(Sub)Exponential advantage of adiabatic quantum computation with no sign problem

Matthew Hastings
Microsoft

András Gilyén
Caltech

Umesh Vazirani
UC Berkeley

arXiv:2005.03791

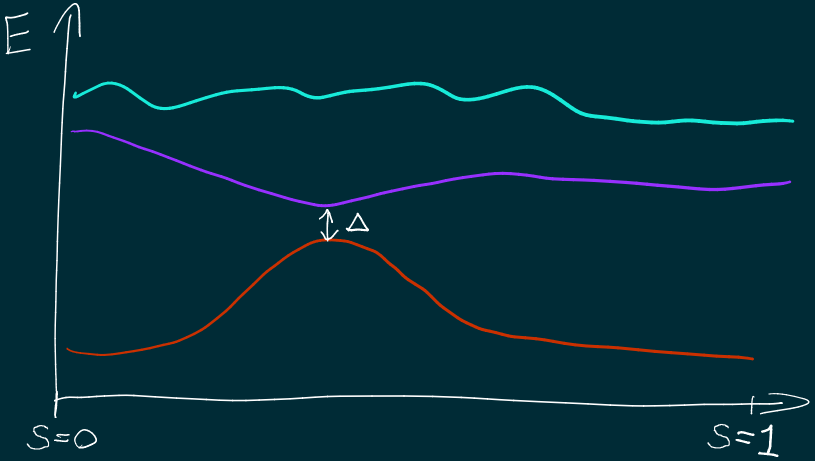
arXiv:2011.09495

Adiabatic quantum computation – background

- A model of quantum computation utilizing a slowly changing Hamiltonian $H(s)$: The adiabatic evolution stays in the ground state $\psi(s)$ and maps $\psi(0)$ to $\psi(1)$.

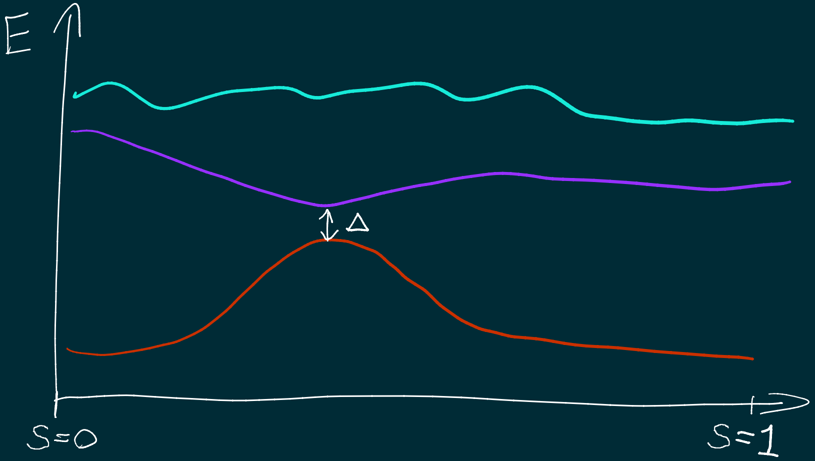
Adiabatic quantum computation – background

- A model of quantum computation utilizing a slowly changing Hamiltonian $H(s)$: The adiabatic evolution stays in the ground state $\psi(s)$ and maps $\psi(0)$ to $\psi(1)$.



Adiabatic quantum computation – background

- A model of quantum computation utilizing a slowly changing Hamiltonian $H(s)$: The adiabatic evolution stays in the ground state $\psi(s)$ and maps $\psi(0)$ to $\psi(1)$.



- The ground state is accurately mapped if the evolution time is $t = \Theta(1/\mathrm{poly}(\Delta))$

Adiabatic quantum computation

- Is an equivalent universal model of quantum computation [Aharonov et al. 2004]

Adiabatic quantum computation

- Is an equivalent universal model of quantum computation [Aharonov et al. 2004]

- A popular heuristics for combinatorial optimization

$$H(s) = (s-1)\sum_{i=1}^{n} X_i + s \cdot H_z,$$

  where H_z is the diagonal Hamiltonian describing the classical objective function

Adiabatic quantum computation

- Is an equivalent universal model of quantum computation [Aharonov et al. 2004]

- A popular heuristics for combinatorial optimization

$$H(s) = (s - 1)\sum_{i=1}^{n} X_i + s \cdot H_z,$$

  where H_z is the diagonal Hamiltonian describing the classical objective function

- Unfortunately, for hard problems the gap Δ tends to become small

Adiabatic quantum computation

- Is an equivalent universal model of quantum computation [Aharonov et al. 2004]

- A popular heuristics for combinatorial optimization

$$H(s) = (s-1)\sum_{i=1}^{n} X_i + s \cdot H_z,$$

  where H_z is the diagonal Hamiltonian describing the classical objective function

- Unfortunately, for hard problems the gap Δ tends to become small

- An interesting special case is when the Hamiltonians have “no sign problem”, i.e., all off-diagonal matrix elements of $H(s)$ are non-positive for every $s \in [0,1]$ (note that this is a basis-dependent property)

No sign problem (stoquastic) – classically tractable?

