mathematics

Mathematics 11, 4222 (2023)
https://doi.org/10.3390/math11194222

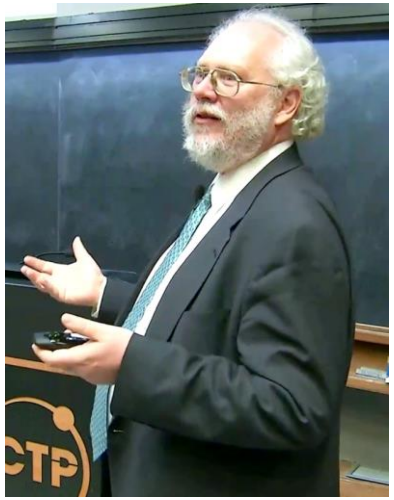
Article

Large-Scale Simulation of Shor's Quantum Factoring Algorithm

Dennis Willsch [1,*], Madita Willsch [1,2], Fengping Jin [1], Hans De Raedt [1,3] and Kristel Michielsen [1,2,4]

SHOR'S QUANTUM FACTORING ALGORITHM

Overview



- Given a composite integer N, find a nontrivial factor 1 < p < N
- For an L-bit **semiprime** N = p*q, find a prime factor p
  - We don't know a classical algorithm that runs in polynomial time (in L)
  - A lot of **Internet security** is based on this "hardness"

    → Public key cryptosystems like RSA used in TLS, SSH, …

- **Shor's algorithm** for a gate-based quantum computer can find p in polynomial time

$$\mathcal{O}(e^{\cdots L \cdots}) \longrightarrow \mathcal{O}(\cdots L \cdots)$$

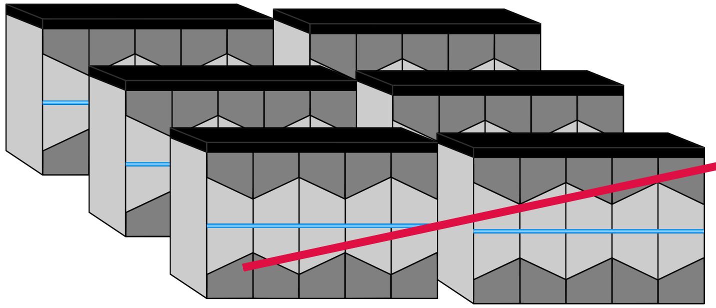
SIMULATING QUANTUM COMPUTERS ON GPUS

The quantum state

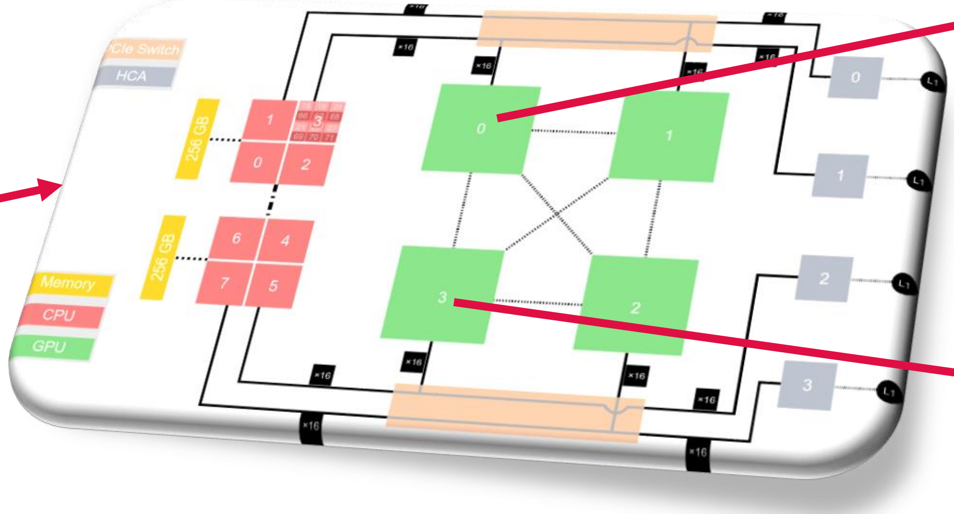
- Each simulation step transforms a quantum state $|\psi\rangle = (\psi_{\cdots q_3 q_2 q_1 q_0})$
- Distribute all complex numbers on the GPUs (**NVIDIA A100: ≤** 40GB per GPU)

