Title: Disentangling quantum systems: a new perspective in computational quantum physics

Date: Feb 02, 2005 03:50 PM

URL: http://pirsa.org/05020003

Abstract:

Disentangling quantum systems: a new perspective in Computational Quantum Physics

PIquDoS

Perimeter Institute, February 2nd 2005

Guifre Vidal - Institute for Quantum Information - CALTECH

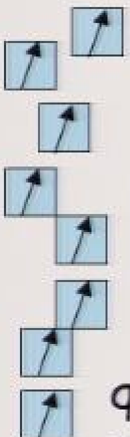
Outline

1) Simulation of Quantum Systems.
2) Efficient Decomposition vs Efficient Simulation.
3) Disentangling Quantum Systems.
4) Summary of Recent Progress
5) Conclusions

1 - Simulation of Quantum Systems

1 - Simulation of Quantum Systems

experimental setting: many-body state $\left|\Phi_{(n)}\right\rangle$

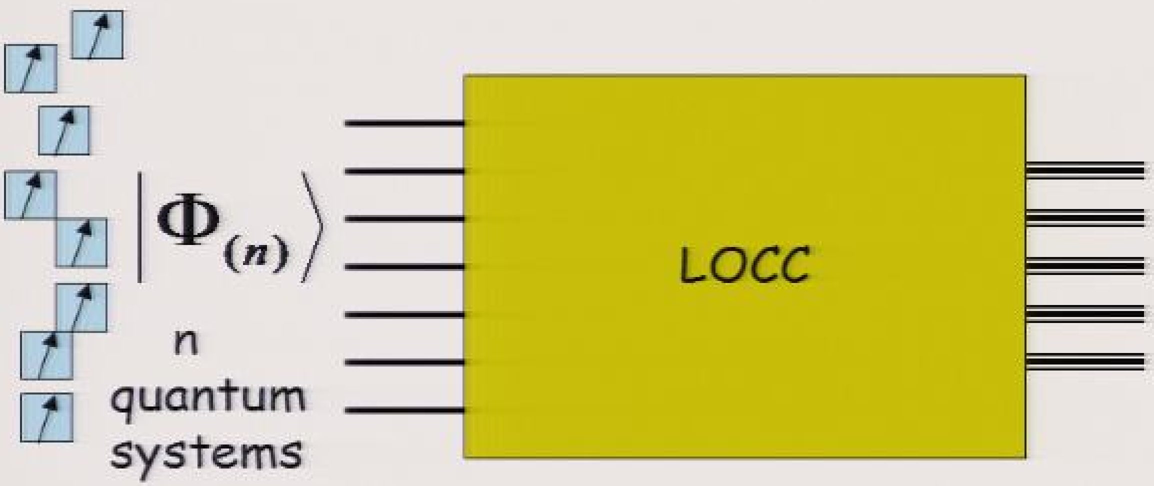


$\left|\Phi_{(n)}\right\rangle$

n
quantum
systems

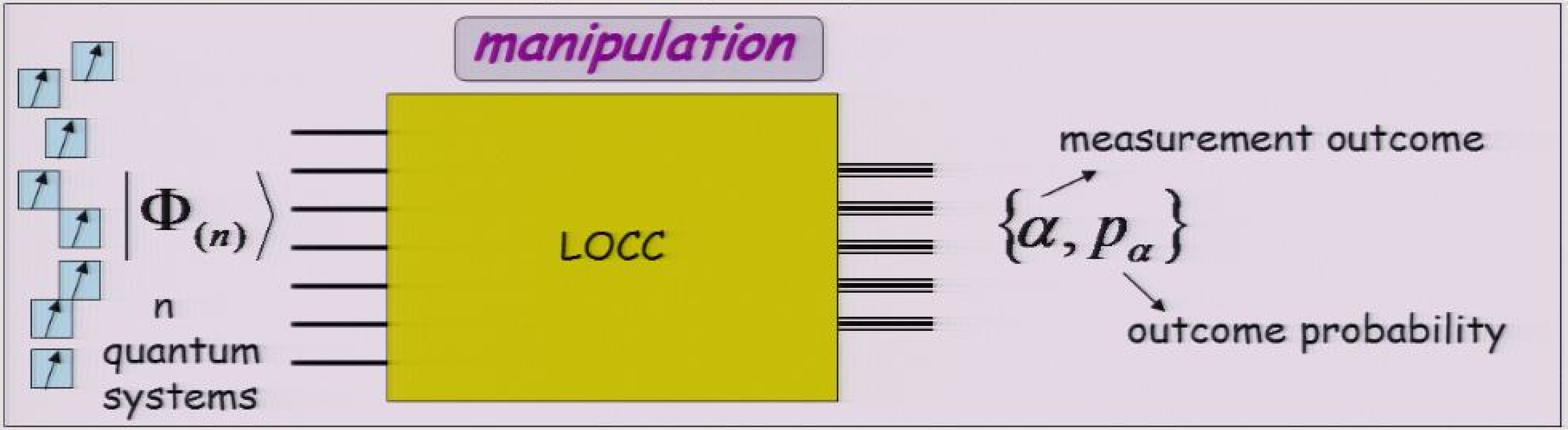
1 - Simulation of Quantum Systems

experimental setting: many-body state $\left|\Phi_{(n)}\right\rangle$



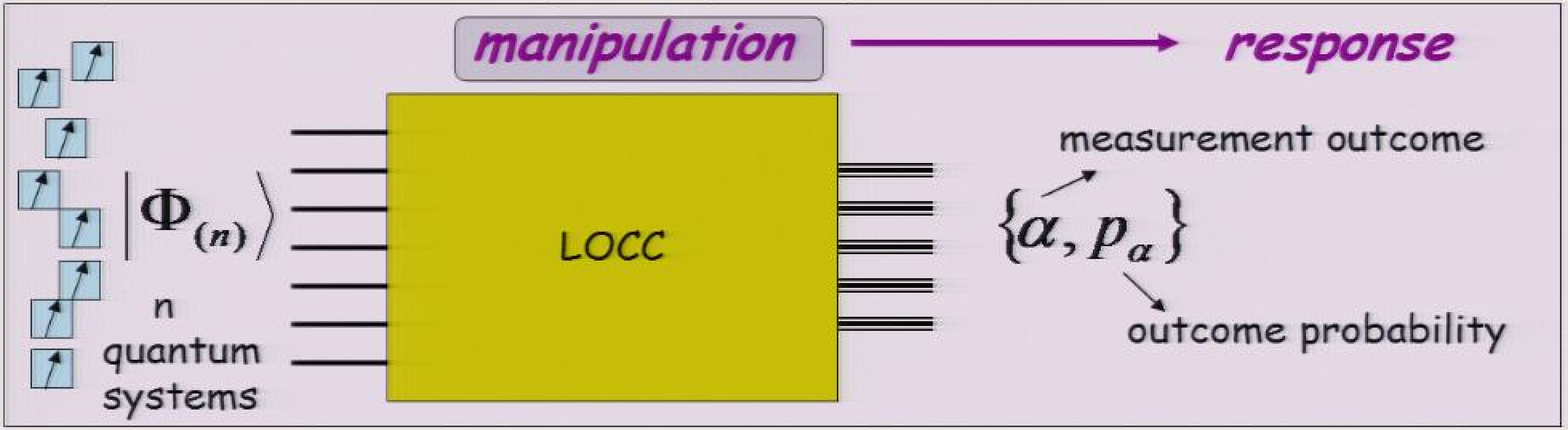
1 - Simulation of Quantum Systems