- The lack of signs prevents destructive interference in the ground state
  Perron-Frobenius theorem: all amplitudes in the ground state have the same sign

No sign problem (stoquastic) – classically tractable?

- The lack of signs prevents destructive interference in the ground state
  Perron-Frobenius theorem: all amplitudes in the ground state have the same sign

- If the Hamiltonians are also frustration-free, then the adiabatic evolution can be classically efficiently simulated [Bravyi & Terhal, 2008]

No sign problem (stoquastic) – classically tractable?

- The lack of signs prevents destructive interference in the ground state
  Perron-Frobenius theorem: all amplitudes in the ground state have the same sign

- If the Hamiltonians are also frustration-free, then the adiabatic evolution can be classically efficiently simulated [Bravyi & Terhal, 2008]

- Monte Carlo methods tend to work remarkably well in practice

No sign problem (stoquastic) – classically tractable?

- The lack of signs prevents destructive interference in the ground state
  Perron-Frobenius theorem: all amplitudes in the ground state have the same sign
- If the Hamiltonians are also frustration-free, then the adiabatic evolution can be classically efficiently simulated [Bravyi & Terhal, 2008]
- Monte Carlo methods tend to work remarkably well in practice
- However, there are topological obstruction to path integral Monte Carlo [Hastings & Freedman, 2013]

No sign problem (stoquastic) – classically tractable?

- The lack of signs prevents destructive interference in the ground state
  Perron-Frobenius theorem: all amplitudes in the ground state have the same sign
- If the Hamiltonians are also frustration-free, then the adiabatic evolution can be classically efficiently simulated [Bravyi & Terhal, 2008]
- Monte Carlo methods tend to work remarkably well in practice
- However, there are topological obstruction to path integral Monte Carlo [Hastings & Freedman, 2013]
- Diffusion Monte Carlo is sensitive to ℓ_1 vs. ℓ_2 differences in the ground state [Jarret, Jordan, & Lackey, 2016]

Topological obstructions

- Plain Monte Carlo algorithms perform a random walk on the state space

Topological obstructions

- Plain Monte Carlo algorithms perform a random walk on the state space
- Path integral Monte Carlo is a random walk among paths on the state space

Topological obstructions

- Plain Monte Carlo algorithms perform a random walk on the state space
- Path integral Monte Carlo is a random walk among paths on the state space

Obstructions to path integral Monte Carlo

- Local moves gradually grow / shrink the paths between the moves

Topological obstructions

- Plain Monte Carlo algorithms perform a random walk on the state space
- Path integral Monte Carlo is a random walk among paths on the state space

Obstructions to path integral Monte Carlo

- Local moves gradually grow / shrink the paths between the moves
- Long cycles that cannot be approximated by shorter cycles are problematic (especially if the long cycles make important contribution, e.g., particle on a torus)

Topological obstructions

- Plain Monte Carlo algorithms perform a random walk on the state space
- Path integral Monte Carlo is a random walk among paths on the state space

Obstructions to path integral Monte Carlo

- Local moves gradually grow / shrink the paths between the moves
- Long cycles that cannot be approximated by shorter cycles are problematic (especially if the long cycles make important contribution, e.g., particle on a torus)
- In practice this issue can be often fixed by adding appropriate non-local moves

Exponential quantum advantage with no sign problem

Main result (informal)

There is a family of sparse sign-problem-free Hamiltonians on n qubits with a straight adiabatic path featuring a spectral gap $\Delta = \Omega(1/\mathrm{poly}(n))$, whose evolution requires $2^{\sqrt[5]{n}}$ queries to the Hamiltonian entries to simulate classically.

Exponential quantum advantage with no sign problem

Main result (informal)

There is a family of sparse sign-problem-free Hamiltonians on n qubits with a straight adiabatic path featuring a spectral gap $\Delta = \Omega(1/\mathrm{poly}(n))$, whose evolution requires $2^{\sqrt[5]{n}}$ queries to the Hamiltonian entries to simulate classically.

