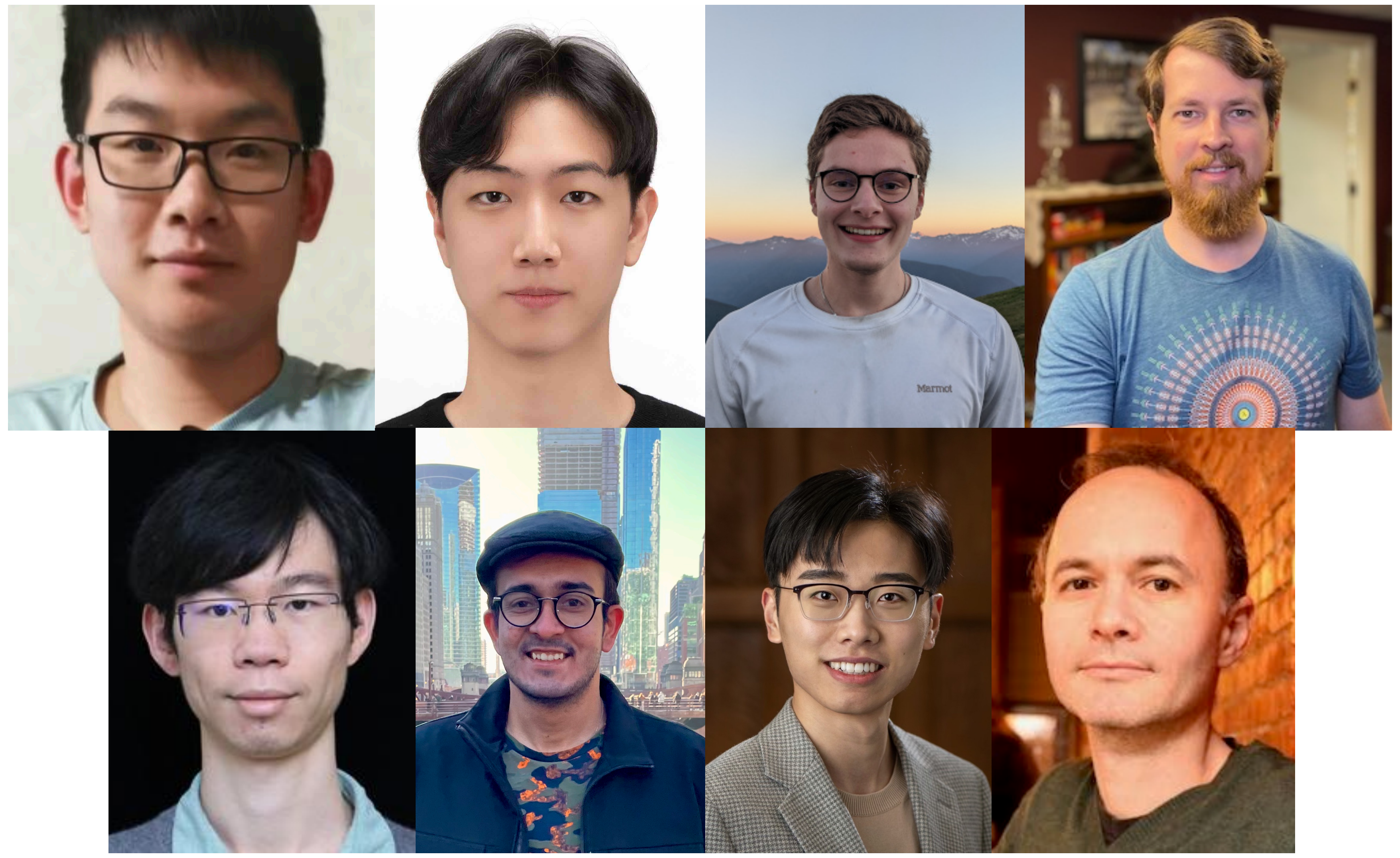
GraFeyn

Efficient Parallel Sparse Simulation of Quantum Circuits



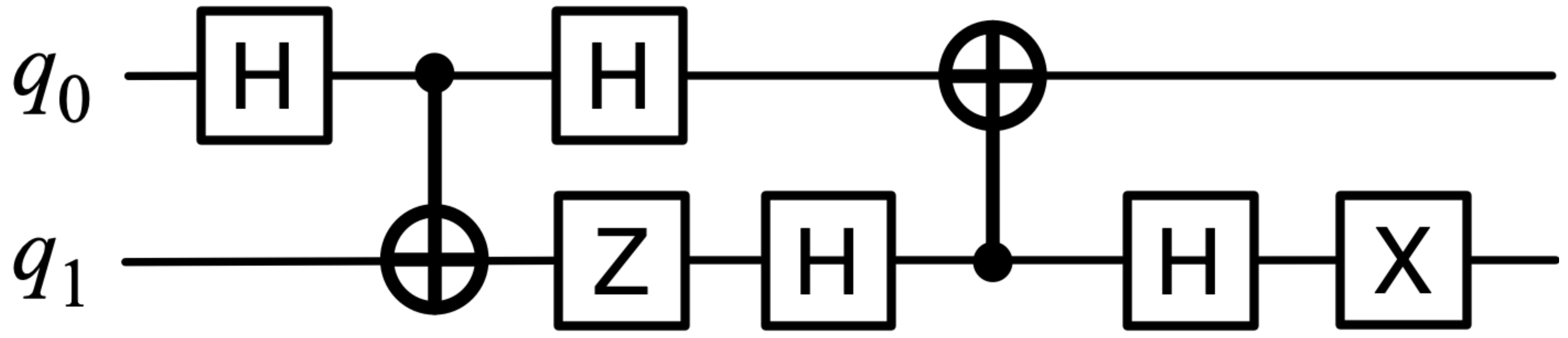
[1] Carnegie Mellon University
[2] New York University
[3] Yale University