experimental setting: many-body state $\left|\Phi_{(n)}\right\rangle$



1 - Simulation of Quantum Systems

experimental setting: many-body state $\left|\Phi_{(n)}\right\rangle$

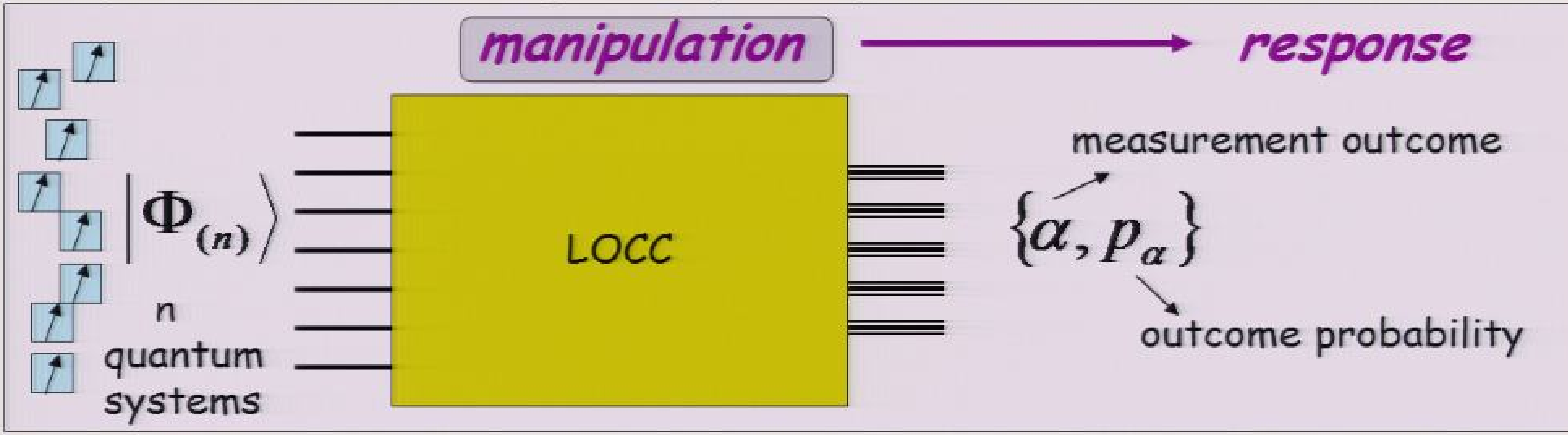


$$\{\alpha, p_\alpha\}$$

classical simulation of $\left|\Phi_{(n)}\right\rangle$

1 - Simulation of Quantum Systems

experimental setting: many-body state $\left|\Phi_{(n)}\right\rangle$

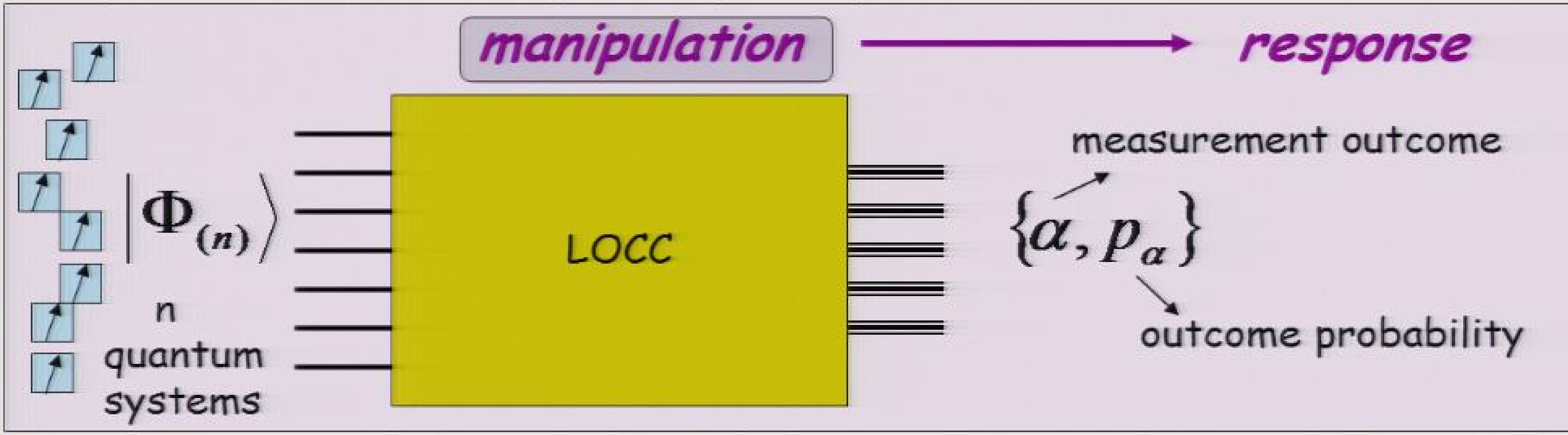


classical simulation of $\left|\Phi_{(n)}\right\rangle$

specification of $\left|\Phi_{(n)}\right\rangle$

1 - Simulation of Quantum Systems

experimental setting: many-body state $|\Phi_{(n)}\rangle$



classical simulation of $|\Phi_{(n)}\rangle$

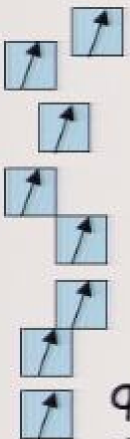
specification of $|\Phi_{(n)}\rangle$ + *classical computation*

experimental setting: many-body evolution

$$\left|\Phi_{(n)}^{final}\right\rangle = U\left|\Phi_{(n)}^{initial}\right\rangle$$

experimental setting: many-body evolution

$$\left|\Phi_{(n)}^{final}\right\rangle = U\left|\Phi_{(n)}^{initial}\right\rangle$$