- This is an oracle separation

Exponential quantum advantage with no sign problem

Main result (informal)

There is a family of sparse sign-problem-free Hamiltonians on n qubits with a straight adiabatic path featuring a spectral gap $\Delta = \Omega(1/\mathrm{poly}(n))$, whose evolution requires $2^{\sqrt[5]{n}}$ queries to the Hamiltonian entries to simulate classically.

- This is an oracle separation

- If we want a provable advantage we need to rely on oracles

Exponential quantum advantage with no sign problem

Main result (informal)

There is a family of sparse sign-problem-free Hamiltonians on n qubits with a straight adiabatic path featuring a spectral gap $\Delta = \Omega(1/\mathrm{poly}(n))$, whose evolution requires $2^{\sqrt[5]{n}}$ queries to the Hamiltonian entries to simulate classically.

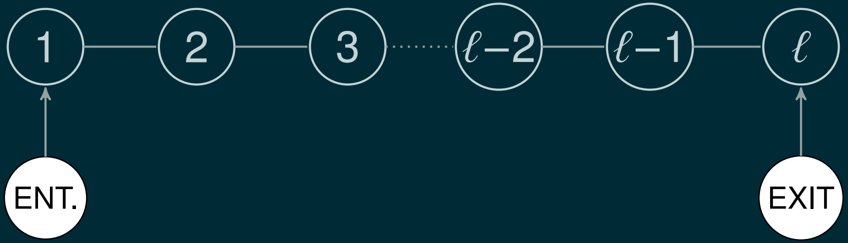
- This is an oracle separation
- If we want a provable advantage we need to rely on oracles

The corresponding graph problem (informal)

Given (via black-box access) a sparse graph on 2^n vertices and a specific "ENTRANCE" vertex, we can find the "EXIT" vertex in $\mathrm{poly}(n)$ time via adiabatic evolution of a corresponding Hamiltonian which has $1/\mathrm{poly}(n)$ spectral gap and no sign problem. At the same time any classical randomized algorithm must make at least $2^{\sqrt[5]{n}}$ queries to the graph (i.e., to the black-box) for finding the "EXIT".

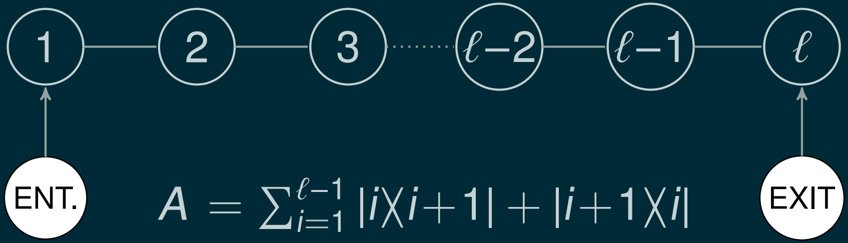
The underlying adiabatic path ($\ell = \mathrm{poly}(n)$)

$$H_\ell = -A$$



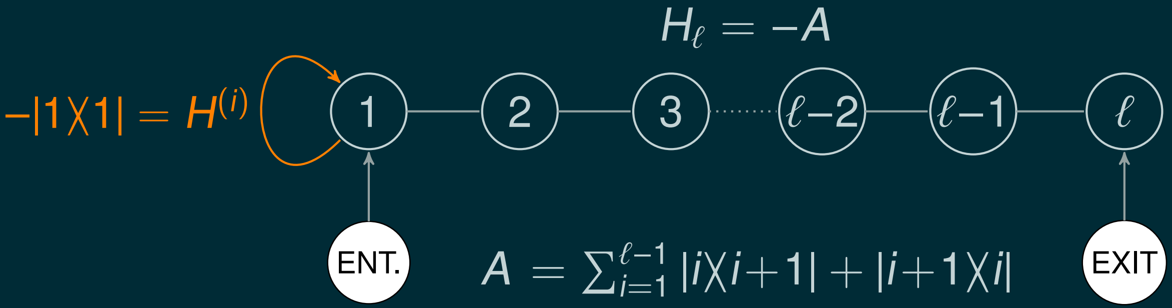
The underlying adiabatic path ($\ell = \mathrm{poly}(n)$)

