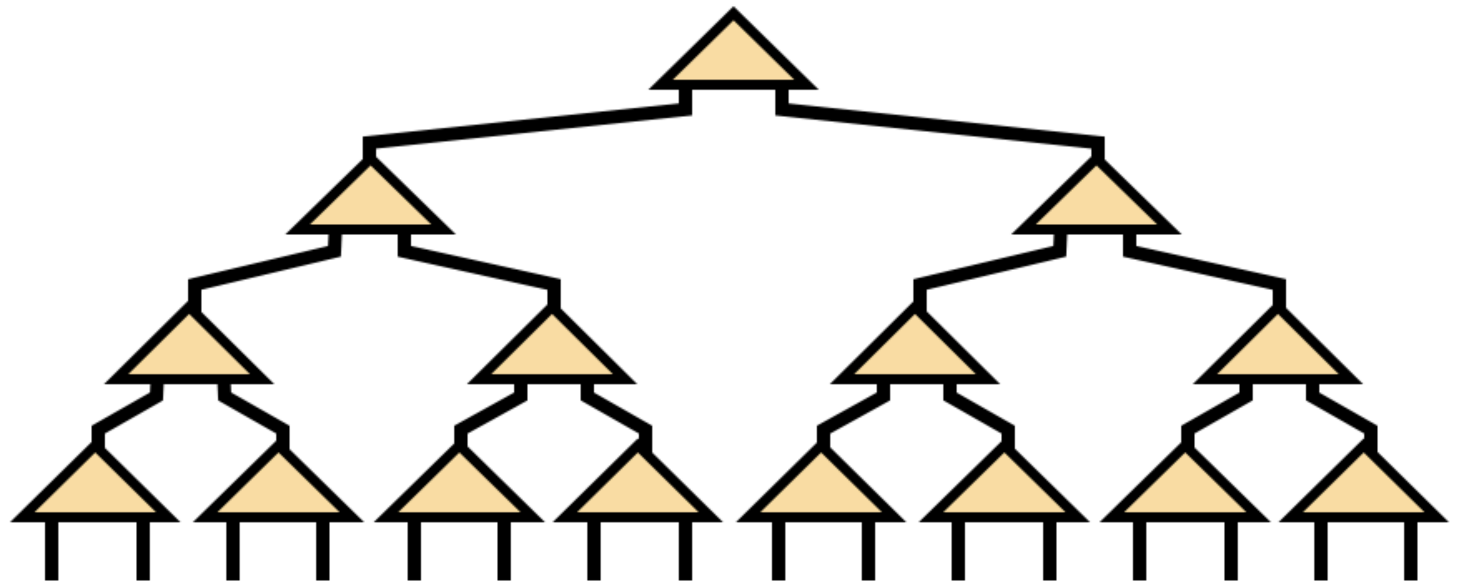
Matrix Product State / Tensor Train



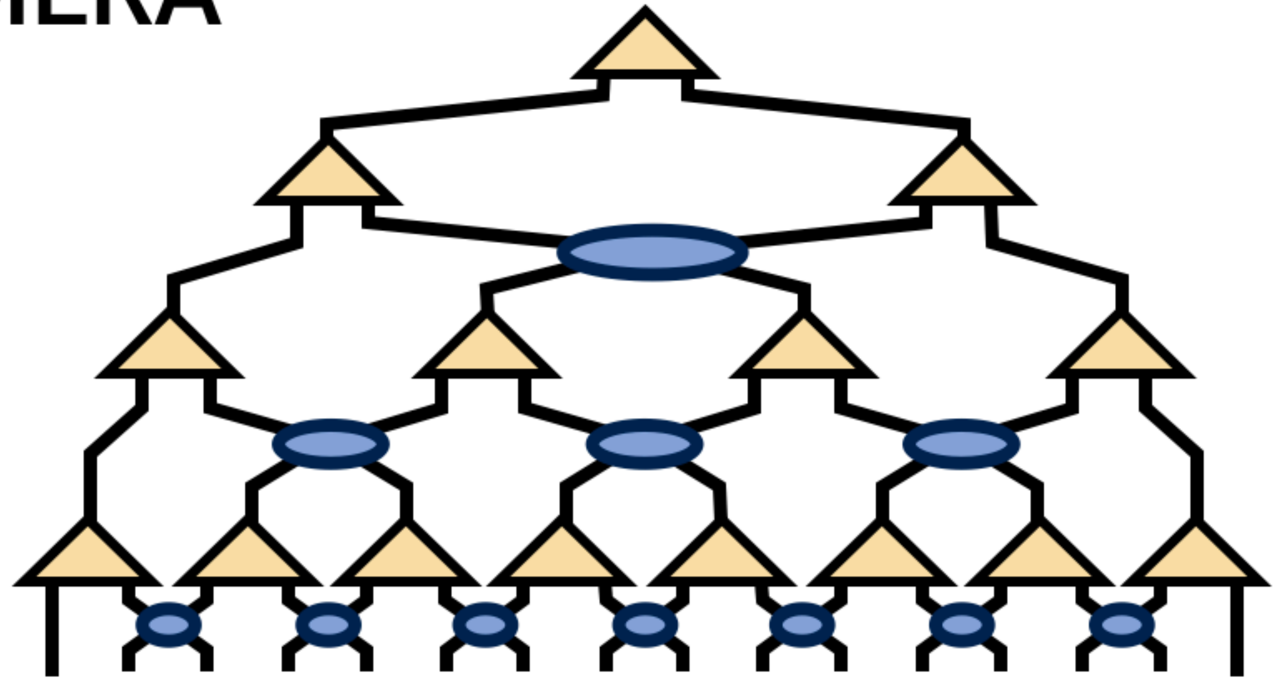
PEPS



Tree Tensor Network / Hierarchical Tucker

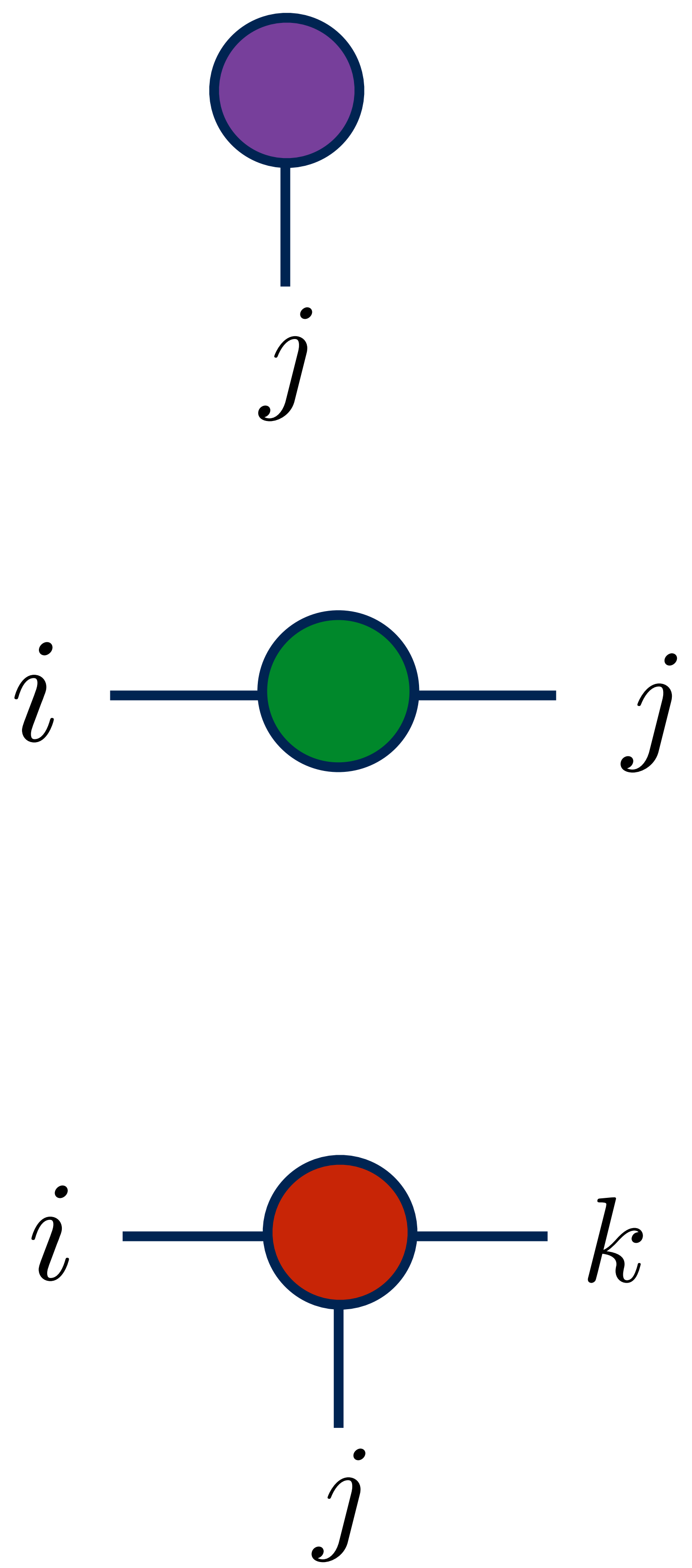


MERA



A way to efficiently express multivariable functions

Tensor Notations



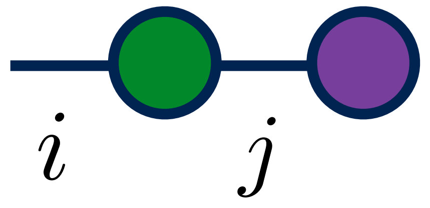
v_j

M_{ij}

T_{ijk}

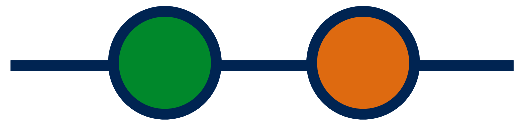
Miles Stoudenmire http://tensornetwork.org/