Sam Westrick[2] Pengyu Liu[1] Byeongjee Kang[1] Colin McDonald[1] Mike Rainey[1]
Mingkuan Xu[1] Jatin Arora[1] Yongshan Ding[3] Umut Acar[1]

Pure Quantum Simulation

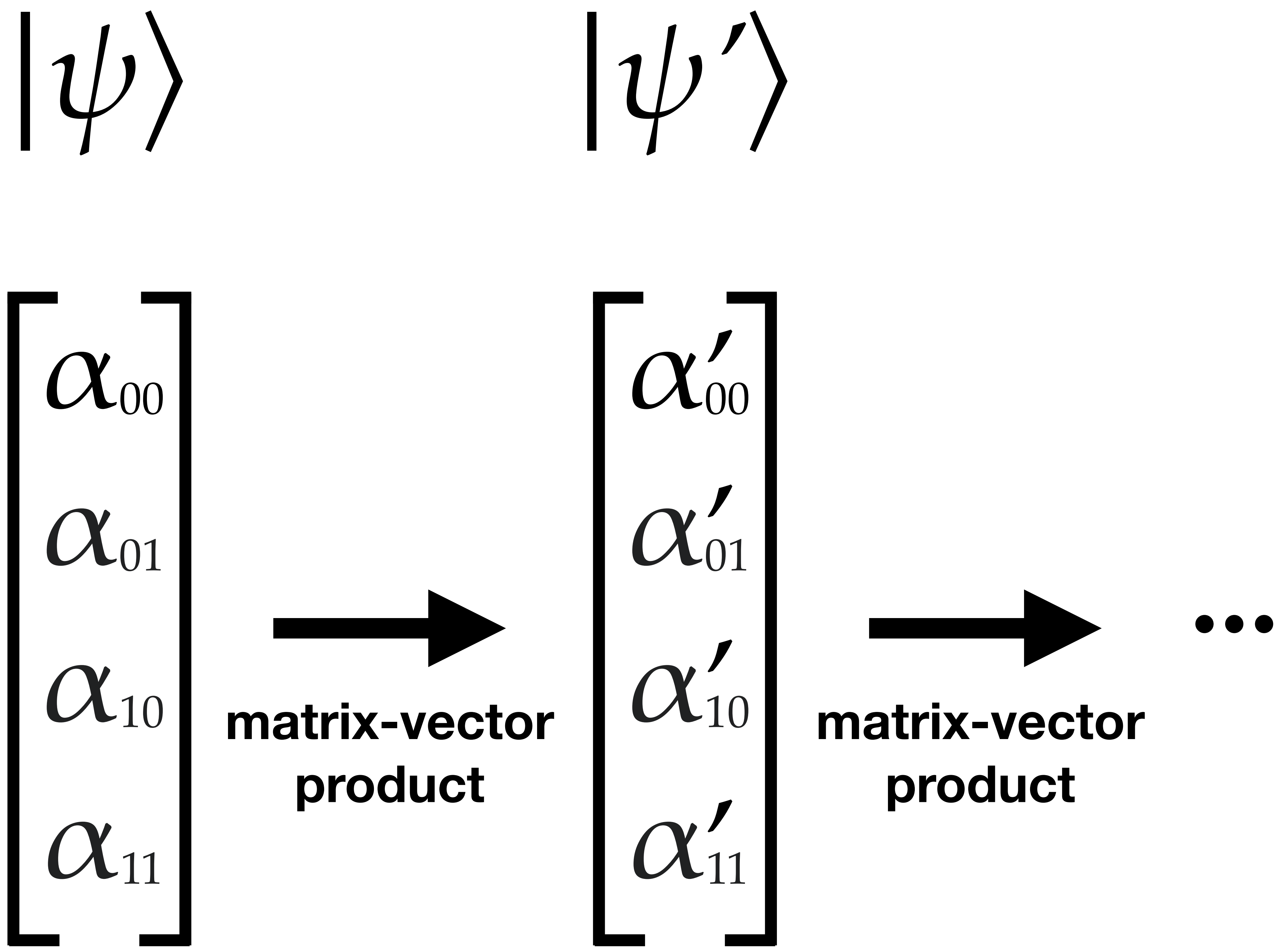
- classical simulation of quantum circuits
- just pure quantum gates (no measurement, resets, barriers, etc.)
- full input and output state vectors
- useful for testing, debugging...
  - and perhaps hybrid quantum/classical?