$$H_\ell = -A$$

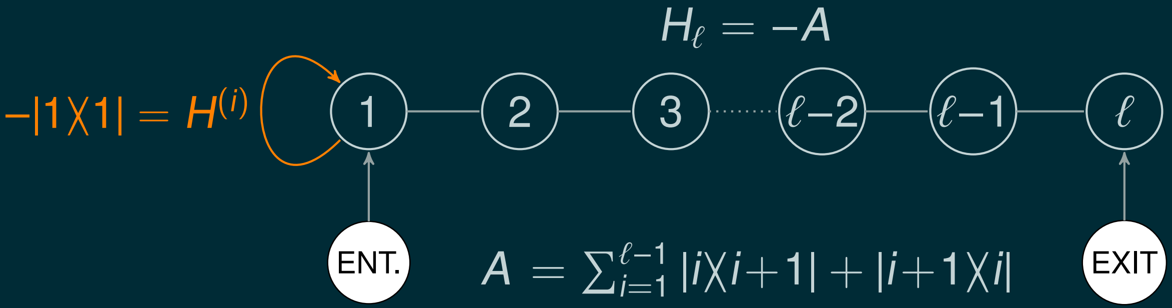


$$A = \sum_{i=1}^{\ell-1} |i\rangle\langle i+1| + |i+1\rangle\langle i|$$

The underlying adiabatic path ($\ell = \mathrm{poly}(n)$)



The underlying adiabatic path ($\ell = \text{poly}(n)$)

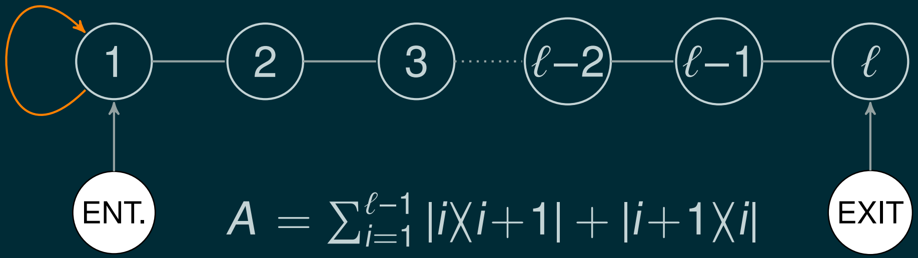


We use a simple adiabatic path $H_\ell(s)$ interpolating between $H^{(i)}$ and H_ℓ:

The underlying adiabatic path ($\ell = \mathrm{poly}(n)$)

$$H_\ell = -A$$

$$-|1\rangle\langle 1| = H^{(i)}$$

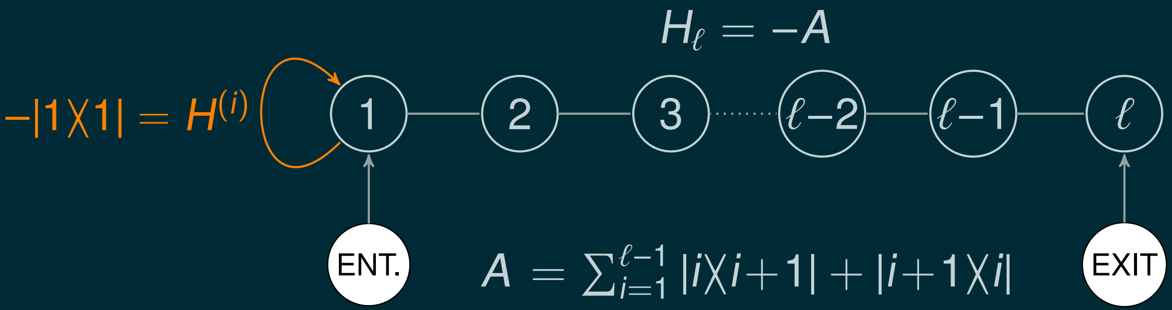


$$A = \sum_{i=1}^{\ell-1} |i\rangle\langle i+1| + |i+1\rangle\langle i|$$

We use a simple adiabatic path $H_\ell(s)$ interpolating between $H^{(i)}$ and H_ℓ:

$$H_\ell(s) = (1-s)H^{(i)} + sH_\ell$$

The underlying adiabatic path ($\ell = \mathrm{poly}(n)$)



We use a simple adiabatic path $H_\ell(s)$ interpolating between $H^{(i)}$ and H_ℓ:

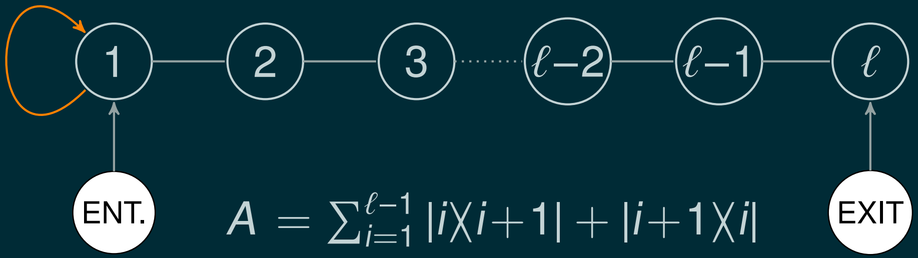
$$H_\ell(s) = (1-s)H^{(i)} + sH_\ell$$

$$\Delta = \Theta(1/\ell^2) = \Theta(1/\mathrm{poly}(n))$$

The underlying adiabatic path ($\ell = \mathrm{poly}(n)$)

$$H_\ell = -A$$

$$-|1\rangle\langle 1| = H^{(i)}$$

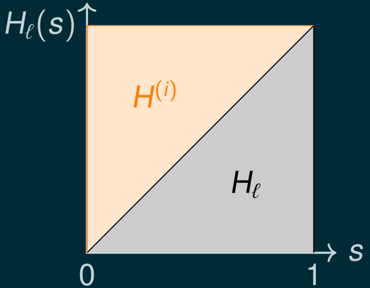


$$A = \sum_{i=1}^{\ell-1} |i\rangle\langle i+1| + |i+1\rangle\langle i|$$

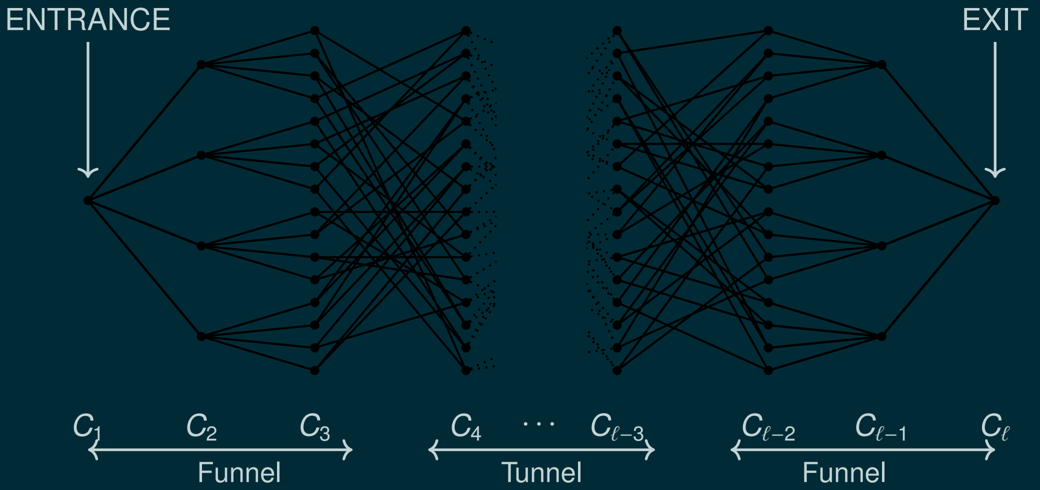
We use a simple adiabatic path $H_\ell(s)$ interpolating between $H^{(i)}$ and H_ℓ:

$$H_\ell(s) = (1-s)H^{(i)} + sH_\ell$$

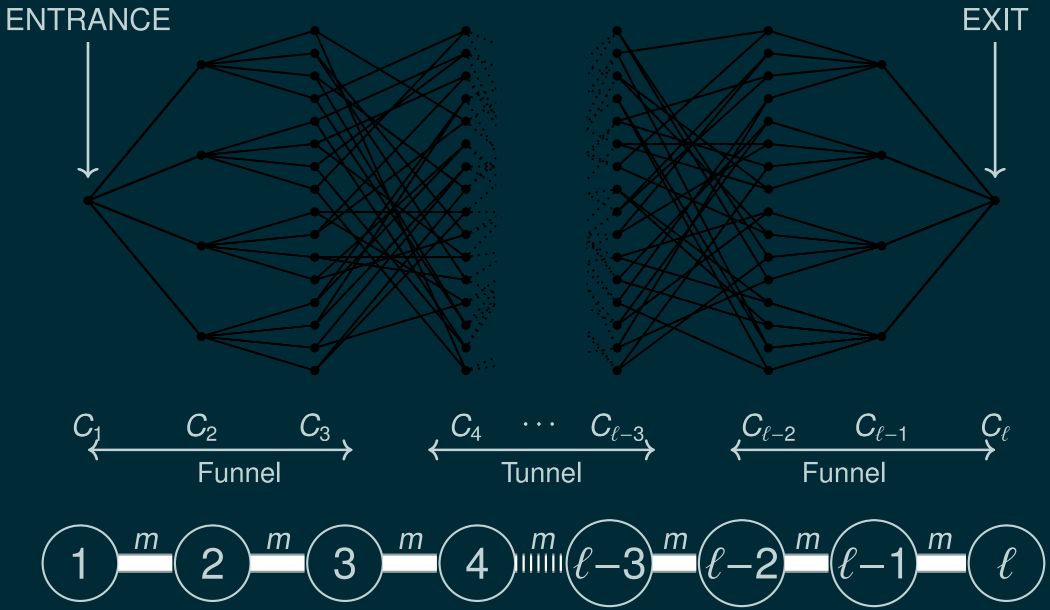
$$\Delta = \Theta(1/\ell^2) = \Theta(1/\mathrm{poly}(n))$$