Tensor Contractions



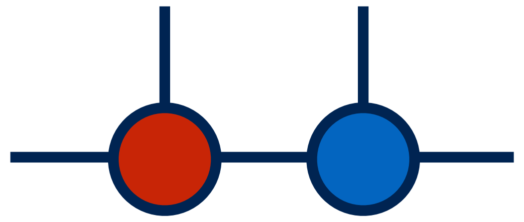
$\longleftrightarrow$

$$M_{ij}v_j$$



$\longleftrightarrow$

$$A_{ij}B_{jk}$$

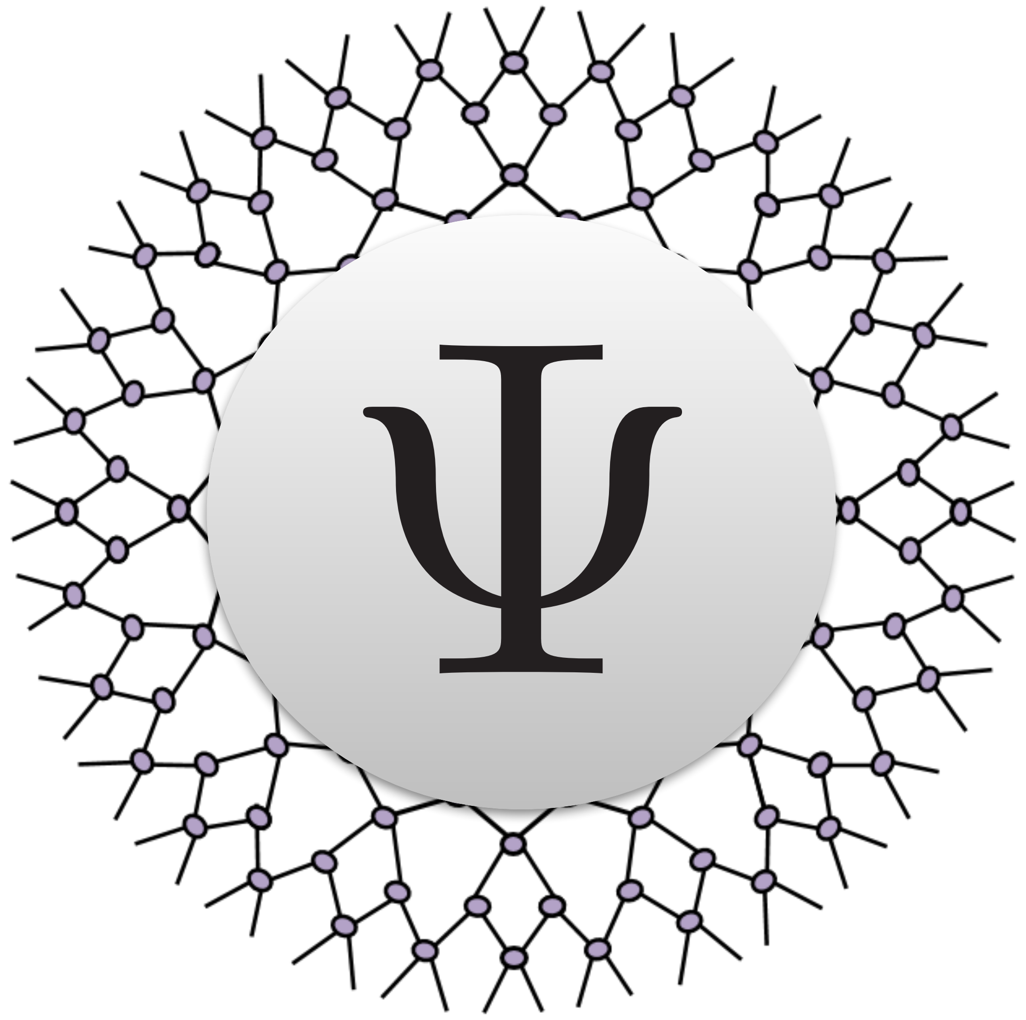


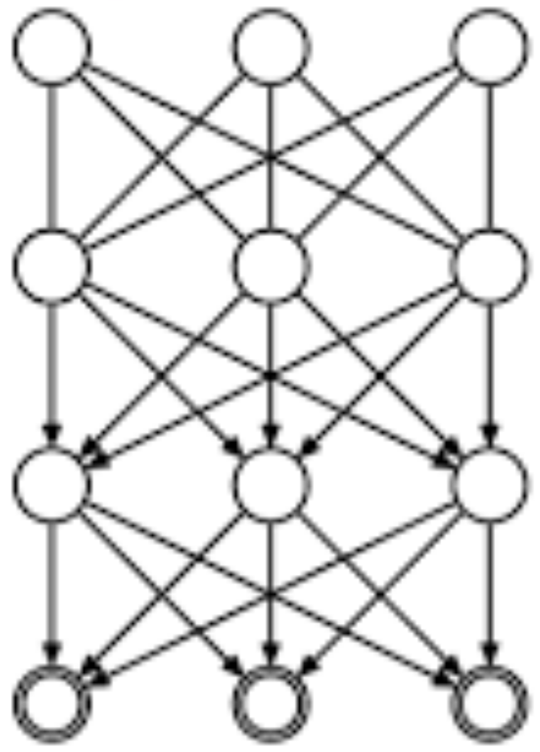
$\longleftrightarrow$

$$A_{ijk}B_{klm}$$

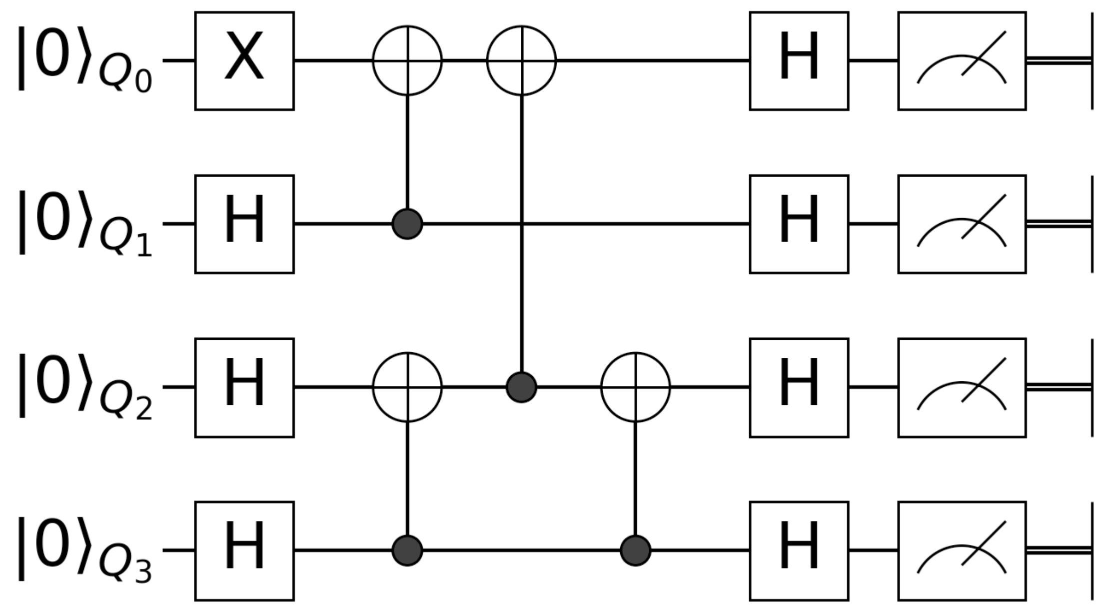
“Tensor network is 21st century’s matrix”

—Mario Szegedy





Neural networks and Probabilistic graphical models



Quantum circuit architecture, parametrization, and simulation

Tropical algebra

$$x \oplus y = \min(x, y)$$

$$x \odot y = x + y$$

Tropical algebra

Multiplication table

$\odot$	**1**	**2**	**3**	**4**	**5**	**6**	**7**
1	2	3	4	5	6	7	8
2	3	4	5	6	7	8	9
3	4	5	6	7	8	9	10
4	5	6	7	8	9	10	11
5	6	7	8	9	10	11	12
6	7	8	9	10	11	12	13
7	8	9	10	11	12	13	14

Novel Algebras for Advanced Analytics in Julia

2013 IEEE High Performance Extreme Computing Conference

Viral B. Shah*, Alan Edelman†, Stefan Karpinski‡, Jeff Bezanson§, Jeremy Kepner¶
*Email: viral@mayin.org
†Email: edelman@math.mit.edu
‡Email: stefan@karpinski.org
§Email: bezanson@mit.edu
¶Email: kepner@ll.mit.edu

***Abstract*—A linear algebraic approach to graph algorithms that exploits the sparse adjacency matrix representation of graphs can provide a variety of benefits. These benefits include syntactic simplicity, easier implementation, and higher performance. One way to employ linear algebra techniques for graph algorithms is to use a broader definition of matrix and vector multiplication. We demonstrate through the use of the Julia language system how easy it is to explore semirings using linear algebraic methodologies.**

I. Introduction

A. Semiring algebra

The duality between the canonical representation of graphs as abstract collections of vertices and edges and a sparse adjacency matrix representation has been a part of graph theory since its inception [5], [6]. Matrix algebra has been recognized as a useful tool in graph theory for nearly as long (see [3] and references therein). A linear algebraic approach to

II. Application Example

A classic example of the utility of the semiring approach is in finding the minimum path between all vertices in graph (see [11] in [4]. Given a weighted adjacency matrix for a graph where $A(i,j) = w_{ij}$ is the weight of a directed edge from vertex i to vertex j. Let $C(i,j)_2$ be the minimum 2-hop cost from vertex i to vertex j. C_2 can be computed via the semiring matrix product:

$$C_2 = A \ \min.+ A$$

Likewise, C_3 can be computed via

$$C_3 = A \ \min.+ A \min.+ A$$

and more generally

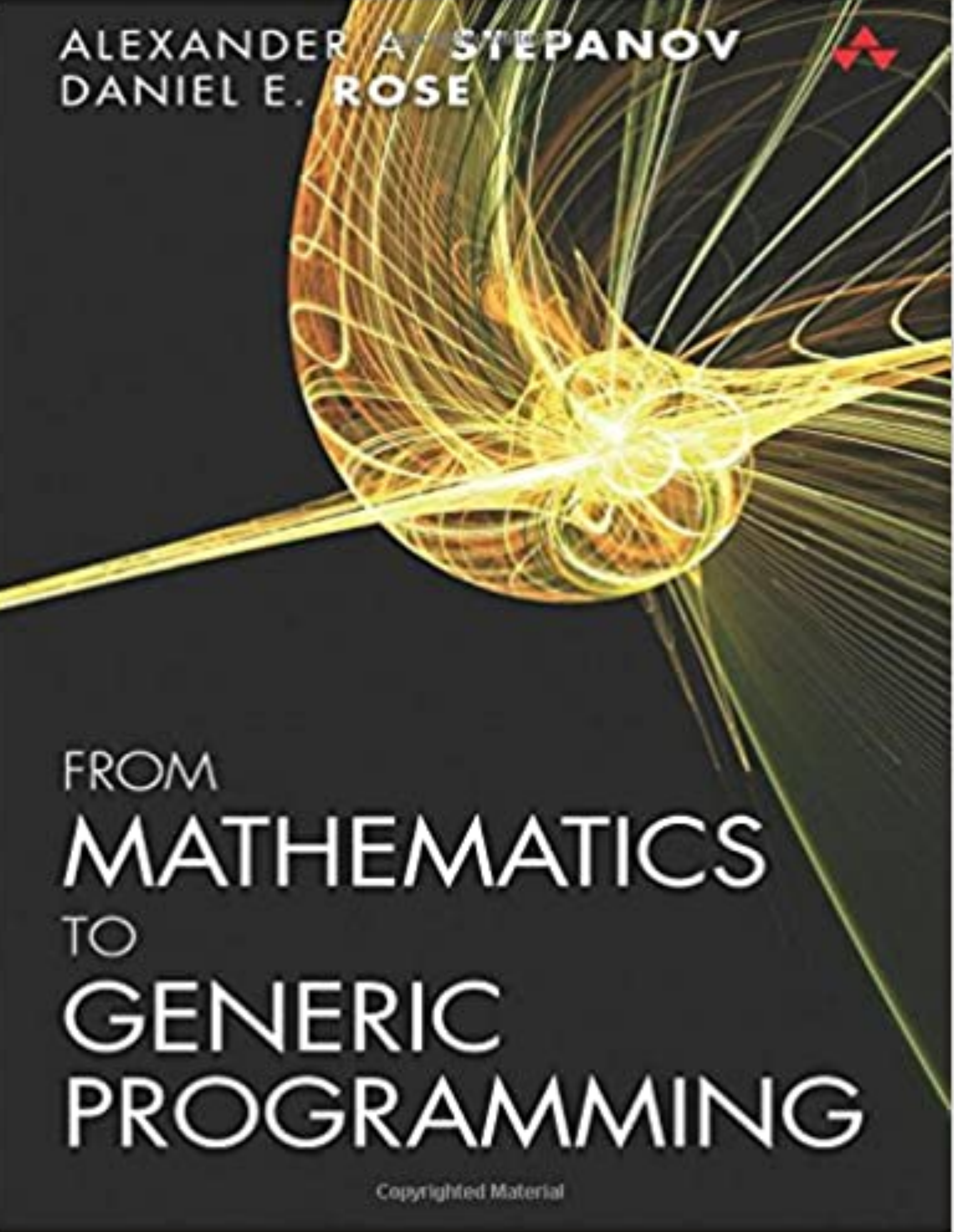
```julia
struct Tropical{T} <: Number
    x::T
end

Base.:*(a::Tropical, b::Tropical) = Tropical(a.x + b.x)
Base.:+(a::Tropical, b::Tropical) = Tropical(min(a.x, b.x))
```

Tropical tensor networks

$$T = \begin{bmatrix} \cdot & \cdot & \odot \\ \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot \end{bmatrix}$$

Tropical number

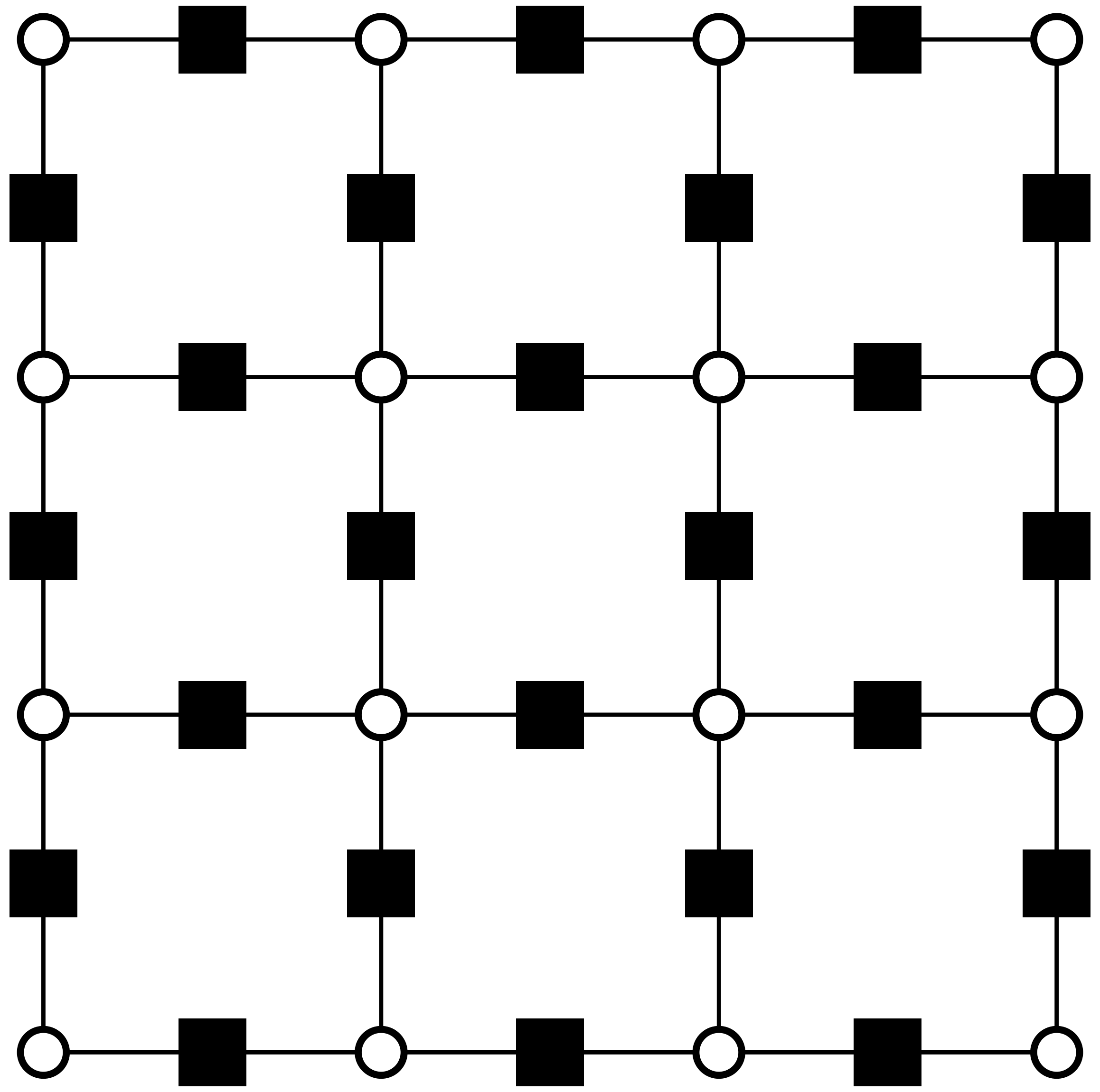
$(\oplus, \odot)$ semiring is sufficient to define tensor network contraction