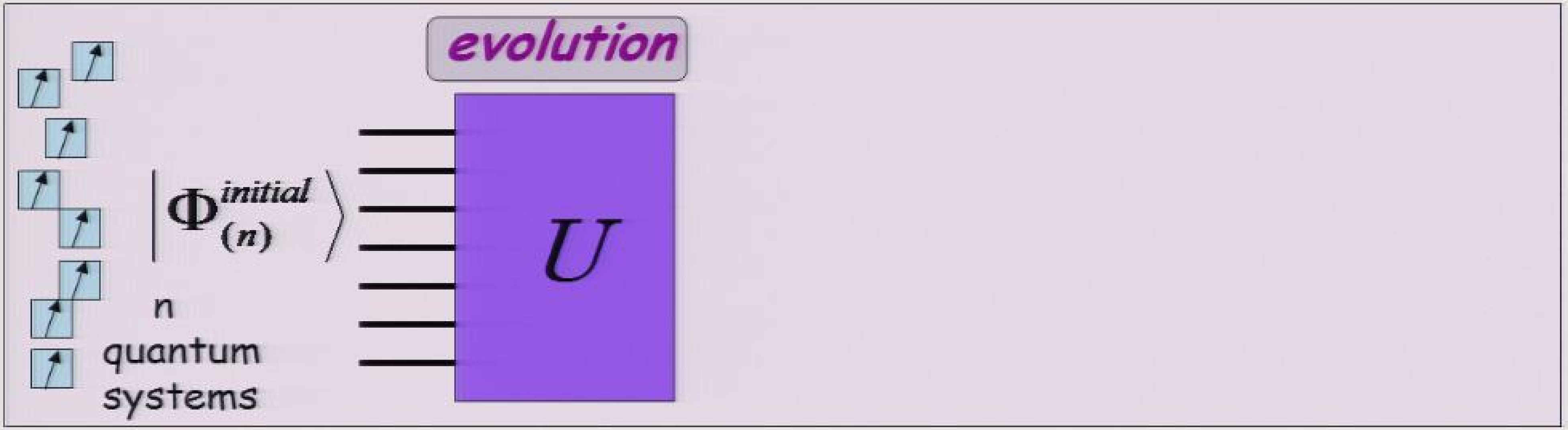


$$\left|\Phi_{(n)}^{initial}\right\rangle$$

n
quantum
systems

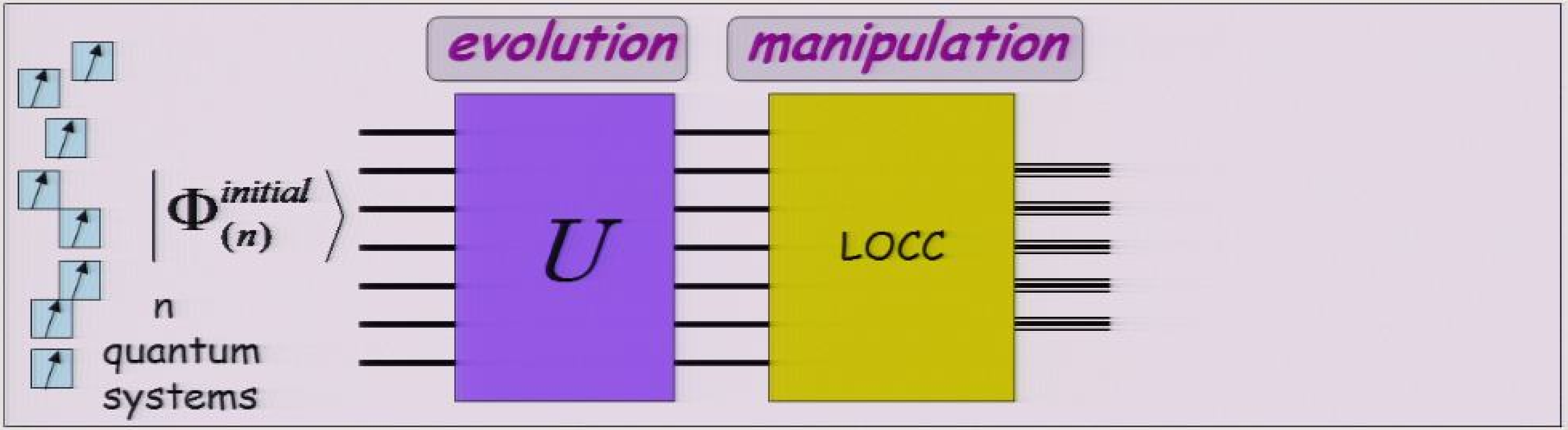
experimental setting: many-body evolution

$$\left|\Phi_{(n)}^{final}\right\rangle = U\left|\Phi_{(n)}^{initial}\right\rangle$$