$|\psi\rangle \rightarrow$

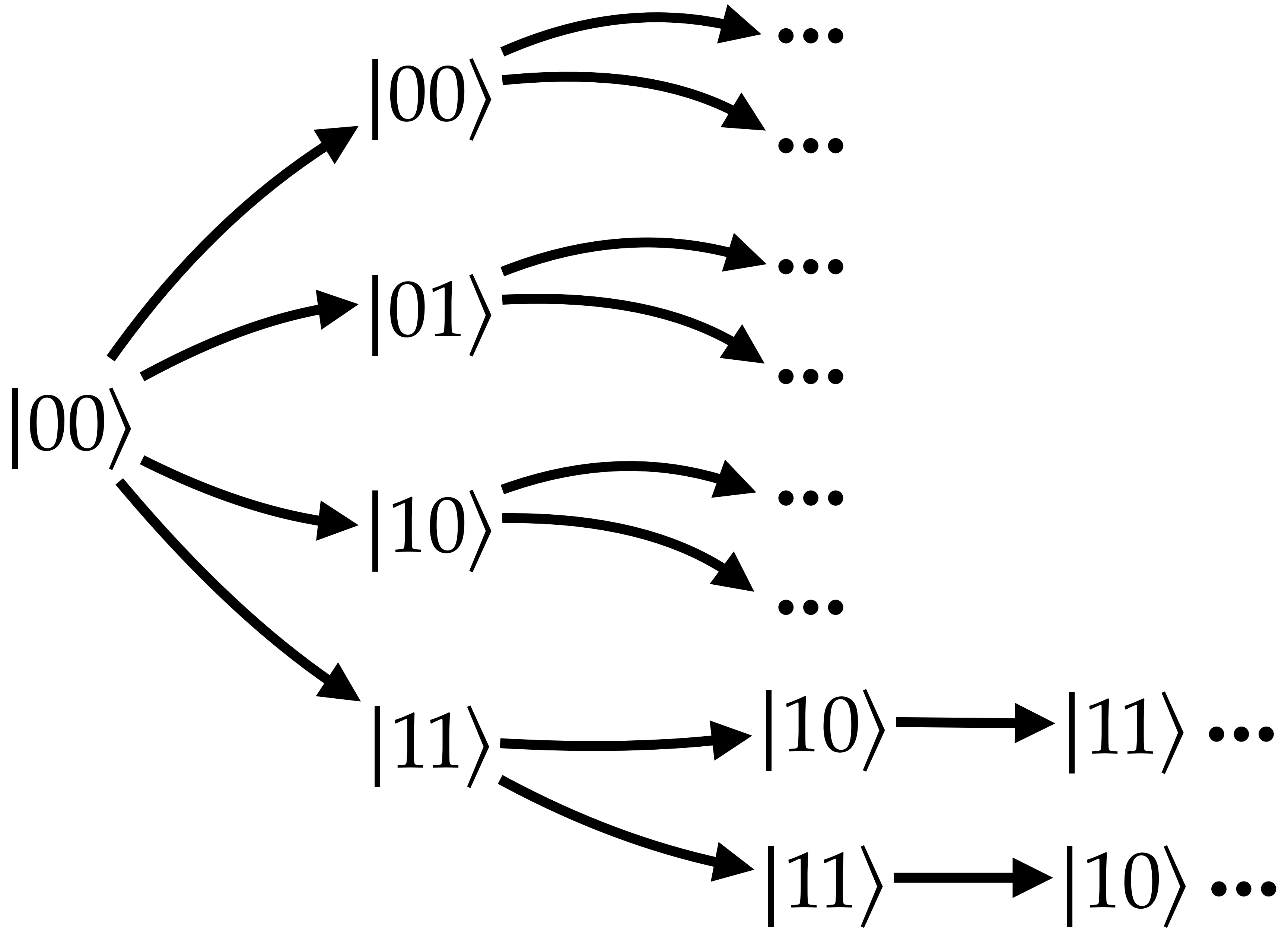


$\rightarrow |\psi'\rangle$

(Very Broadly) Two Approaches to Simulation

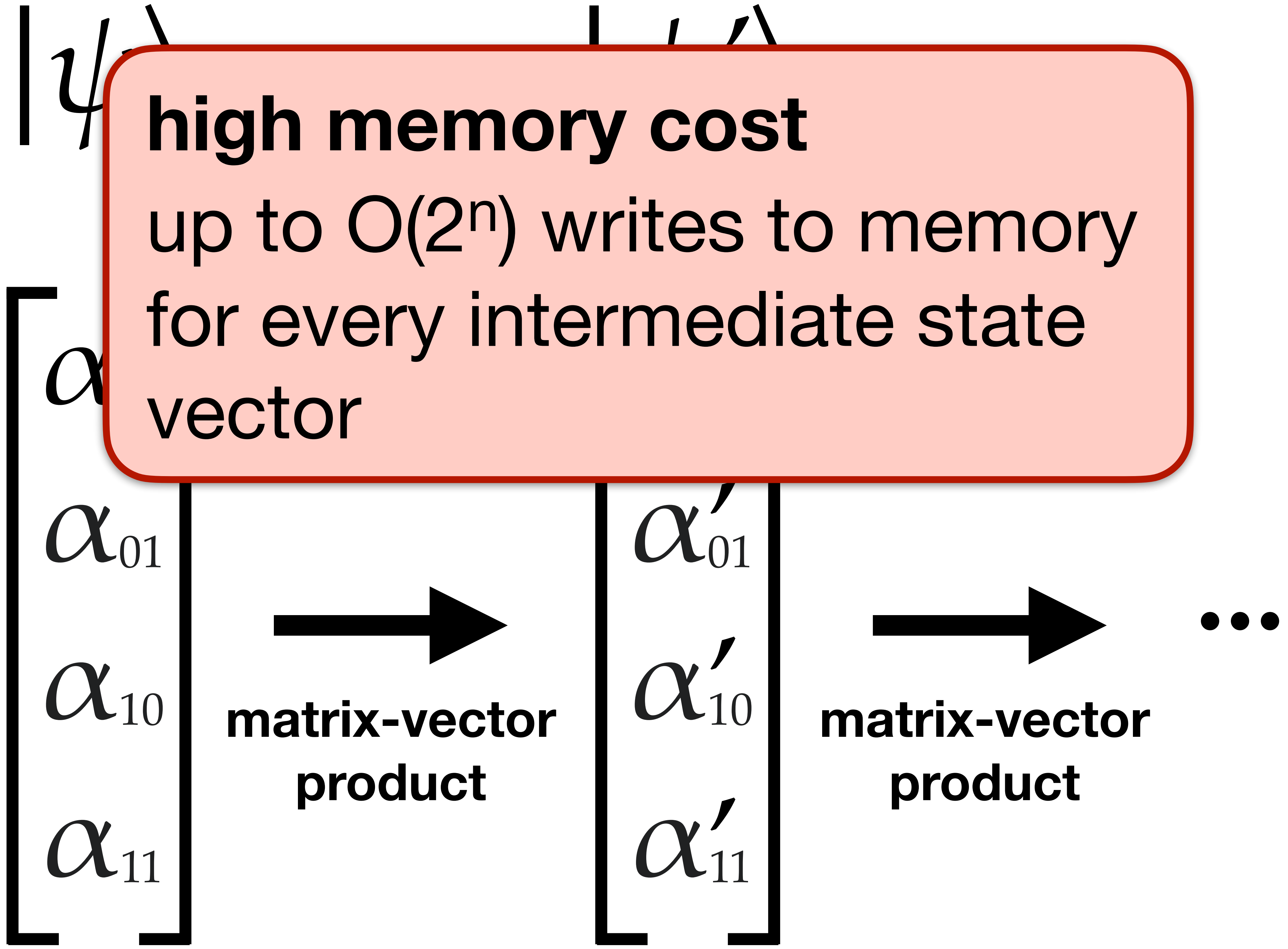


"Schrödinger-style" (state vector)