For 40 qubits: 2^{40} $\psi'\mathrm{s}$ = 16 TiB complex numbers = 16 GiB per GPU with 1024 GPUs

JUWELS Booster
936 compute nodes



4 A100 GPUs per node



GPU rank 0 = 0b0000000000:

$$\begin{pmatrix} \psi_{0000000000\,0\cdots0} \\ \vdots \\ \psi_{0000000000\,1\cdots1} \end{pmatrix}$$

GPU rank 3 = 0b0000000011:

$$\begin{pmatrix} \psi_{0000000011\,0\cdots0} \\ \vdots \\ \psi_{0000000011\,1\cdots1} \end{pmatrix}$$

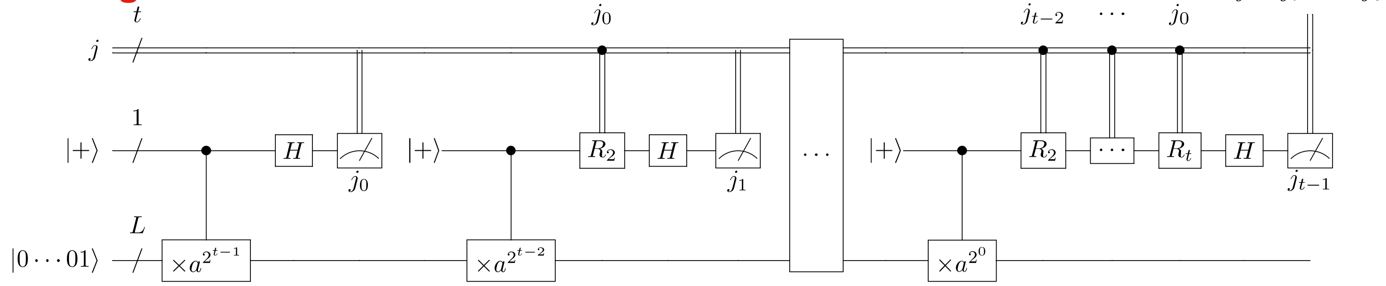
QUANTUM FACTORING ALGORITHM SIMULATOR

qubits n	semiprime N	t
5	15	8
6	21	9
⋮	⋮	
40	549 755 813 699	78
40	549 755 813 701	78
40	549 755 813 713	78
⋮	⋮	

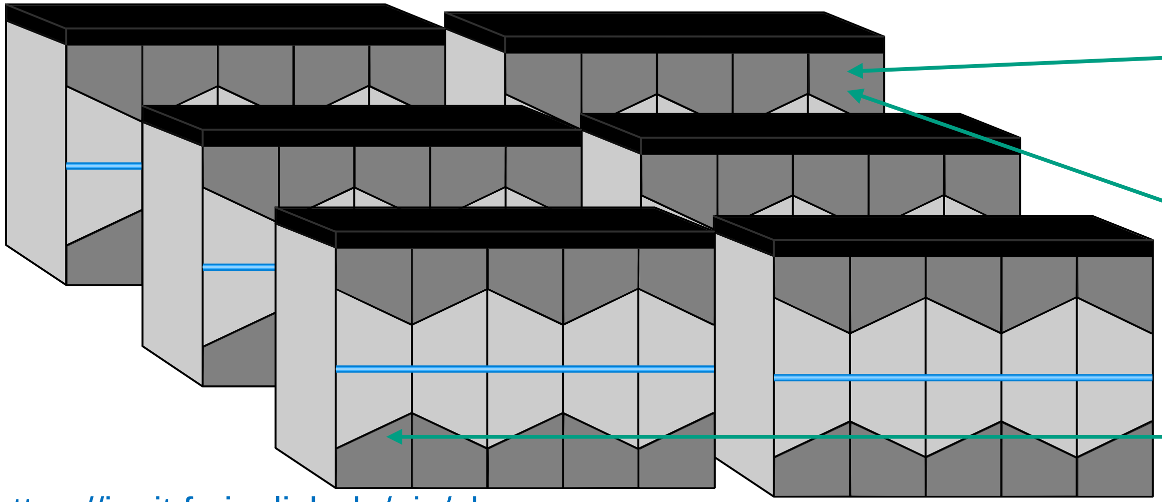
select problem

$N = 549\,755\,813\,701$
$a = 439\,314\,602\,906$

Shor algorithm



SHORGPU simulation on 2048 GPUs



GPU #0

$$\begin{pmatrix} \psi_{00000000000\,0\cdots0} \\ \vdots \\ \psi_{00000000000\,1\cdots1} \end{pmatrix}$$