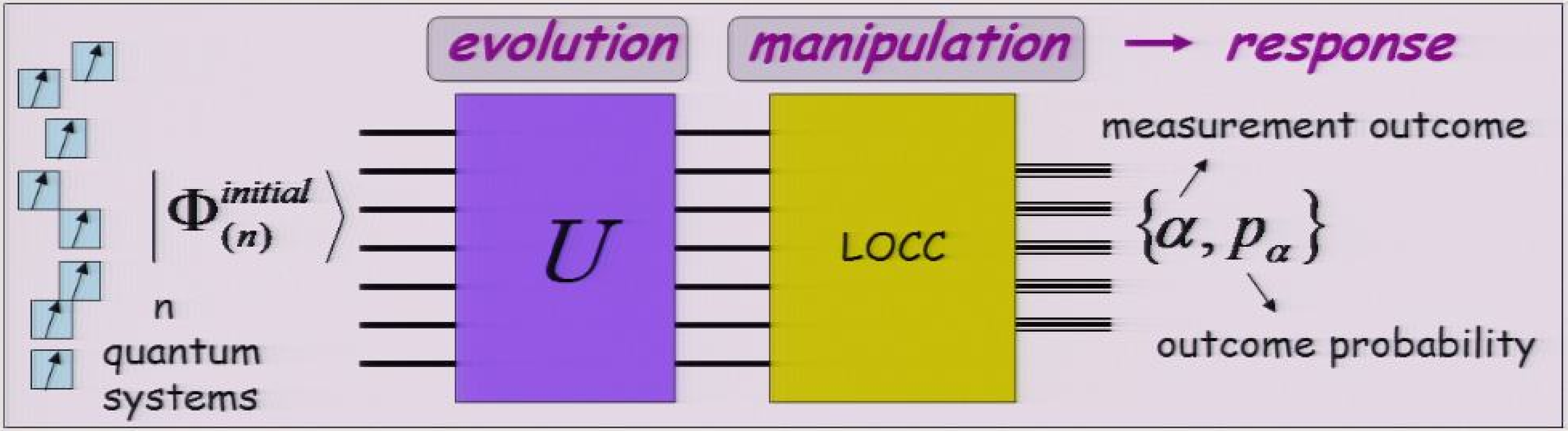
experimental setting: many-body evolution

$$\left|\Phi_{(n)}^{final}\right\rangle = U\left|\Phi_{(n)}^{initial}\right\rangle$$



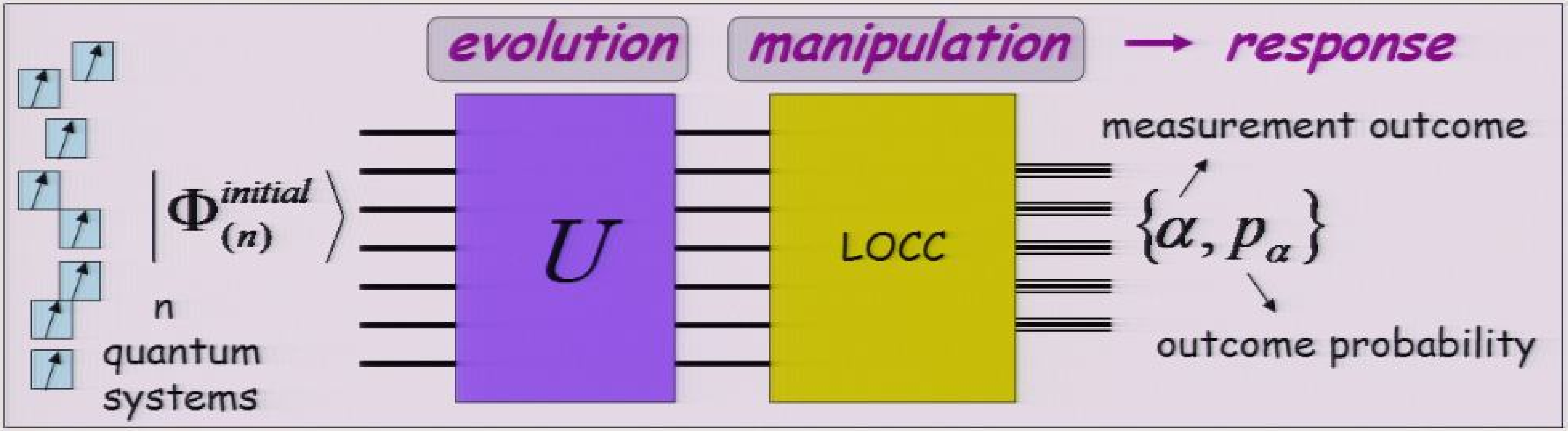
experimental setting: many-body evolution

$$\left|\Phi_{(n)}^{final}\right\rangle = U\left|\Phi_{(n)}^{initial}\right\rangle$$



experimental setting: many-body evolution

$$\left|\Phi_{(n)}^{final}\right\rangle = U \left|\Phi_{(n)}^{initial}\right\rangle$$



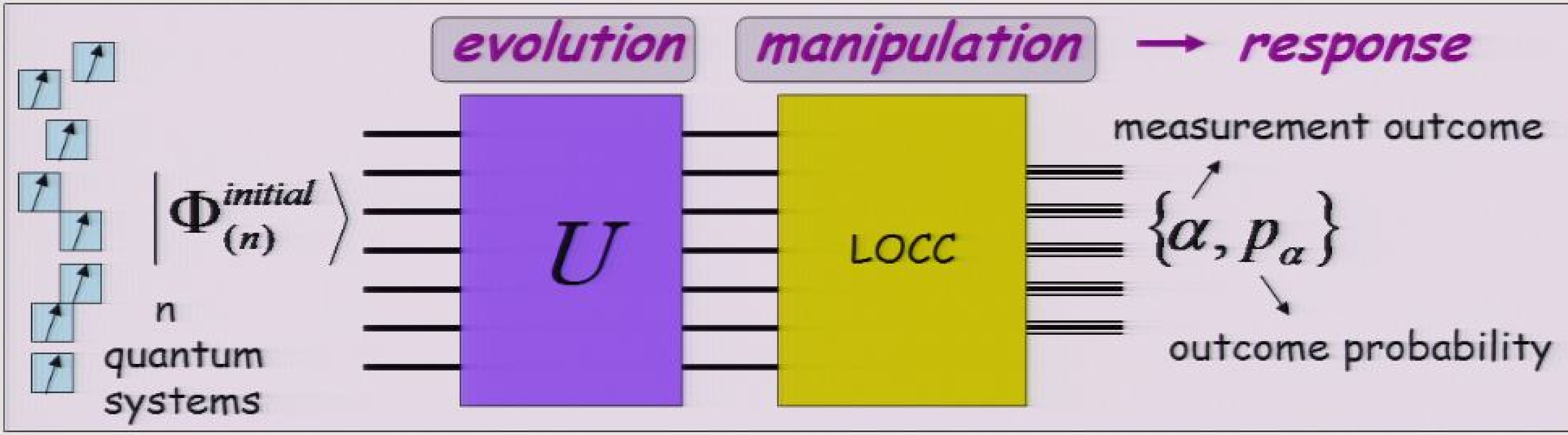
classical simulation of evolution U starting from $\left|\Phi_{(n)}^{initial}\right\rangle$

specification of

$\left|\Phi_{(n)}^{initial}\right\rangle$, U

experimental setting: many-body evolution

$$\left|\Phi^{final}_{(n)}\right\rangle = U\left|\Phi^{initial}_{(n)}\right\rangle$$



classical simulation of evolution U starting from $\left|\Phi^{initial}_{(n)}\right\rangle$