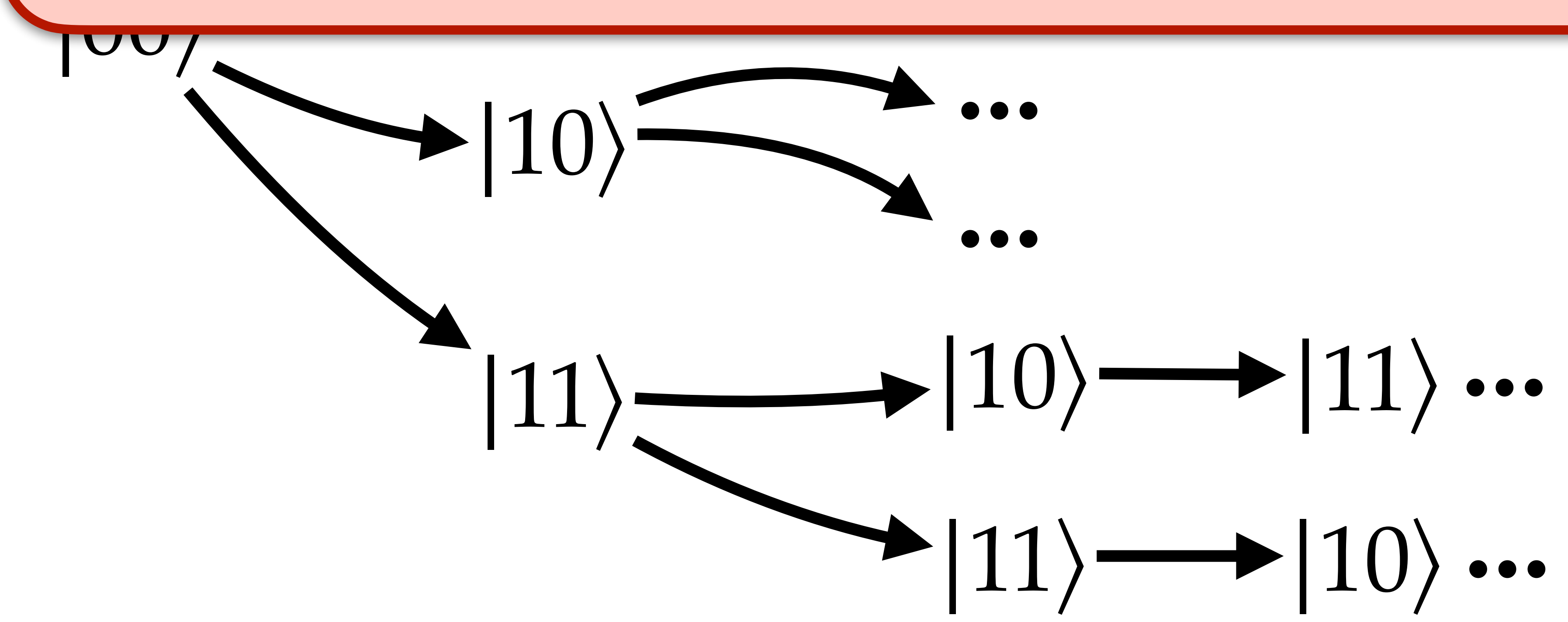
Feynman path summations

(Very Broadly) Two Approaches to Simulation



high memory cost
up to $O(2^n)$ writes to memory for every intermediate state vector

“Schrödinger-style” (state vector)