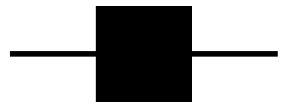


Mathematical abstraction is the key to efficient and generic programs

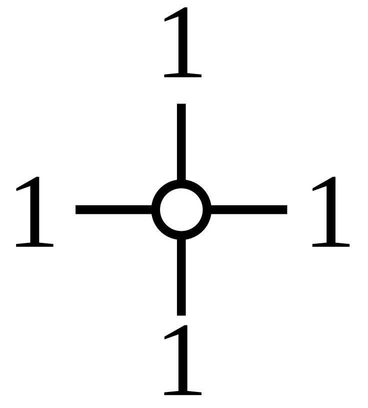
Tropical tensor networks for Ising spin glasses

$$E(\{\sigma\}) = \sum_{i<j} J_{ij}\sigma_i\sigma_j + \sum_i h_i\sigma_i$$

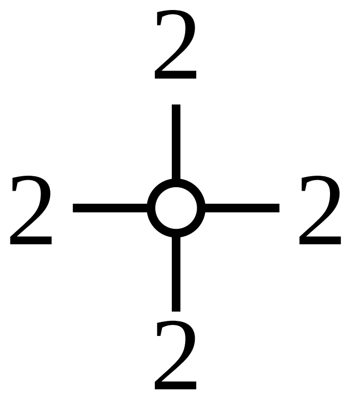




$$= \begin{pmatrix} J_{ij} & -J_{ij} \\ -J_{ij} & J_{ij} \end{pmatrix}$$



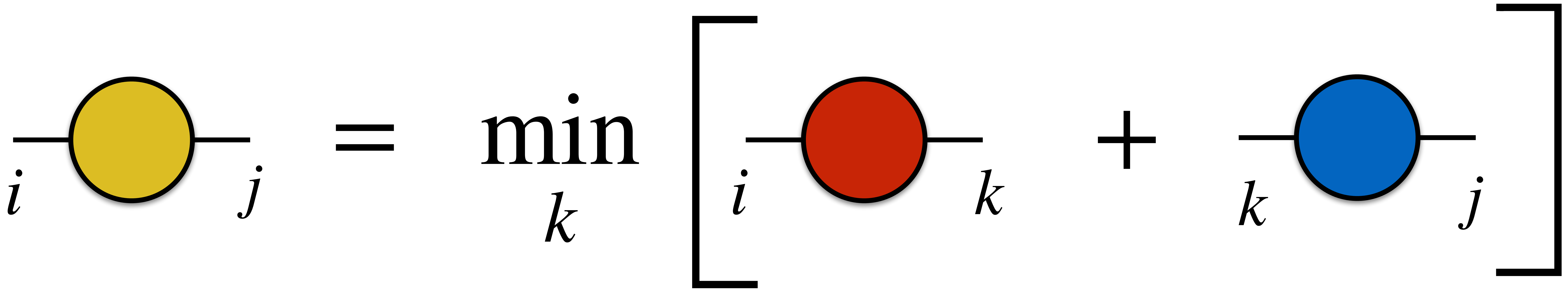
$= h_i$



$= -h_i$

all other elements are ∞

Tropical tensor network contraction
→ ground state energy



Tropical tensor stores min-marginals of the energy

*also known as algebraic dynamic programming

Physical understanding of the tropical algebra

zero temperature limit of the partition function

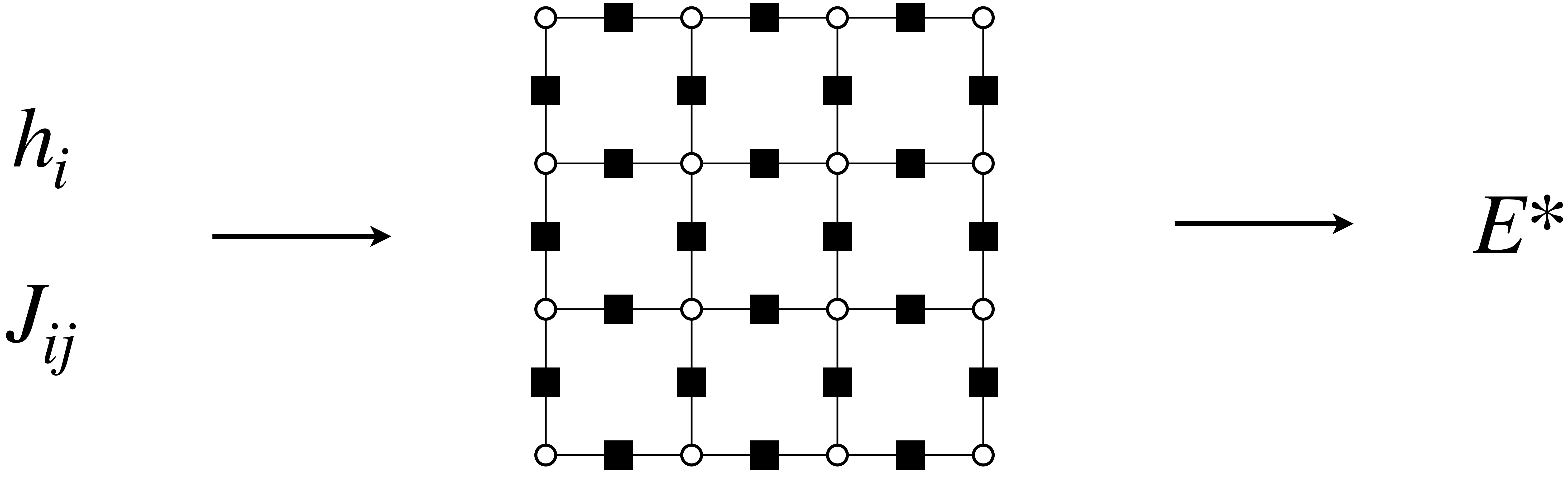
$$Z = \sum_{\{\sigma\}} e^{-\beta E} \qquad \text{consider} \qquad E^* = \lim_{\beta\to\infty} \frac{-1}{\beta} \ln Z$$

a tensor network
with ordinary algebra

$$\lim_{\beta\to\infty} \frac{-1}{\beta} \ln(e^{-\beta x} + e^{-\beta y}) = x \oplus y$$

$$\lim_{\beta\to\infty} \frac{-1}{\beta} \ln(e^{-\beta x} \cdot e^{-\beta y}) = x \odot y$$