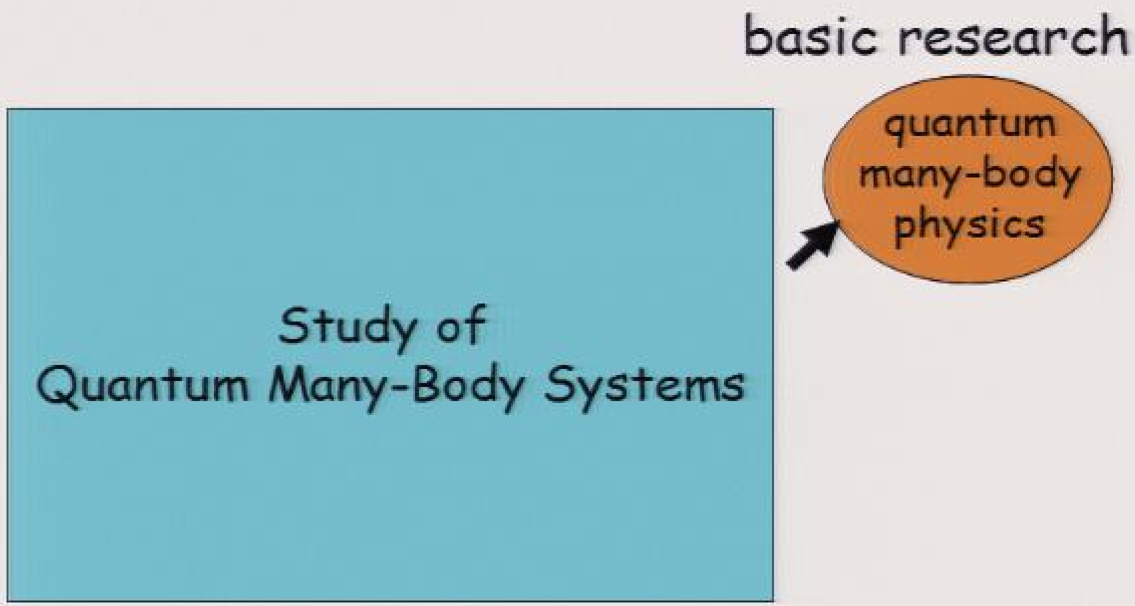
specification of $\left|\Phi^{initial}_{(n)}\right\rangle$, U + ***classical computation***

- Why do we want to simulate quantum systems ?

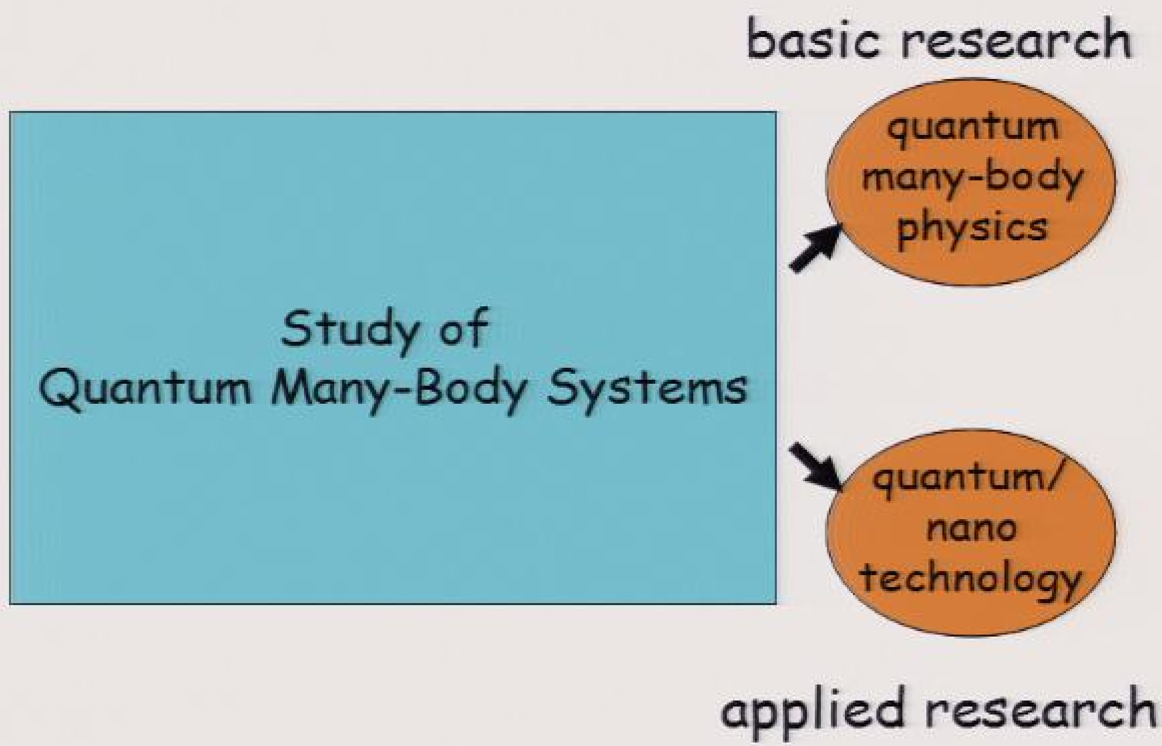
- Why do we want to simulate quantum systems ?