nearly zero memory cost…

but, can be exponentially slower than Schrödinger-style

Feynman path summations

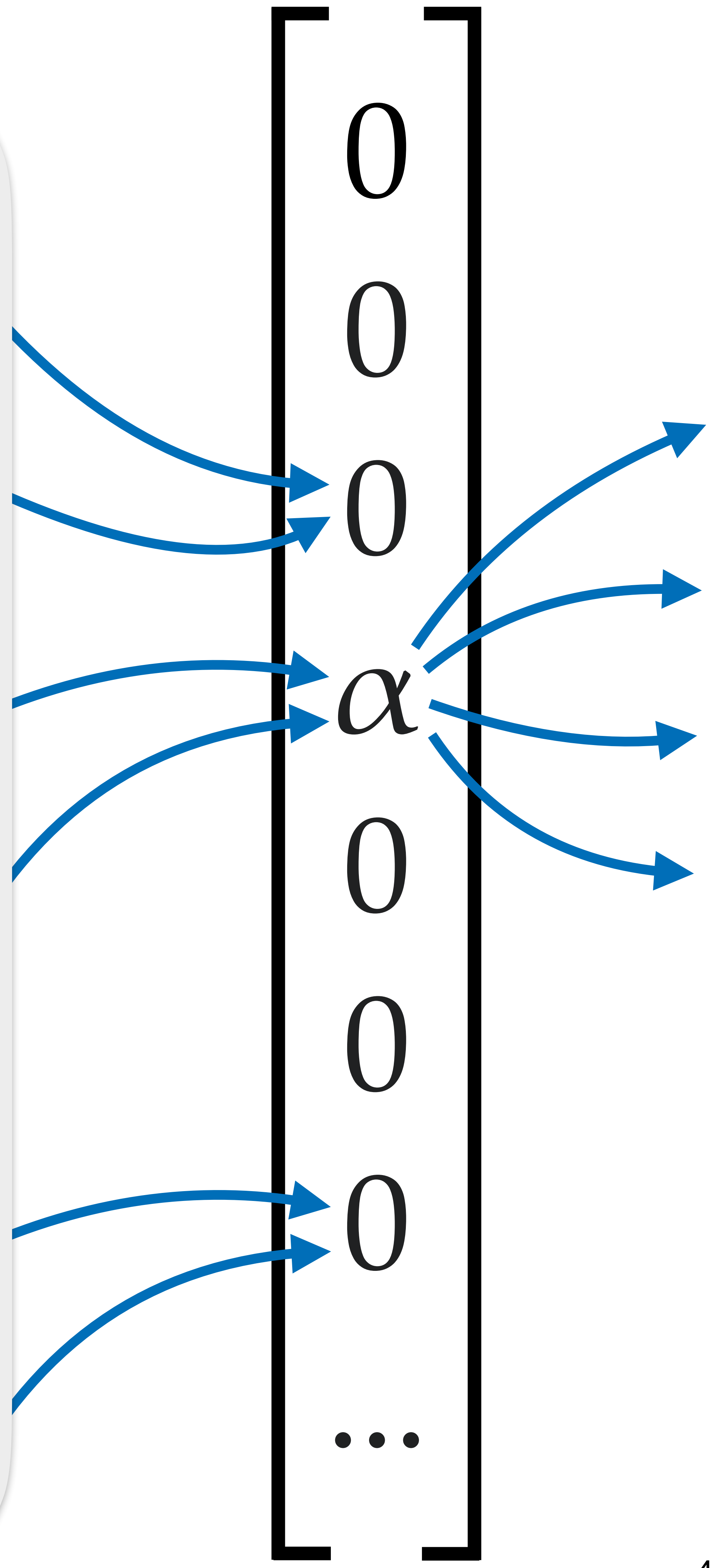
GraFeyn

Getting the Best of Both Worlds

github.com/cmu-top/grafeyn

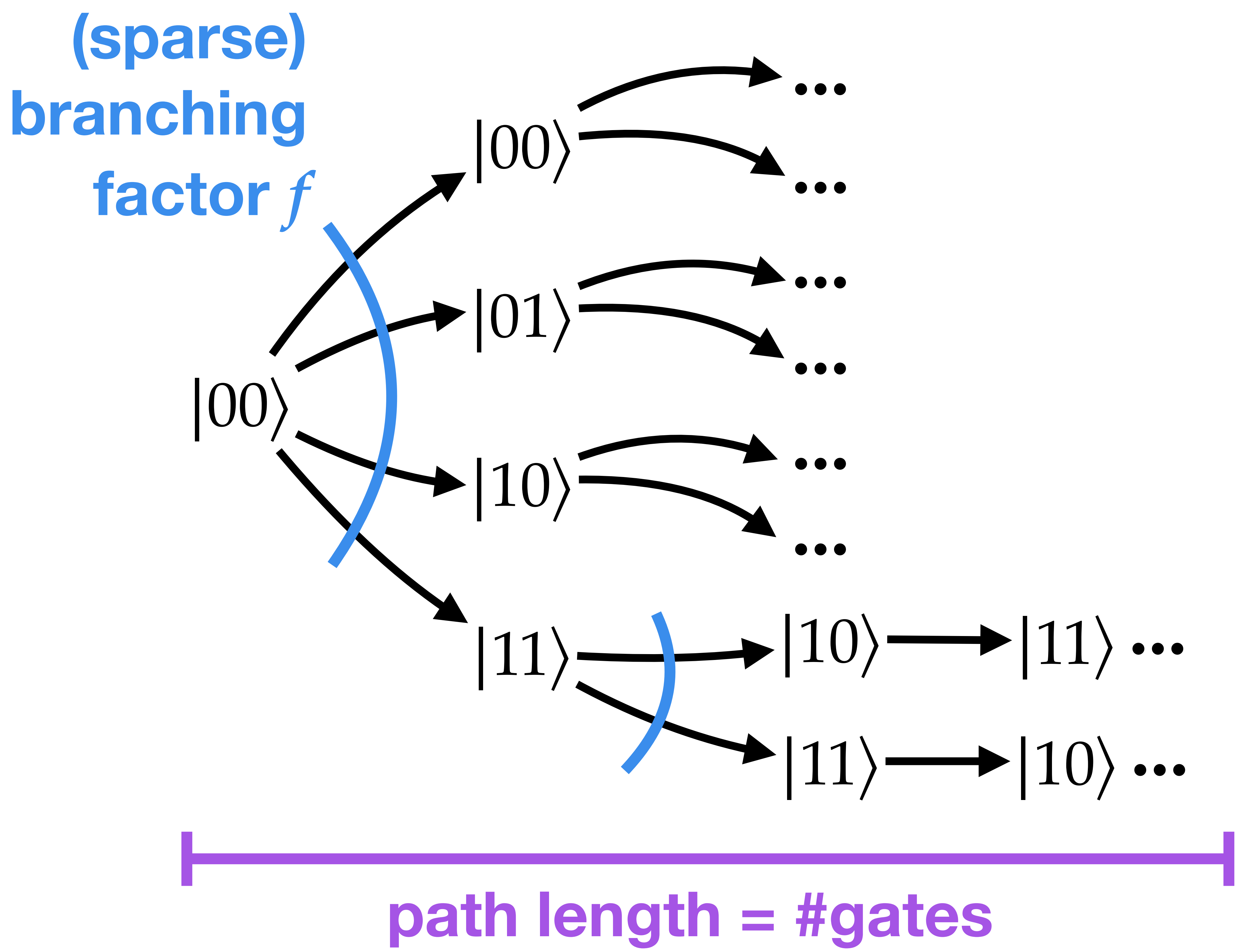
This paper:

- hybrid Feynman-Schrödinger simulator for pure quantum circuits
- exploit **sparsity** for improved performance
    - don't write down the zeros
    - **sparse Feynman paths are efficient!**
    - "synchronize" Feynman paths only when necessary
    - reorder/schedule gates to encourage sparsity
- **easily handles hundreds of qubits on highly sparse circuits**
- comparison with Qiskit and QSim simulators
    - multiple orders of magnitude improvement in some cases



Traversing Feynman Paths...

How (in)efficient is it?