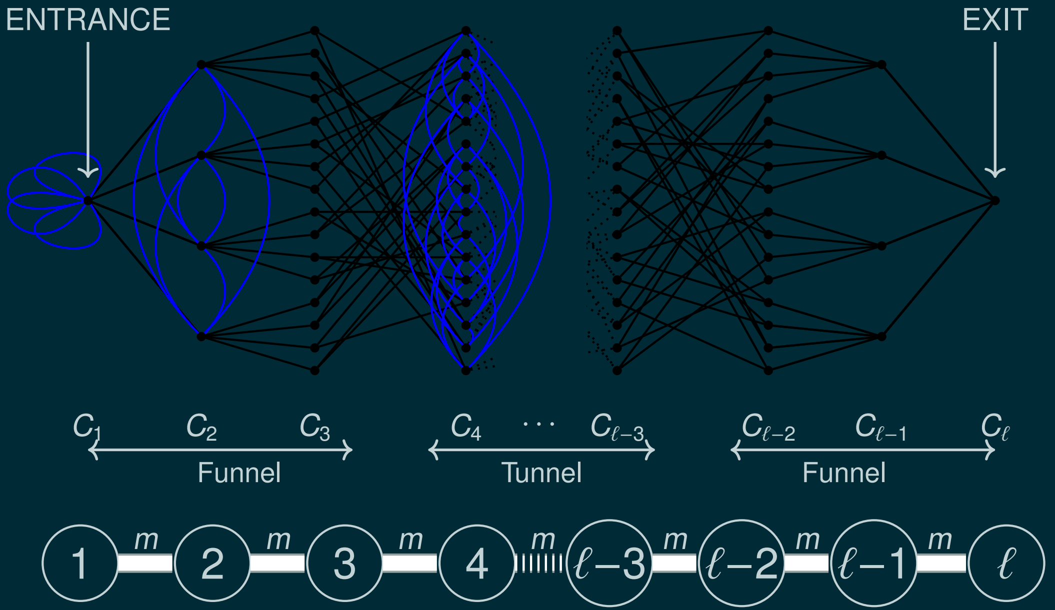
Step 1: obfuscation (tricking path integral Monte Carlo)



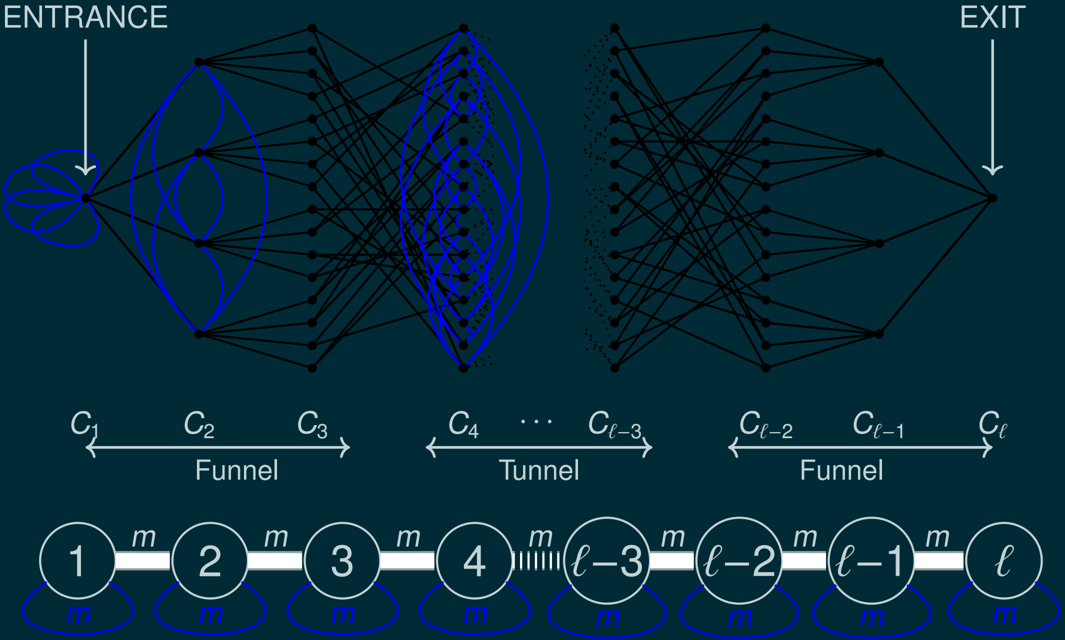
Step 1: obfuscation (tricking path integral Monte Carlo)