GPU #1

$$\begin{pmatrix} \psi_{00000000001\,0\cdots0} \\ \vdots \\ \psi_{00000000001\,1\cdots1} \end{pmatrix}$$

GPU #2047

$$\begin{pmatrix} \psi_{11111111111\,0\cdots0} \\ \vdots \\ \psi_{11111111111\,1\cdots1} \end{pmatrix}$$

https://jugit.fz-juelich.de/qip/shorgpu

produces bitstring with t bits

$$j = 111100110111101001111100011011000100000000111011111101110110110110011100010011_2$$
$$= 287\,448\,630\,931\,475\,209\,611\,027$$

extract order

$s/r \approx j/2^t$
$r = 4\,581\,286\,080$

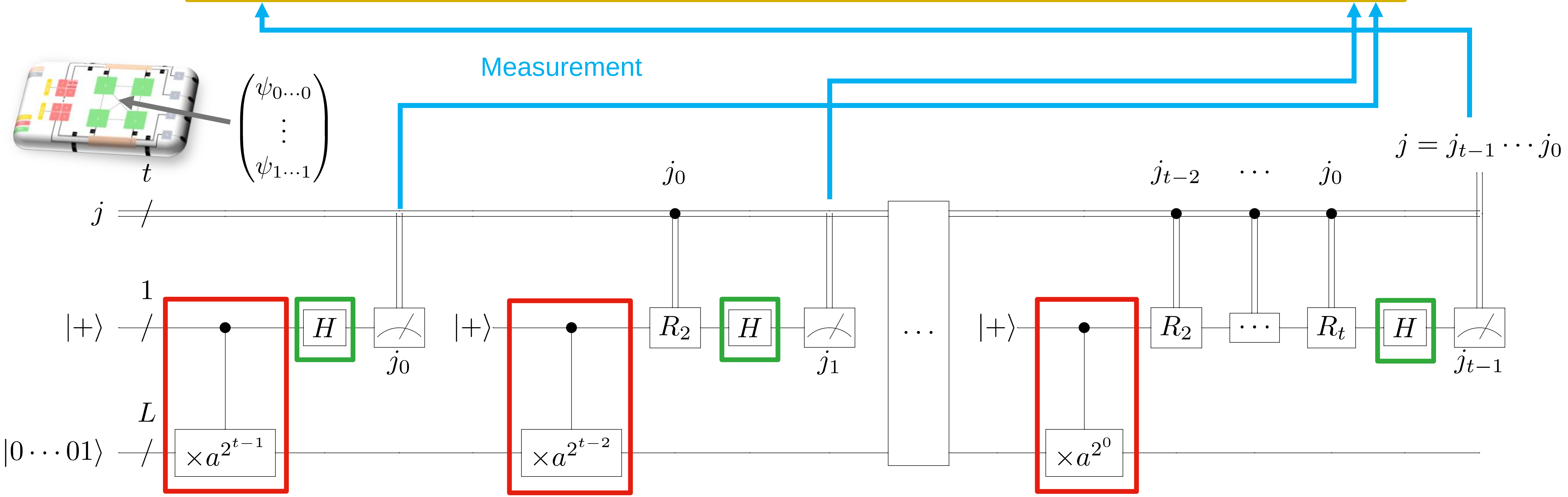
find factors

$549\,755\,813\,701 =$
$712\,321 \times 771\,781$
$\uparrow \gcd(a^{\lfloor r/2 \rfloor}+1, N)$ $\uparrow \gcd(a^{\lfloor r/2 \rfloor}-1, N)$

QUANTUM FACTORING ALGORITHM SIMULATOR

Iterative Shor Algorithm: Details

$$j = 1111001101111010011111000110110001000000000111011111101110110110110011100010011_2$$