Study of
Quantum Many-Body Systems

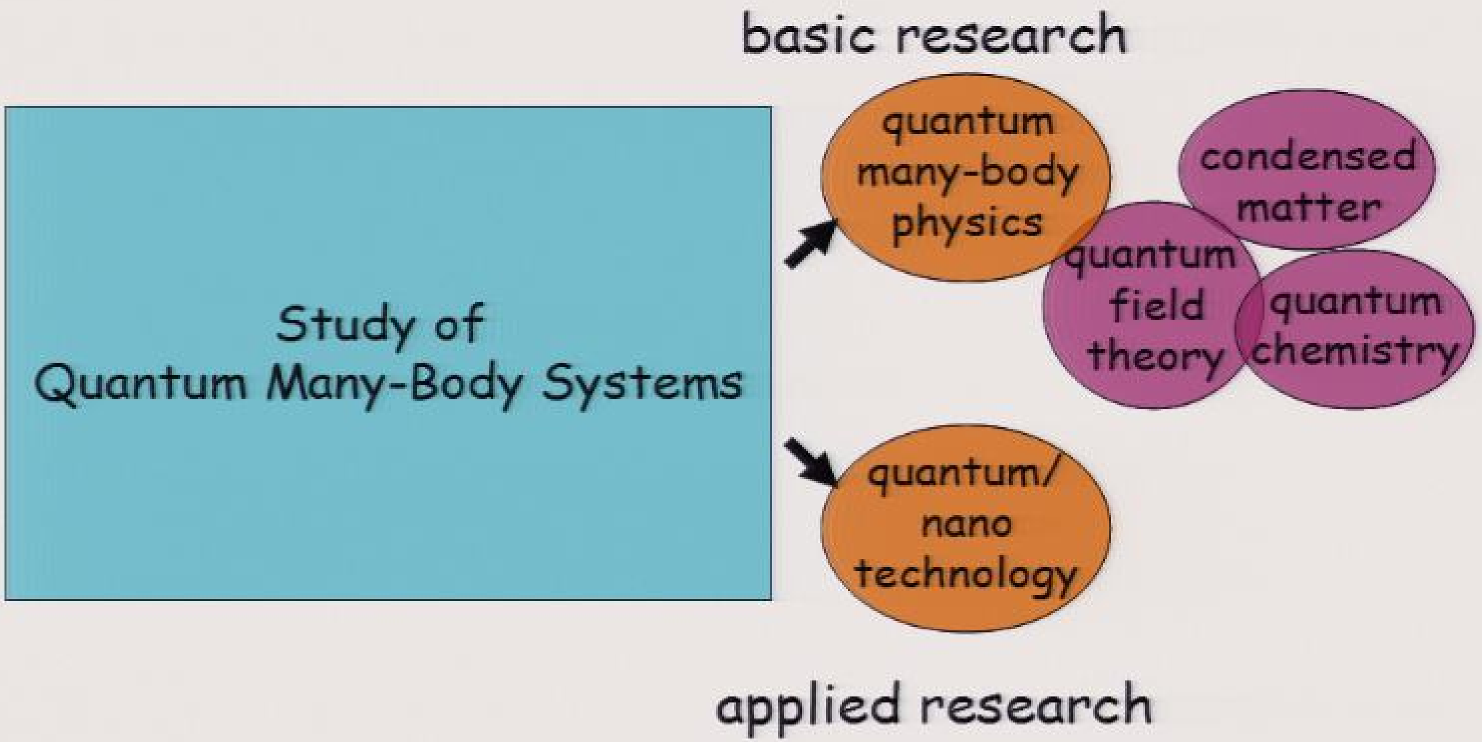
- Why do we want to simulate quantum systems ?



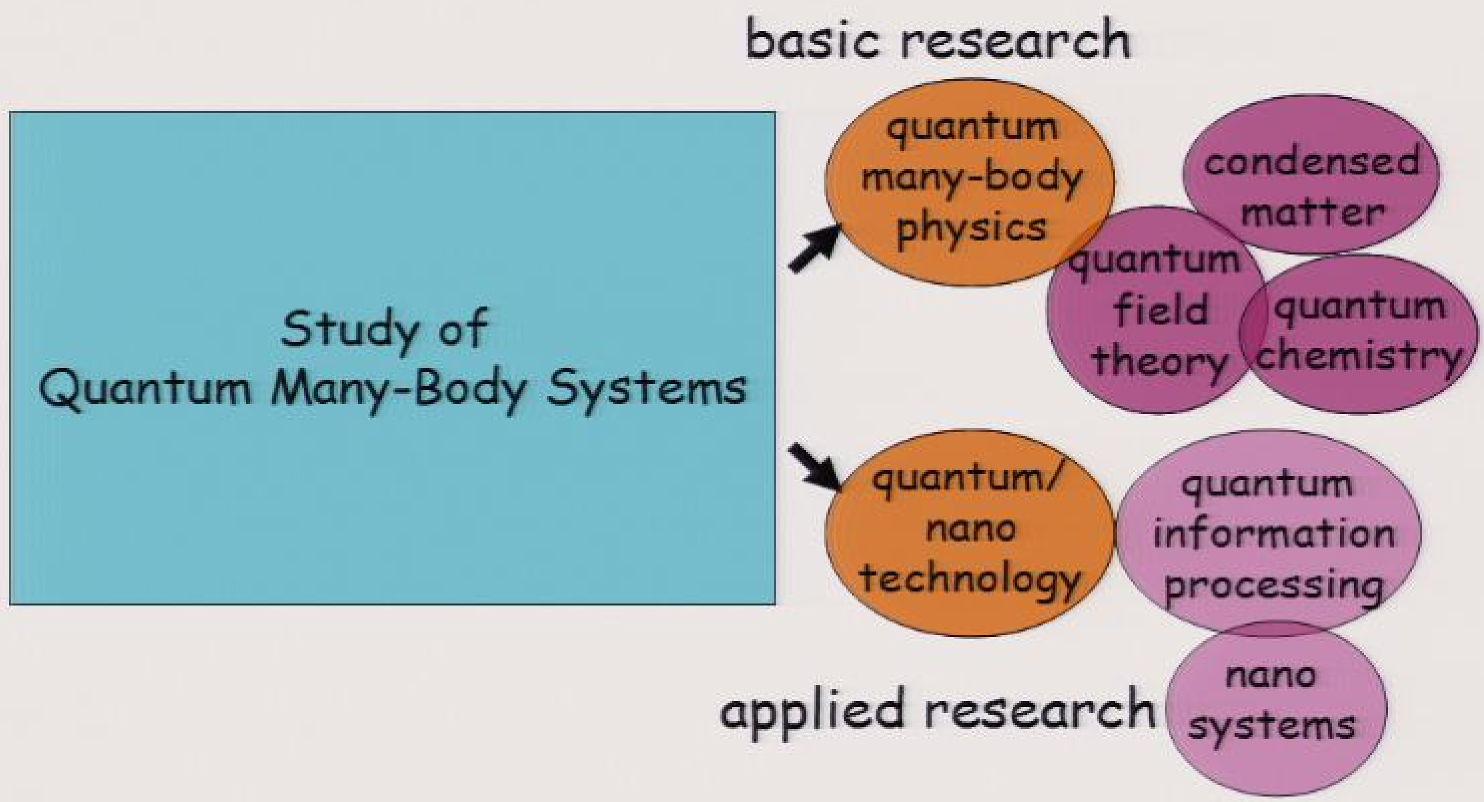
- Why do we want to simulate quantum systems ?