Gradient with respect to the field
→ ground state configuration



$$\sigma_i^* = \frac{\partial E^*}{\partial h_i}$$

Optimal energy is a piecewise linear function of the field

Mix tropical with ordinary algebra
→ground state degeneracy

$$\begin{bmatrix} \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot \end{bmatrix} \qquad (x, n)$$

tropical number

ordinary number

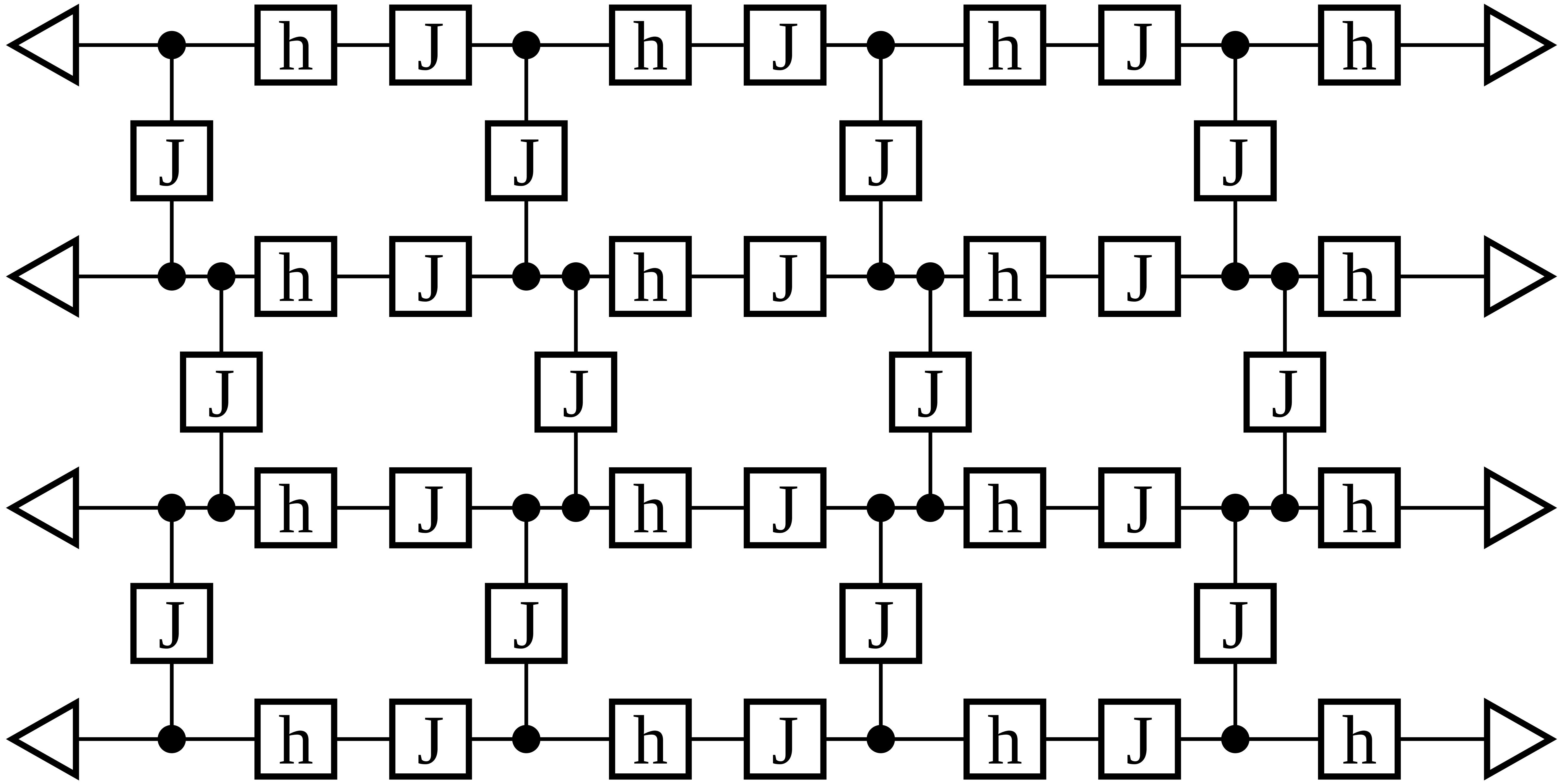
$$(x, n) \odot (y, m) = (?, ?)$$
$$(x, n) \oplus (y, m) = (?, ?)$$

The second field counts the minimal energy configuration

```julia
struct Tropical{T} <: Number
    x::T
    n::T
end

Base.:*(a::Tropical, b::Tropical) = Tropical(a.x + b.x, a.n * b.n)
function Base.:+(a::Tropical, b::Tropical)
    x = ...
    n = ...
    Tropical(x, n)
end
```

Let's contract it with Yao.jl!

https://yaoquantum.org/



Yao.jl is general enough to deal with non-unitary gates with such crazy algebra

https://gist.github.com/wangleiphy/ef1f616f26ab37ef7fd3d329f2a5be0e

```
julia> res = square_solve(L, L, Js, hs)
┌ Warning: Input type of `ArrayReg` is not Complex, got Tropical{Float64}
└ @ YaoArrayRegister ~/.julia/packages/YaoArrayRegister/9wa6b/src/register.jl:92
Layer 1/10
Layer 2/10
Layer 3/10
Layer 4/10
Layer 5/10
Layer 6/10
Layer 7/10
Layer 8/10
Layer 9/10
Layer 10/10
Tropical{Int64}(138, 96)
```