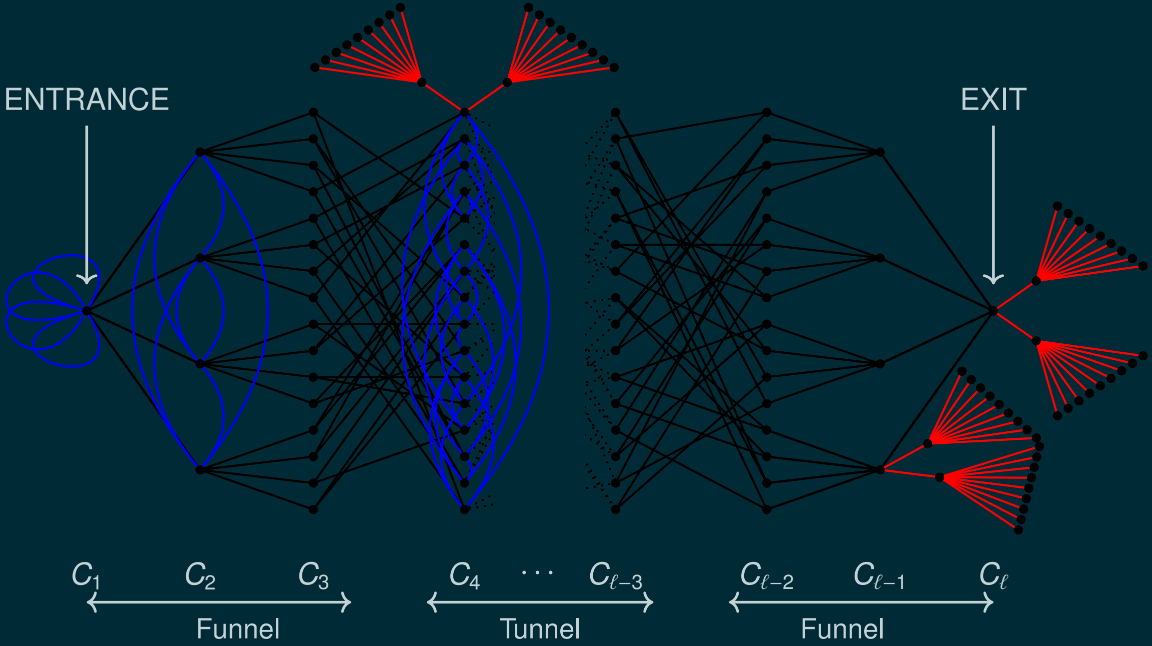
Step 1: obfuscation (tricking path integral Monte Carlo)



Step 1: obfuscation (tricking path integral Monte Carlo)



Step 2a: decoration (tricking diffusion Monte Carlo)



Step 2a: decoration (tricking diffusion Monte Carlo)

- Attach disjoint “camouflage” or “decoration” trees to every vertex

Step 2a: decoration (tricking diffusion Monte Carlo)

- Attach disjoint “camouflage” or “decoration” trees to every vertex
- Each attached tree has a degree bound of $\mathcal{O}(m)$

Step 2a: decoration (tricking diffusion Monte Carlo)

- Attach disjoint “camouflage” or “decoration” trees to every vertex
- Each attached tree has a degree bound of $\mathcal{O}(m)$
- The spectral norm of each tree, and thus the entire forest, is $\mathcal{O}(\sqrt{m})$