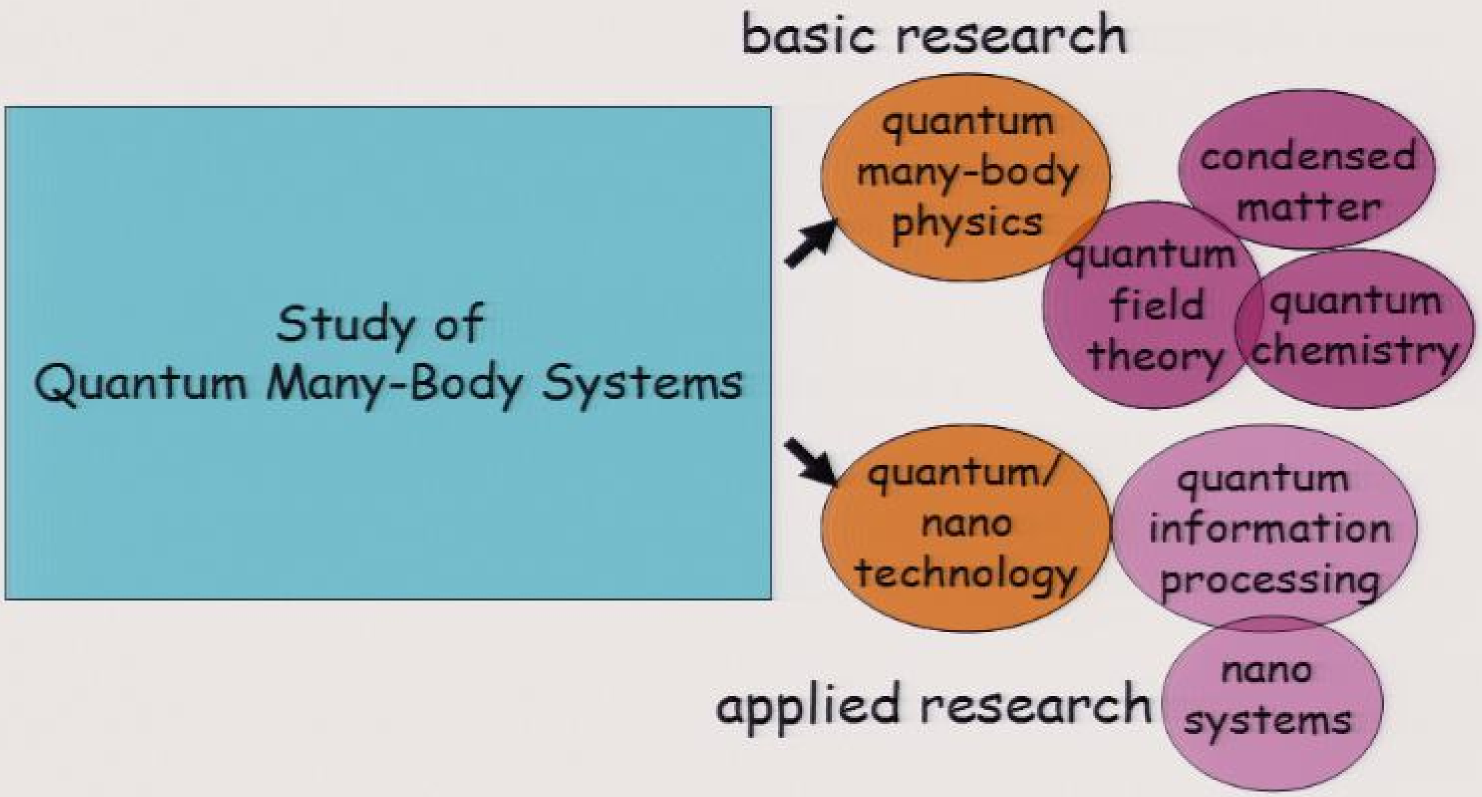
- Why do we want to simulate quantum systems ?



- Why do we want to simulate quantum systems ?



- Why do we want to simulate quantum systems ?



Some simulation algorithms are known [quantum montecarlo, density functional theory, density matrix renormalization group, …] but they only apply to very specific situations.