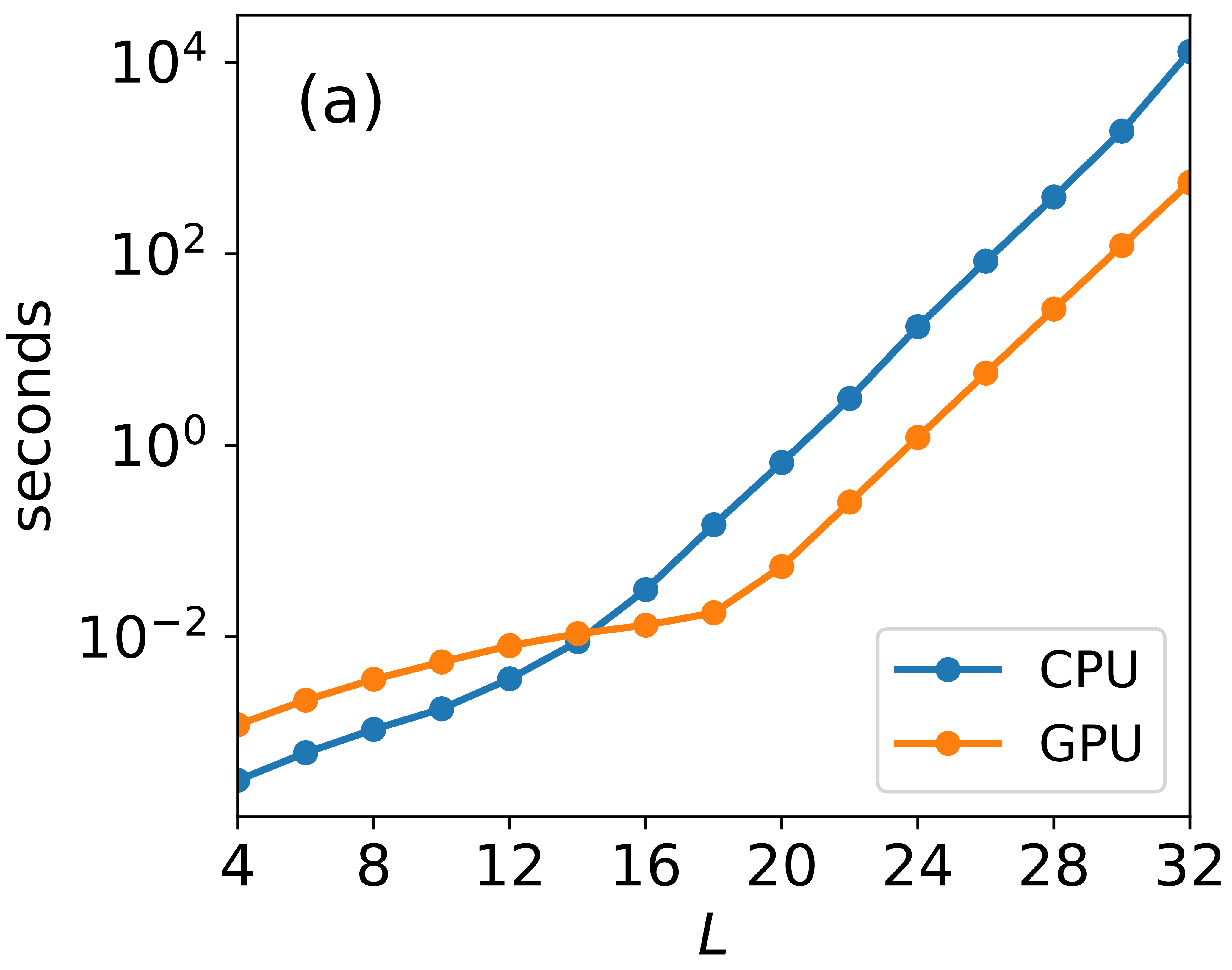
Nevertheless, it works!

One can repurpose quantum circuit simulators for combinatorial optimization problems

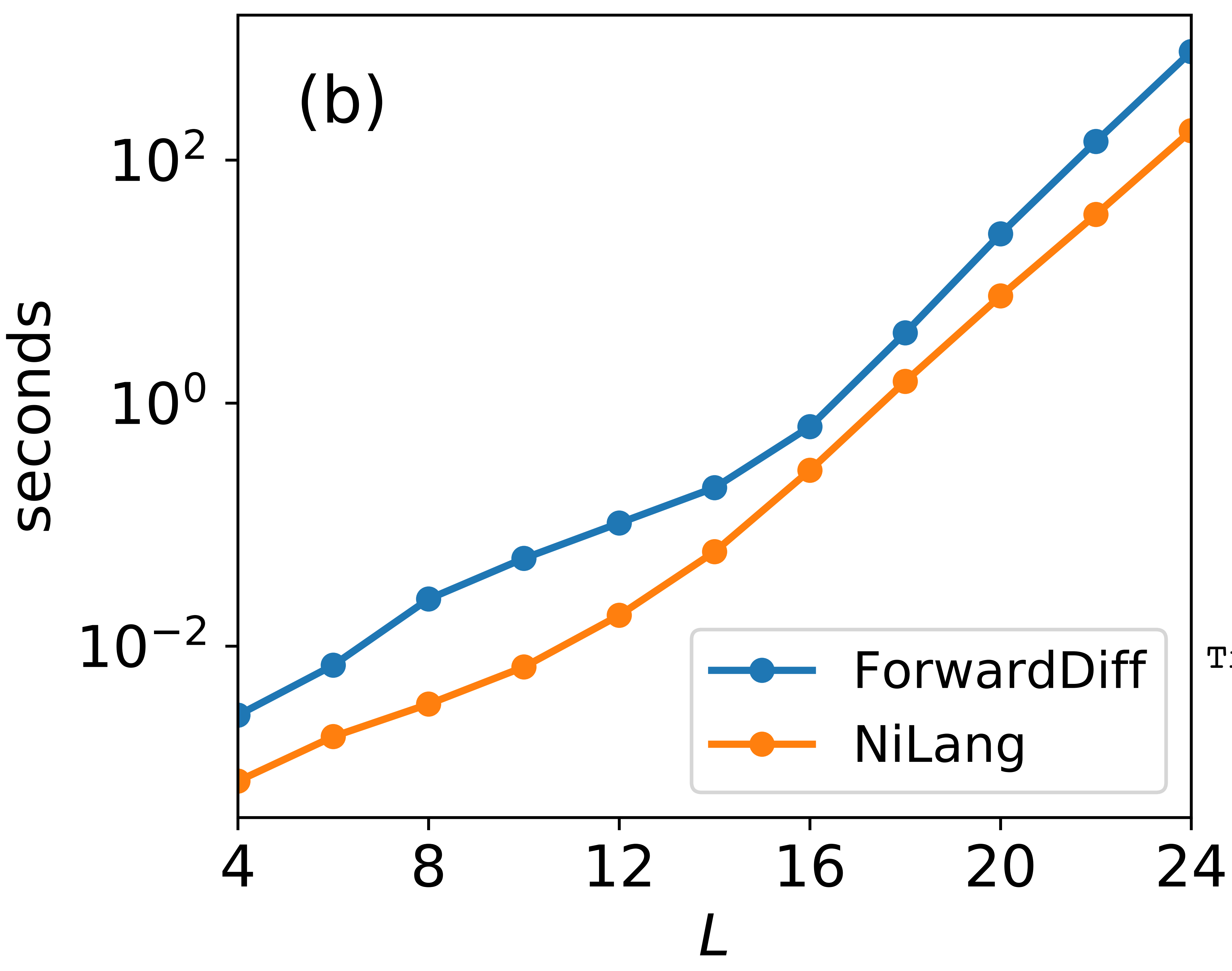
BLAS → “Tropical BLAS”