Step 2a: decoration (tricking diffusion Monte Carlo)

- Attach disjoint “camouflage” or “decoration” trees to every vertex
- Each attached tree has a degree bound of $\mathcal{O}(m)$
- The spectral norm of each tree, and thus the entire forest, is $\mathcal{O}\left(\sqrt{m}\right)$
- The spectral gap before decoration is $\Delta = \Theta(m/\ell^2)$

Step 2a: decoration (tricking diffusion Monte Carlo)

- Attach disjoint “camouflage” or “decoration” trees to every vertex
- Each attached tree has a degree bound of $\mathcal{O}(m)$
- The spectral norm of each tree, and thus the entire forest, is $\mathcal{O}\left(\sqrt{m}\right)$
- The spectral gap before decoration is $\Delta = \Theta(m/\ell^2)$
- If $\sqrt{m} \gg \ell^2$, then the spectral gap cannot close under $\mathcal{O}\left(\sqrt{m}\right)$ perturbation

Step 2a: decoration (tricking diffusion Monte Carlo)

- Attach disjoint “camouflage” or “decoration” trees to every vertex
- Each attached tree has a degree bound of $\mathcal{O}(m)$
- The spectral norm of each tree, and thus the entire forest, is $\mathcal{O}\left(\sqrt{m}\right)$
- The spectral gap before decoration is $\Delta = \Theta(m/\ell^2)$
- If $\sqrt{m} \gg \ell^2$, then the spectral gap cannot close under $\mathcal{O}\left(\sqrt{m}\right)$ perturbation
- Consequently, the ground state suffers only minor perturbation in ℓ_2-norm

Step 2a: decoration (tricking diffusion Monte Carlo)

- Attach disjoint “camouflage” or “decoration” trees to every vertex
- Each attached tree has a degree bound of $\mathcal{O}(m)$
- The spectral norm of each tree, and thus the entire forest, is $\mathcal{O}(\sqrt{m})$
- The spectral gap before decoration is $\Delta = \Theta(m/\ell^2)$
- If $\sqrt{m} \gg \ell^2$, then the spectral gap cannot close under $\mathcal{O}(\sqrt{m})$ perturbation
- Consequently, the ground state suffers only minor perturbation in ℓ_2-norm
- However, if the trees are $\mathrm{poly}(m)$ deep, then the ℓ_1 weight moves onto the trees!

Step 2b: fractal decoration (tricking every classical alg.)

- If all attached trees have the same depth d, we can effectively filter them out

Step 2b: fractal decoration (tricking every classical alg.)

- If all attached trees have the same depth d, we can effectively filter them out
- After traversing an edge perform a non-backtracking trial walk:
  if a leaf is found we know that a camouflage tree was entered

Step 2b: fractal decoration (tricking every classical alg.)

- If all attached trees have the same depth d, we can effectively filter them out
- After traversing an edge perform a non-backtracking trial walk:
  if a leaf is found we know that a camouflage tree was entered
- To prohibit such algorithms we attach trees that have a fractal shape:

Step 2b: fractal decoration (tricking every classical alg.)

- If all attached trees have the same depth d, we can effectively filter them out
- After traversing an edge perform a non-backtracking trial walk: if a leaf is found we know that a camouflage tree was entered
- To prohibit such algorithms we attach trees that have a fractal shape:
- The trees are designed such that the expected “hitting time” of leafs barely changes while moving deeper into a tree – making the “camouflage” trees very hard to recognize

Step 2b: fractal decoration (tricking every classical alg.)

- If all attached trees have the same depth d, we can effectively filter them out
- After traversing an edge perform a non-backtracking trial walk:
  if a leaf is found we know that a camouflage tree was entered
- To prohibit such algorithms we attach trees that have a fractal shape:
- The trees are designed such that the expected “hitting time” of leafs barely changes while moving deeper into a tree – making the “camouflage” trees very hard to recognize
- Due to the random labeling of vertices no classical algorithm can navigate the decorated graph, and any classical algorithm will “get lost” spending an exponential amount of time in the “camouflage forest” before finding the EXIT.

Open questions

- Find a sign-problem-free Hamiltonian of practical interest providing a large speed-up!
- Are sign-problem-free Hamiltonians at least marginally easier in general?
  Is there a general simulation algorithm that works in time for example $\sqrt{2^n}$?
- What is the classical complexity of simulating adiabatic evolution for frustrated local Hamiltonians with no sign problem? (See, e.g., [Bringewatt & Jarret, 2020].)