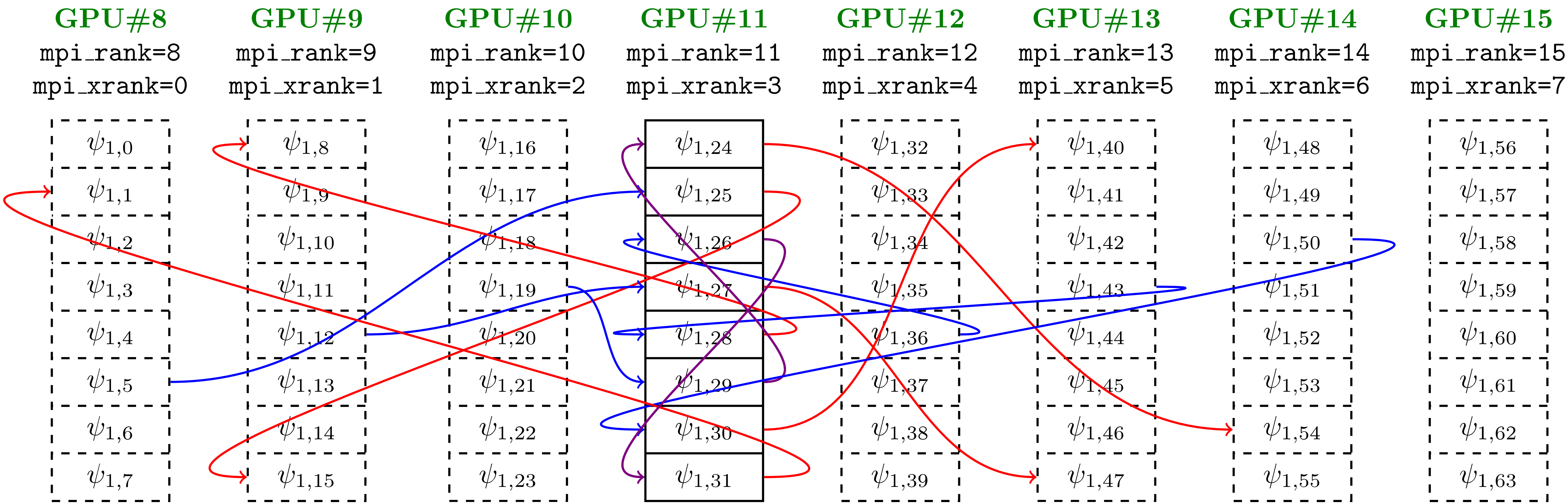


- Controlled modular multiplication gates: Complicated All-to-All MPI Communcation
- Hadamard gates: Pairwise Half-of-all-Memory MPI Exchange

SHORGPU
https://jugit.fz-juelich.de/qip/shorgpu

QUANTUM FACTORING ALGORITHM SIMULATOR

All-to-All MPI Communication Scheme



Every $\psi_{1,y}$ has to be sent to $\psi_{1,y'}$ where $y' = ay \bmod N$

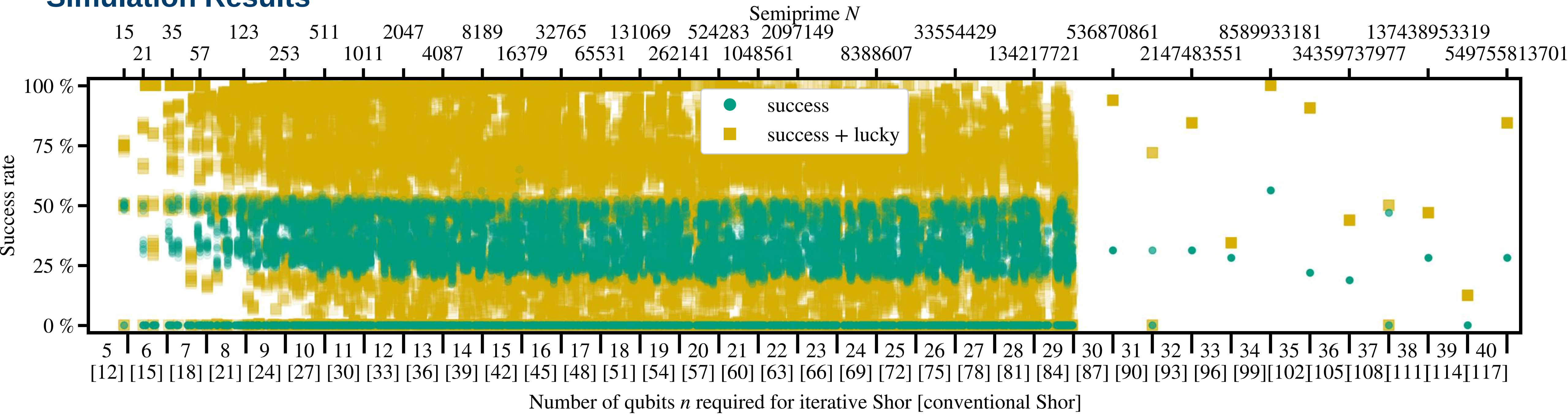
- GPU#11 computes $ay \bmod N$ and MPI_Isend()'s all $\psi_{1,y}$
- GPU#11 computes $a^{-1}y' \bmod N$ and MPI_Irecv()'s all $\psi_{1,y'}$