Square lattice spin glasses

Time for ground state energy



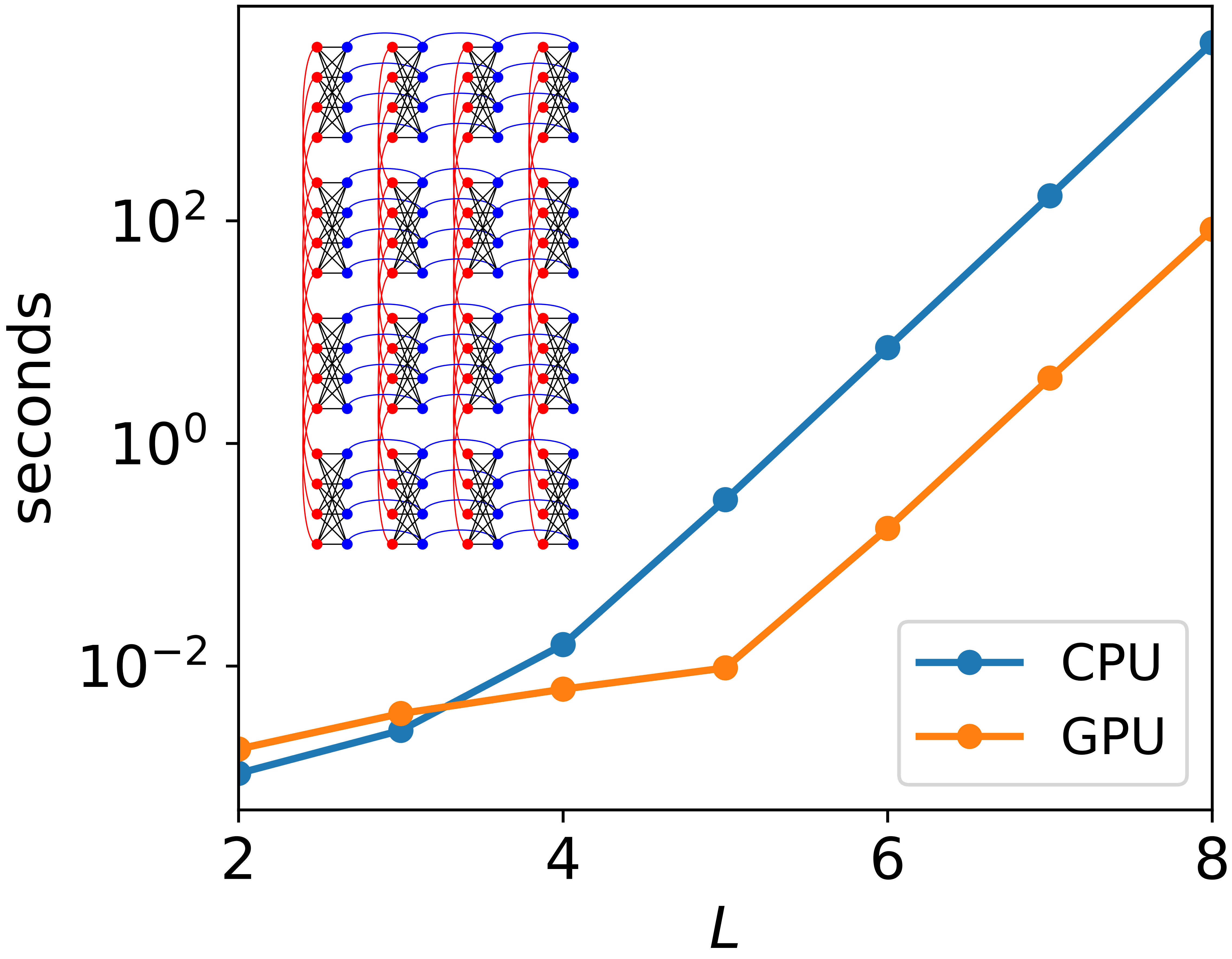
Time for ground state configuration



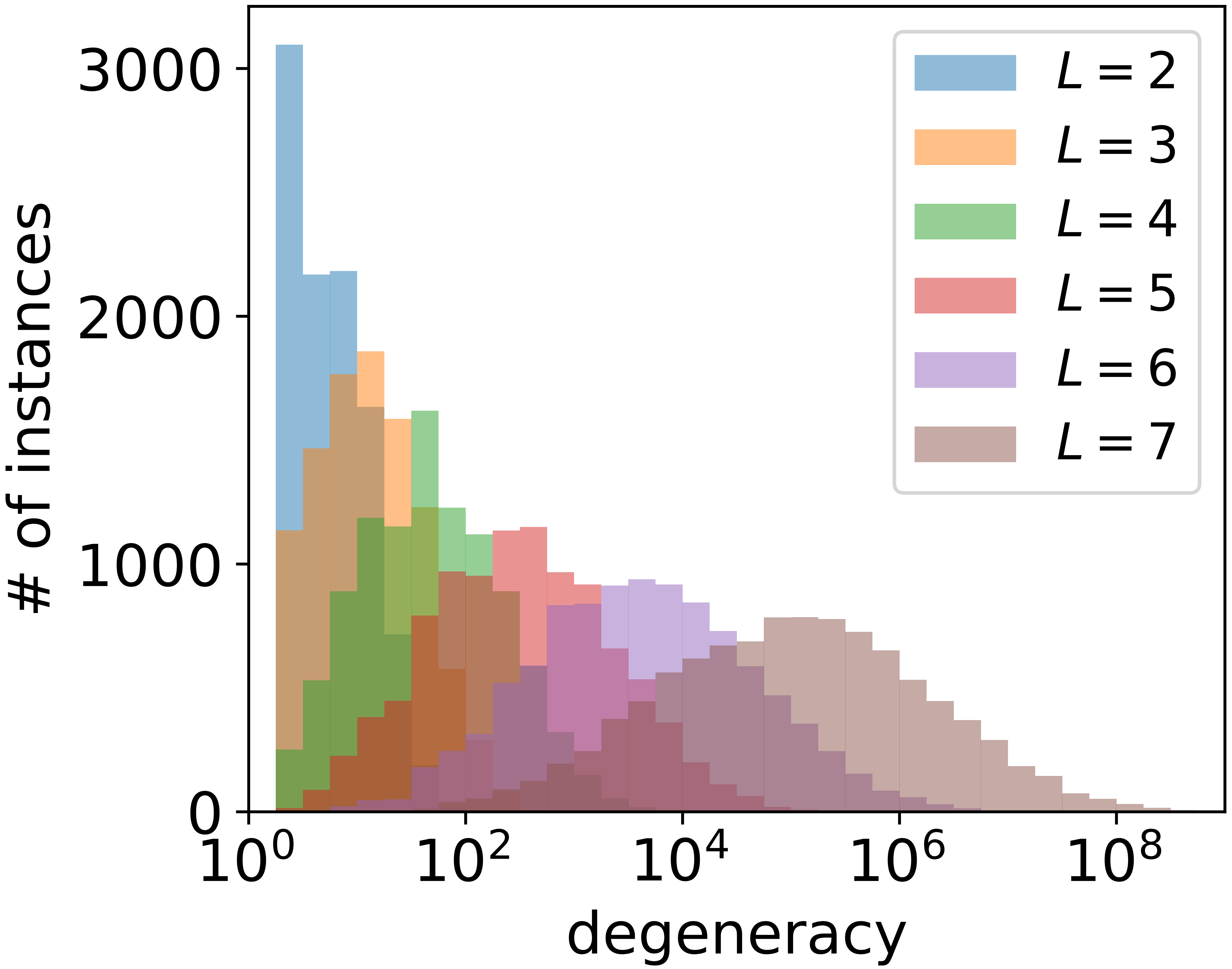
`Tropical{Dual{T}}`

Chimera graph Ising spin glass

Time for ground state energy



Histogram of ground state degeneracy



Summary

Repurpose tensor contractors and quantum simulators for optimization

Differentiable programming as a unified approach to optimal solution

Count # of optimal solutions: application ?

WANTED: Approximated contraction under tropical algebra

谢谢！