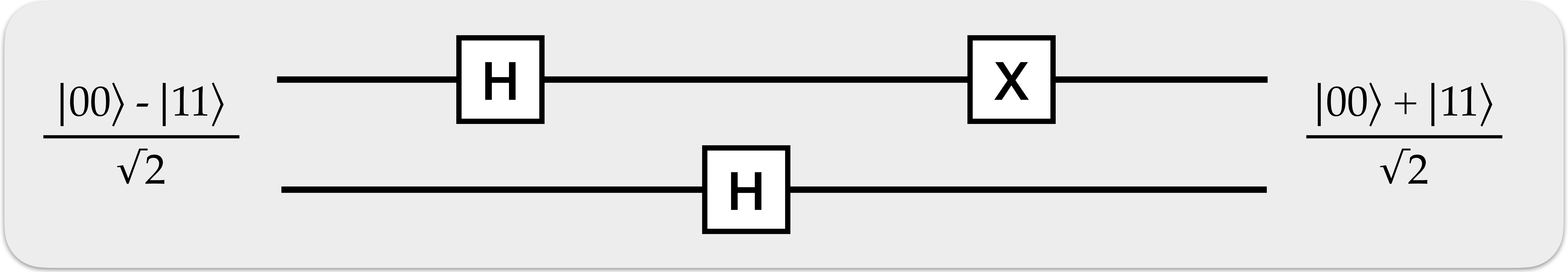


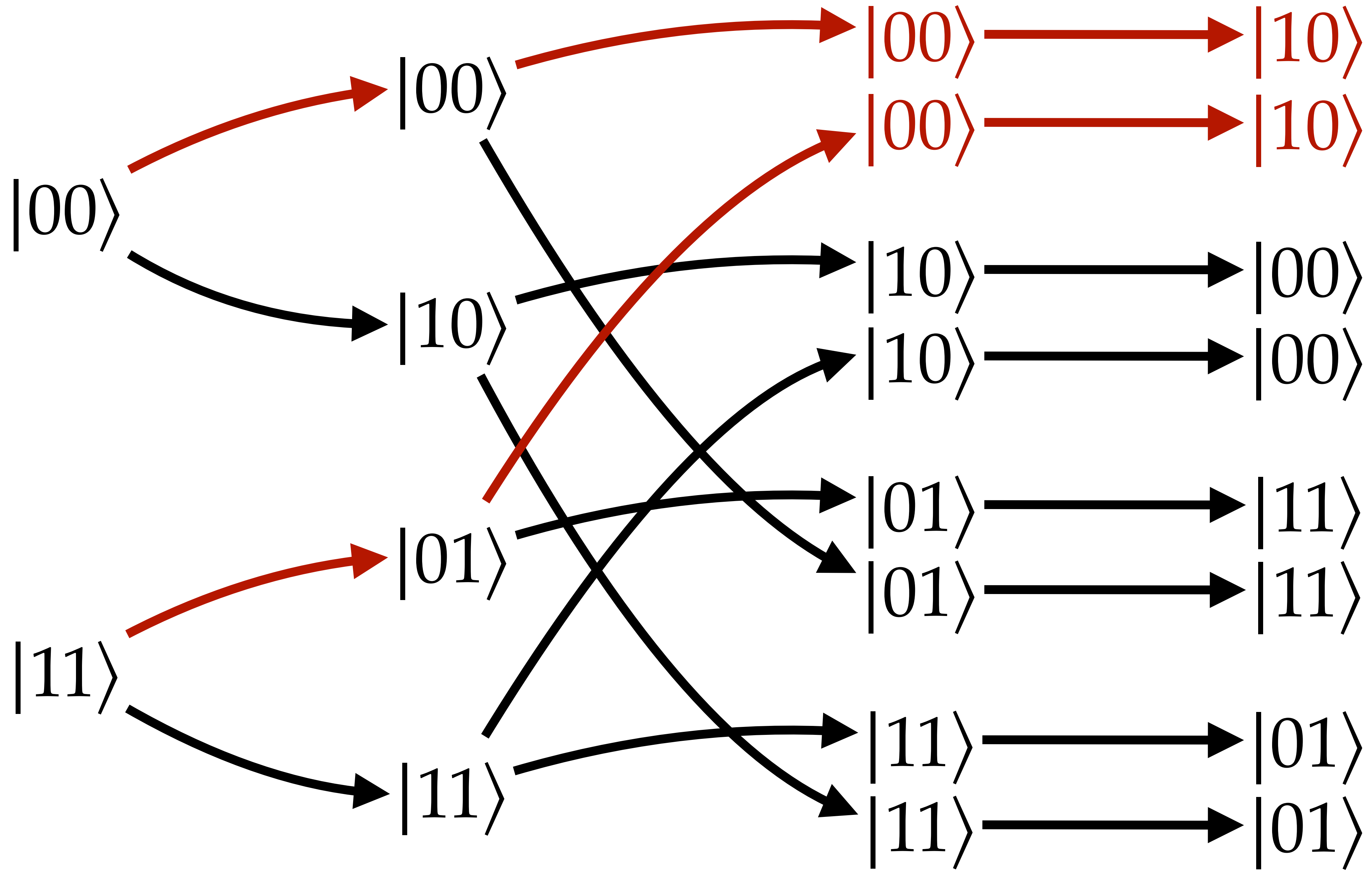
Gate	Description	Branching Factor
X	NOT	1 (*non-branching*)
CX	controlled NOT	1 (*non-branching*)
S	phase	1 (*non-branching*)
Z	phase flip	1 (*non-branching*)
T	$\pi/8$ gate	1 (*non-branching*)
SWAP	qubit swap	1 (*non-branching*)
CCX	Toffoli	1 (*non-branching*)
H	Hadamard	2

number of (sparse) Feynman paths:

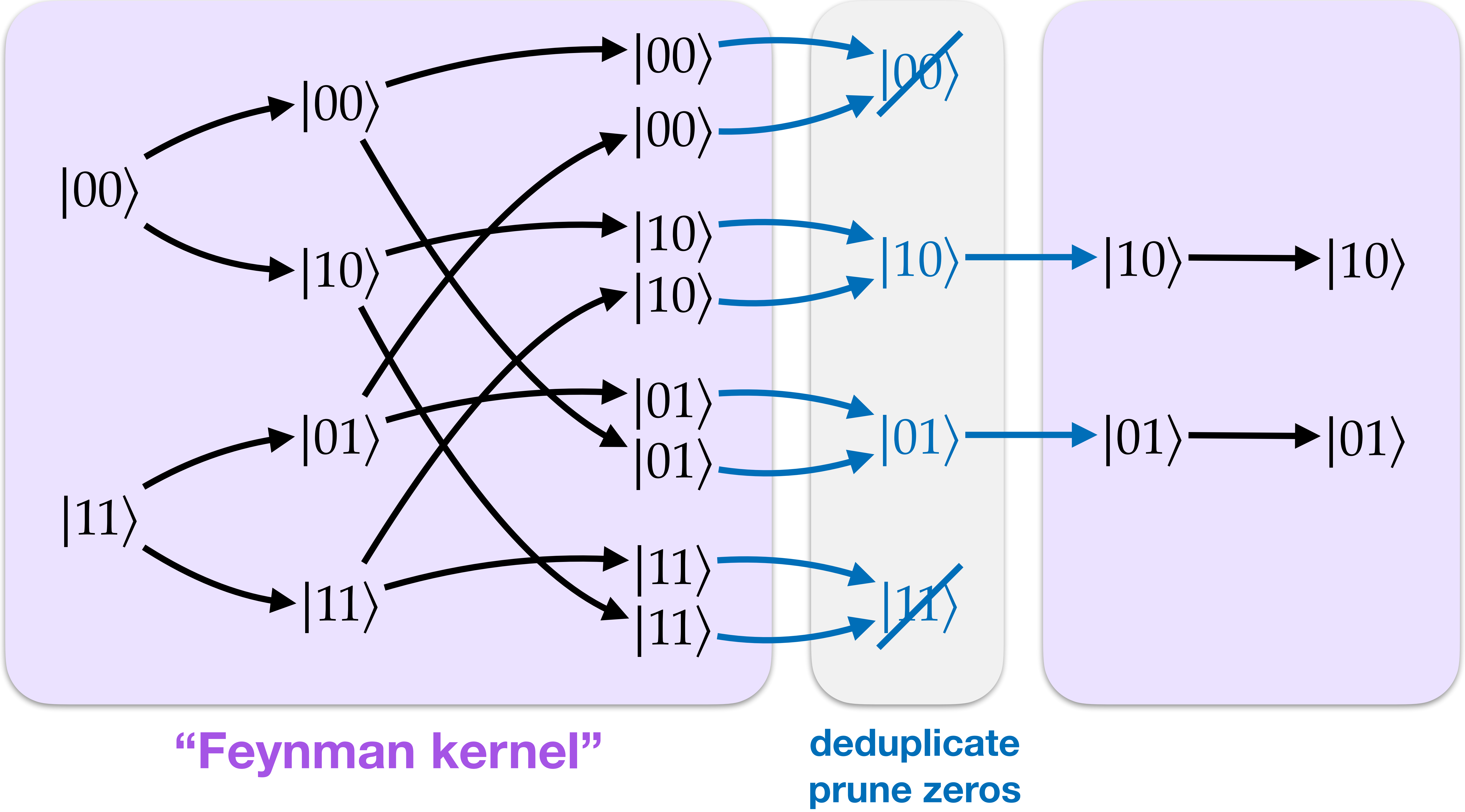
$$\prod_i f(G_i)$$

Feynman Paths “Miss” Interference



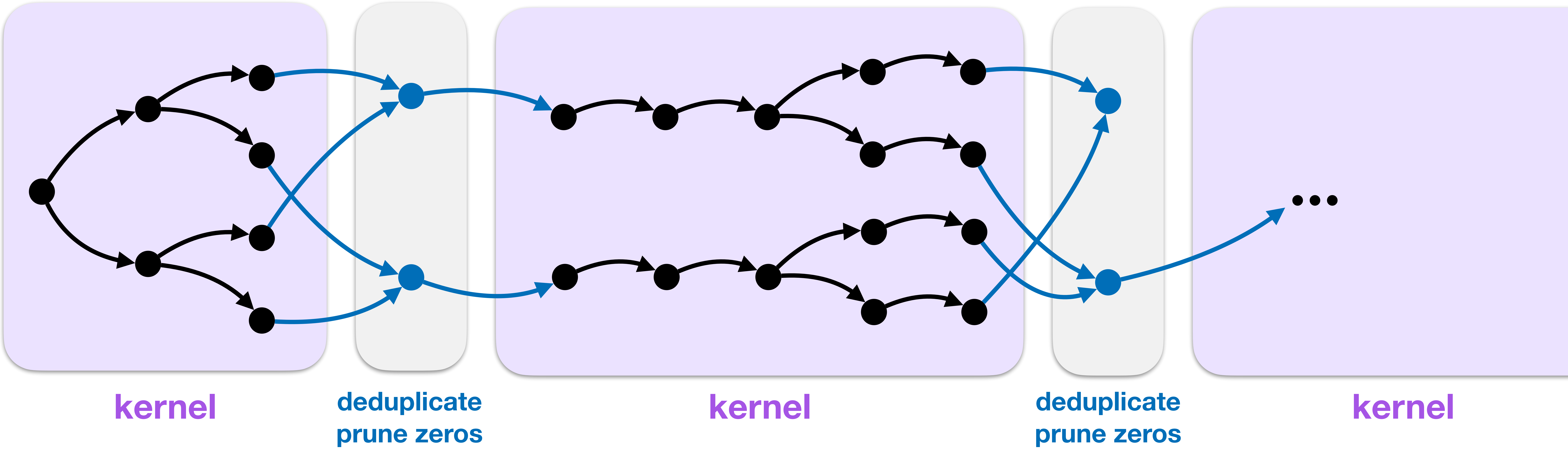


Compute Interference by "Synchronizing" Paths



Essentially a Parallel Graph Traversal...