Interestingly: Faster than **MPI_Alltoallv** and **NCCL**

SHORGPU
https://jugit.fz-juelich.de/qip/shorgpu

QUANTUM FACTORING ALGORITHM SIMULATOR

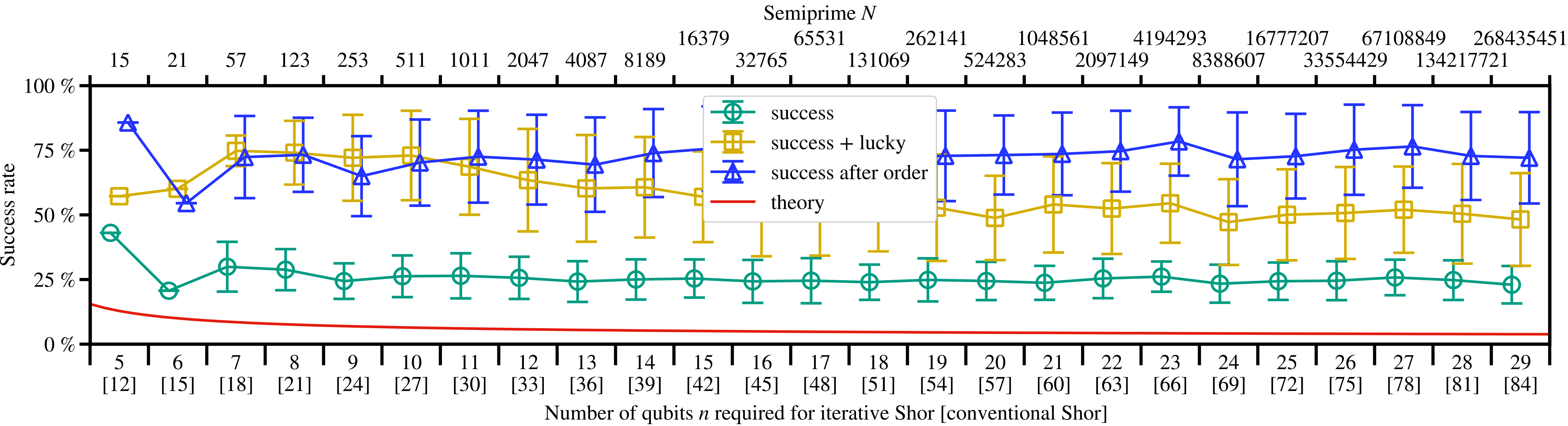
Simulation Results



- Iterative Shor Algorithm needs L+1 qubits for an L-bit semiprime
- Ran Shor’s algorithm for 60000 factoring problems
- Individual large-scale scenarios on 2048 GPUs up to $N = 549\,755\,813\,701$
- **Observation**: There are “lucky” scenarios! “lucky”: factoring works even though theoretical conditions not satisfied

QUANTUM FACTORING ALGORITHM SIMULATOR

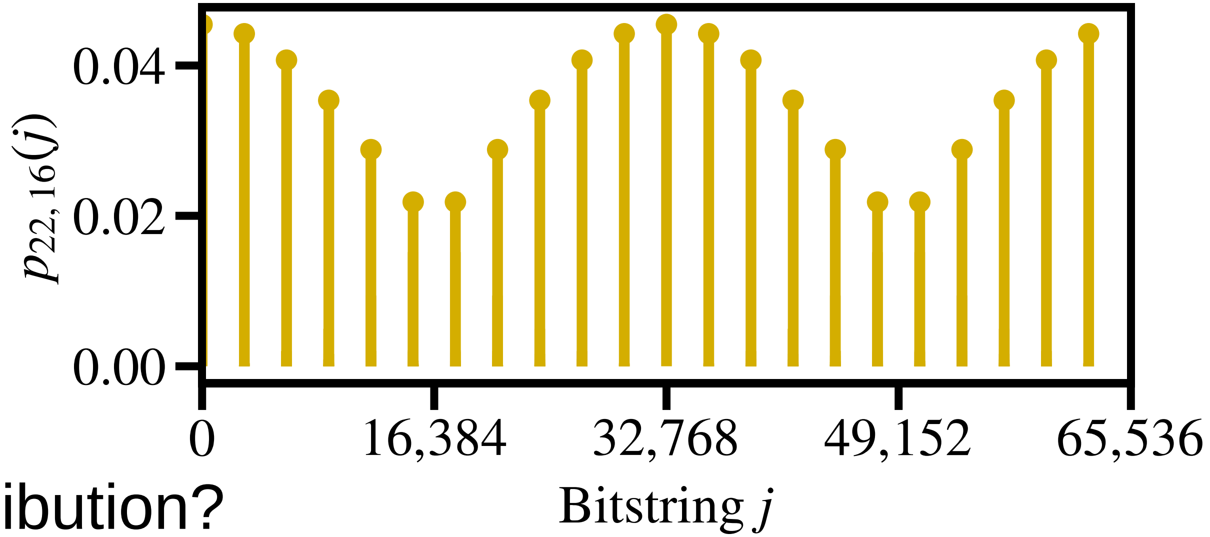
Simulation Results



- **Average shows:** Performance much better than predicted by theory
- Success rate approaches ~ 25 % (without "lucky") and ~ 50 % (with "lucky")
- **Open question**: Does this surprising performance stay for larger semprimes?

FUTURE WORK

Idea 1: Simulation for larger semiprimes



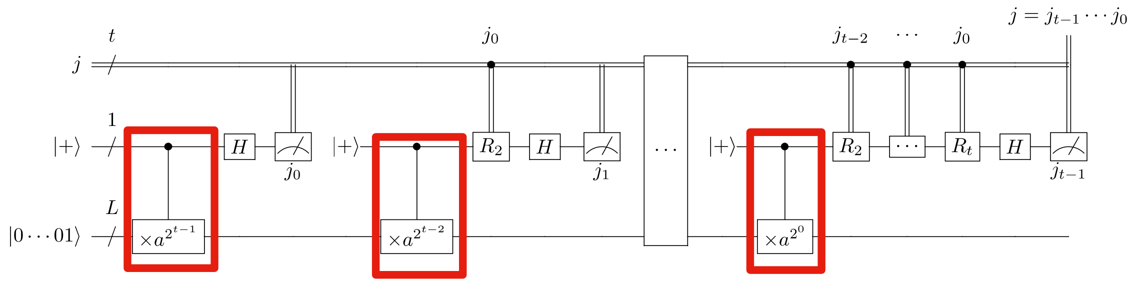
Ekera, "Qunundrum"
https://github.com/ekera/qunundrum

- Simulate more on **exascale** computers!
- **Also:** Improvements to the algorithm?
  - Sample from theoretical probability distribution?

$$p_{\hat{r},t}(j) = \frac{\hat{r}}{2^{2t}} \left(\frac{\sin(\pi \hat{r} j \lfloor \frac{2^t}{\hat{r}} \rfloor / 2^t)}{\sin(\pi \hat{r} j / 2^t)} \right)^2 + \frac{2^t - \hat{r} \lfloor \frac{2^t}{\hat{r}} \rfloor}{2^{2t}} \frac{\sin(\pi \hat{r} j [2 \lfloor \frac{2^t}{\hat{r}} \rfloor + 1] / 2^t)}{\sin(\pi \hat{r} j / 2^t)}$$