(the graph is generated on-the-fly)



Selecting Kernels

schedule gates

- goal: reorder (schedule) gates to encourage sparsity
- algorithm: pick a qubit, **finish** it, pick another qubit, finish it, etc.
  - **"finish a qubit":** execute all gates on that qubit as soon as possible (execute required dependencies only, and no other gates)
- intuition: efficiently handles qubits that have deterministic result

delimit kernels

- select maximal runs of gates without exceeding **maximum branching factor** (tunable parameter)
- intuition: bound the number of unsynchronized Feynman paths

GraFeyn Implementation

`github.com/cmu-top/grafeyn`

- open-source, available on GitHub
- implemented in MPL, a high-performance parallel functional language
- supports subset of OpenQASM
- easy to extend with custom gate definitions
  - just a few lines of code per gate

lots of interesting implementation details!

- parallel traversal of paths
- synchronize paths with concurrent (lock-free) hash tables
  - predict number of non-zeros and automatically resize as needed
- **“direction optimization”**: switch between forward / backward Feynman kernels

Results

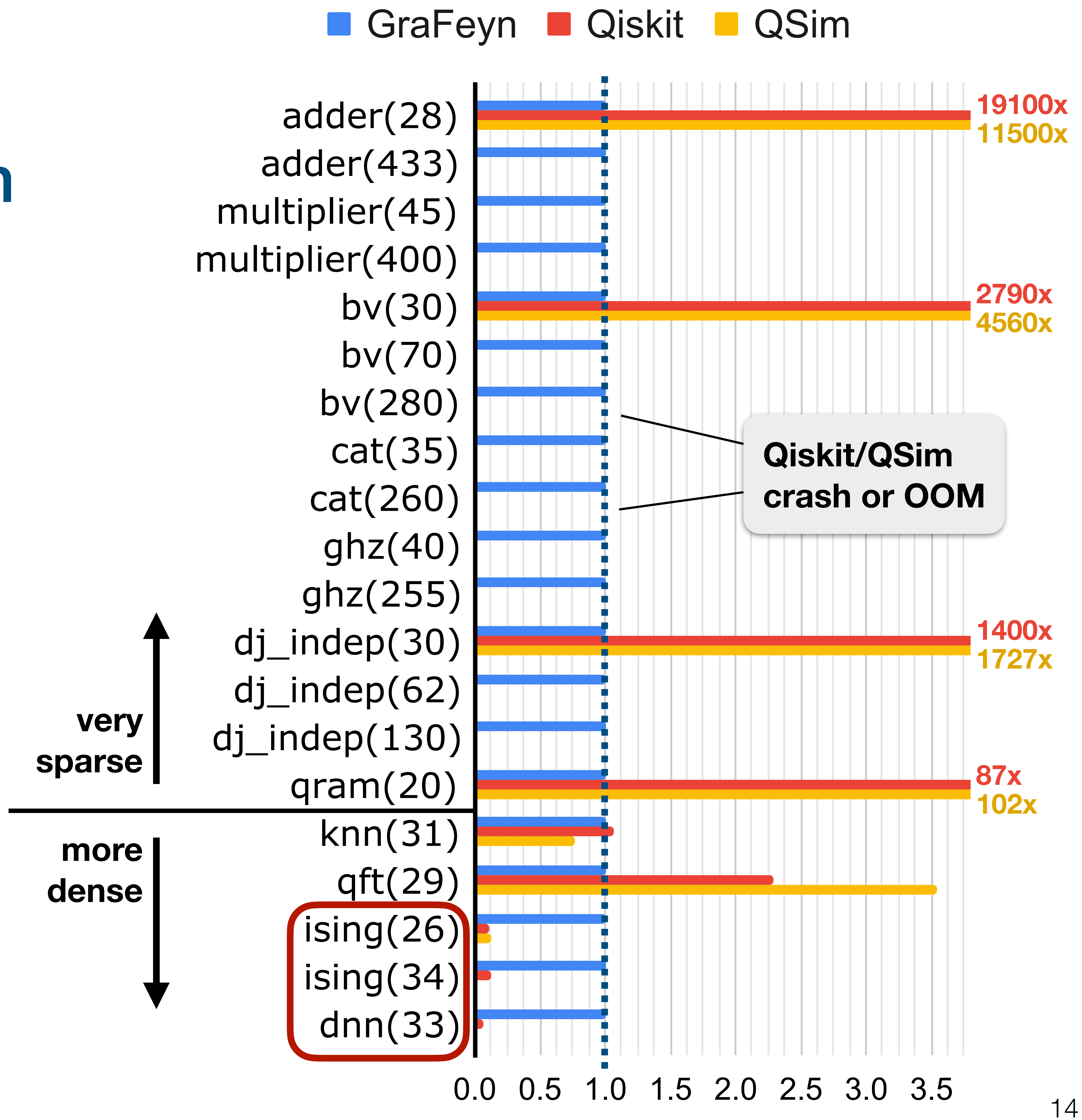
Time normalized w.r.t. GraFeyn

(higher is better for GraFeyn)

multicore CPU execution
64 cores (2x Intel Xeon, 2021)
1TB RAM

comparisons with:
- Qiskit (Aer simulator)
- QSim

benchmarks from:
- QASMBench
- MQTBench



GraFeyn

Getting the Best of Both Worlds

github.com/cmu-top/grafeyn

This paper:

- hybrid Feynman-Schrödinger simulator for pure quantum circuits
- exploit **sparsity** for improved performance
  - don't write down the zeros
  - **sparse Feynman paths are efficient!**
  - "synchronize" Feynman paths only when necessary
  - reorder/schedule gates to encourage sparsity
- **easily handles hundreds of qubits on highly sparse circuits**
- comparison with Qiskit and QSim simulators
  - multiple orders of magnitude improvement in some cases