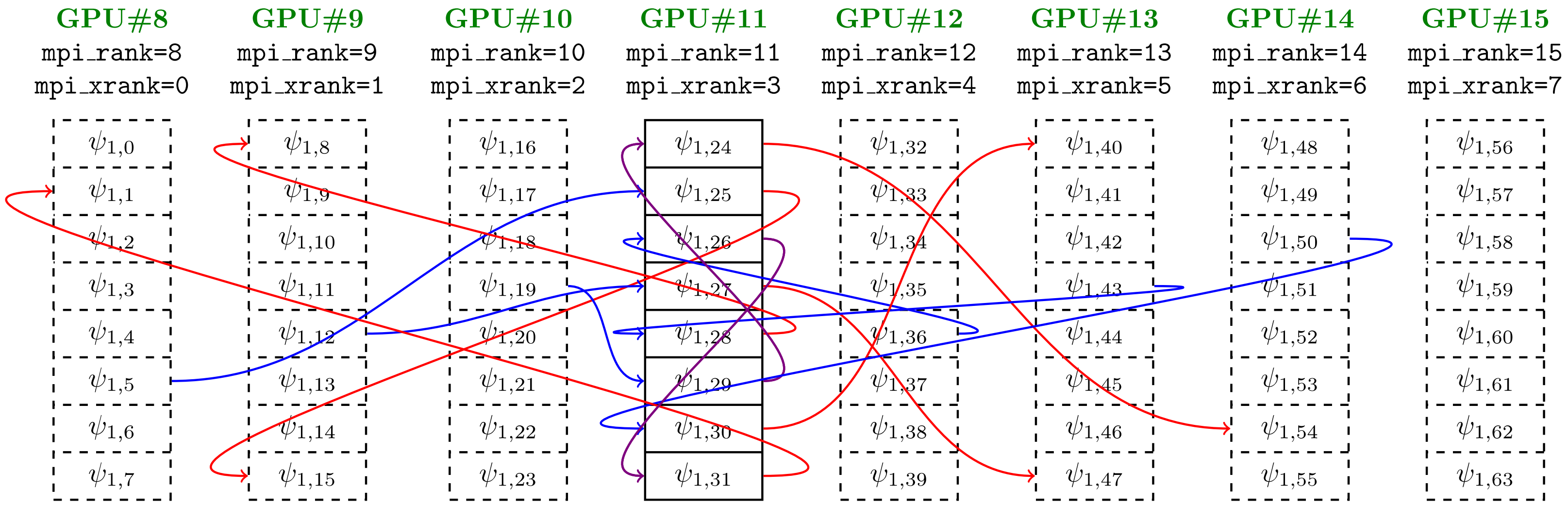
  - **But:** Only works if solution to factoring problem is known
  - **And:** Cannot simulate errors
- Improvements from **number theory**?
  - Can obtain accurate formula for the **average** success probability [in some cases]

FUTURE WORK



Idea 2: Full simulation of multiplication circuits

➢ **So far:** Implemented “oracles” (= controlled multiplication gates) as unitary permutation



➢ **Instead:** Can decompose into standard quantum gate set →

- ➢ Can only simulate half-as-large prolems
- ➢ **But:** Would allow simulation of gate errors!

Takes 2L+3 or 2L+1 qubits
Beauregard, arXiv:quant-ph/0205095 (2003)
Gidney, arXiv:1706.07884 (2018)
(or ~ 1.5L qubits)
Zalka, arXiv:quant-ph/0601097 (2006)
instead of L+1 qubits!

JÜLICH Forschungszentrum | JUNIQ QUANTUM USER FACILITY

SUMMARY

- **SHORGPU** allows us to run Shor's algorithm on 2048 GPUs up to $N = 549\,755\,813\,701$
  - On quantum computers, Shor's algorithm has been run [properly] for $N = 15, 21, 35$

    Smolin et al., Nature 499, 7457 (2013)
    Martín-López et al., Nat. Photonics 6, 773 (2012)
    Monz et al., Science 351, 1068 (2016)
    Amico et al., Phys. Rev. A 100, 012305 (2019)
  - On quantum annealers, an alternative factoring algorithm has been run until $N = 8\,219\,999$

    Andriyash et al., Tech. Rep. 14-1002A-B (2016)
    Jiang et al., Sci. Rep. 8, 17667 (2018)
    Ding et al., Sci. Rep. 14, 3518 (2024)
- Ideas for future work:
  - Larger semiprimes → Exascale!
  - Full multiplication circuits → Simulate gate errors!

THANK YOU FOR YOUR ATTENTION

- More information and references:
  - **SHORGPU:** https://jugit.fz-juelich.de/qip/shorgpu and Mathematics 11, 4222 (2023)
  - **JUQCS:** De Raedt et al., Comput. Phys. Commun. 237, 41 (2019)
  - **JUQCS-G:** Willsch et al., Comput. Phys. Commun. 278, 108411 (2022)