straightforward classical simulation

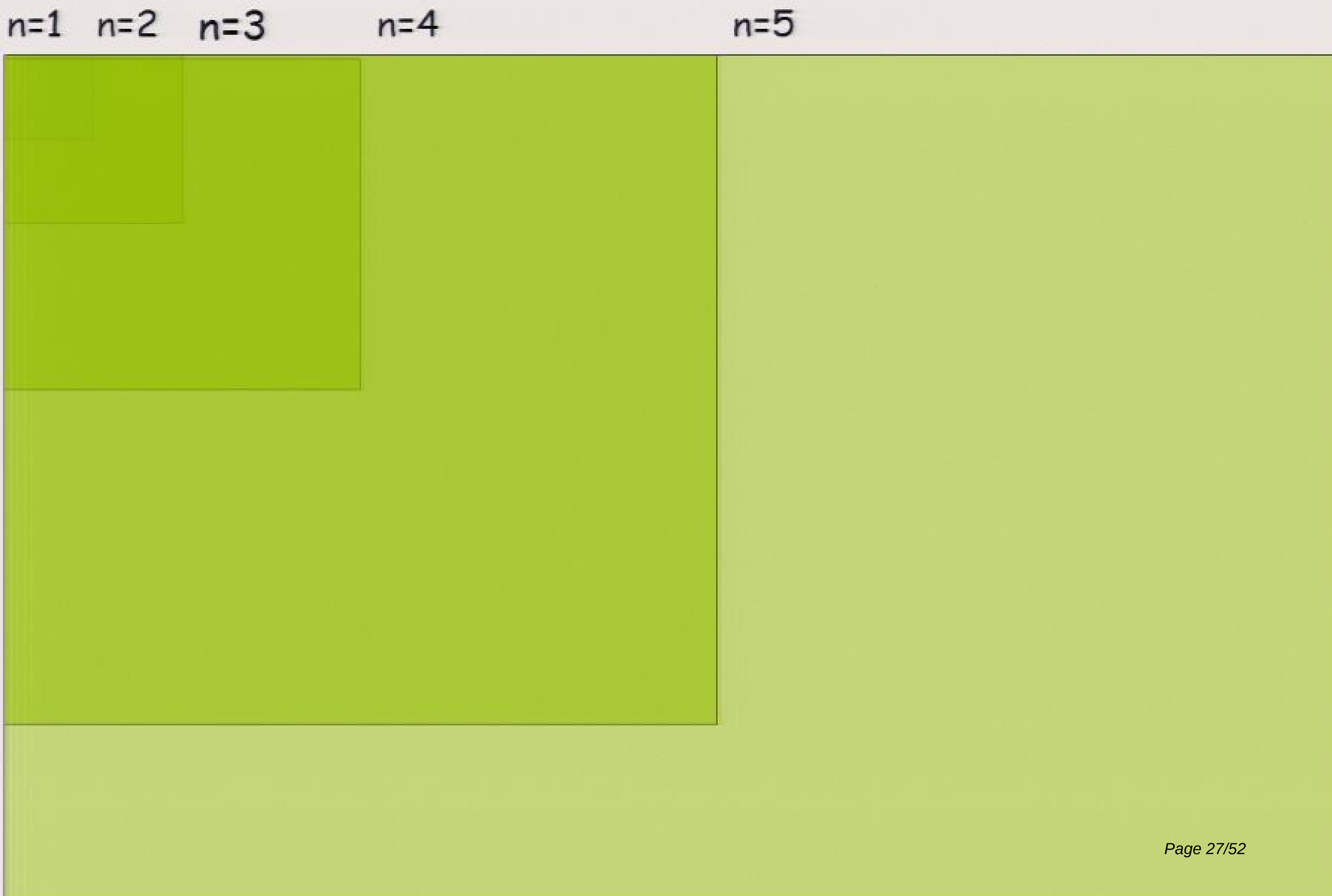
$$\dim H_n = 2^n$$

dimension of Hilbert space is exponentially large in n

straightforward
classical simulation

$$\dim H_n = 2^n$$

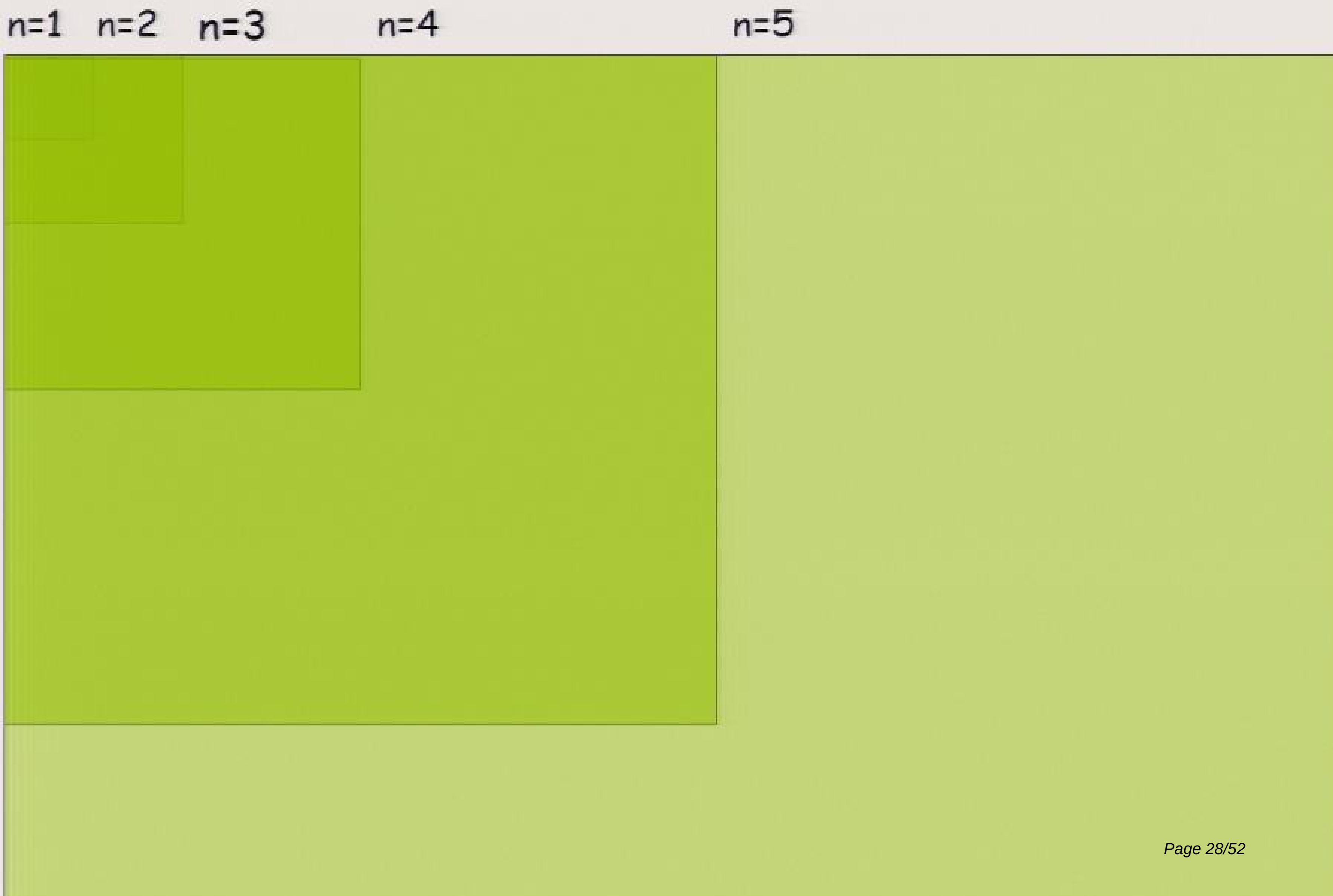
dimension of Hilbert space is exponentially large in n



straightforward classical simulation

$$\dim H_n = 2^n$$

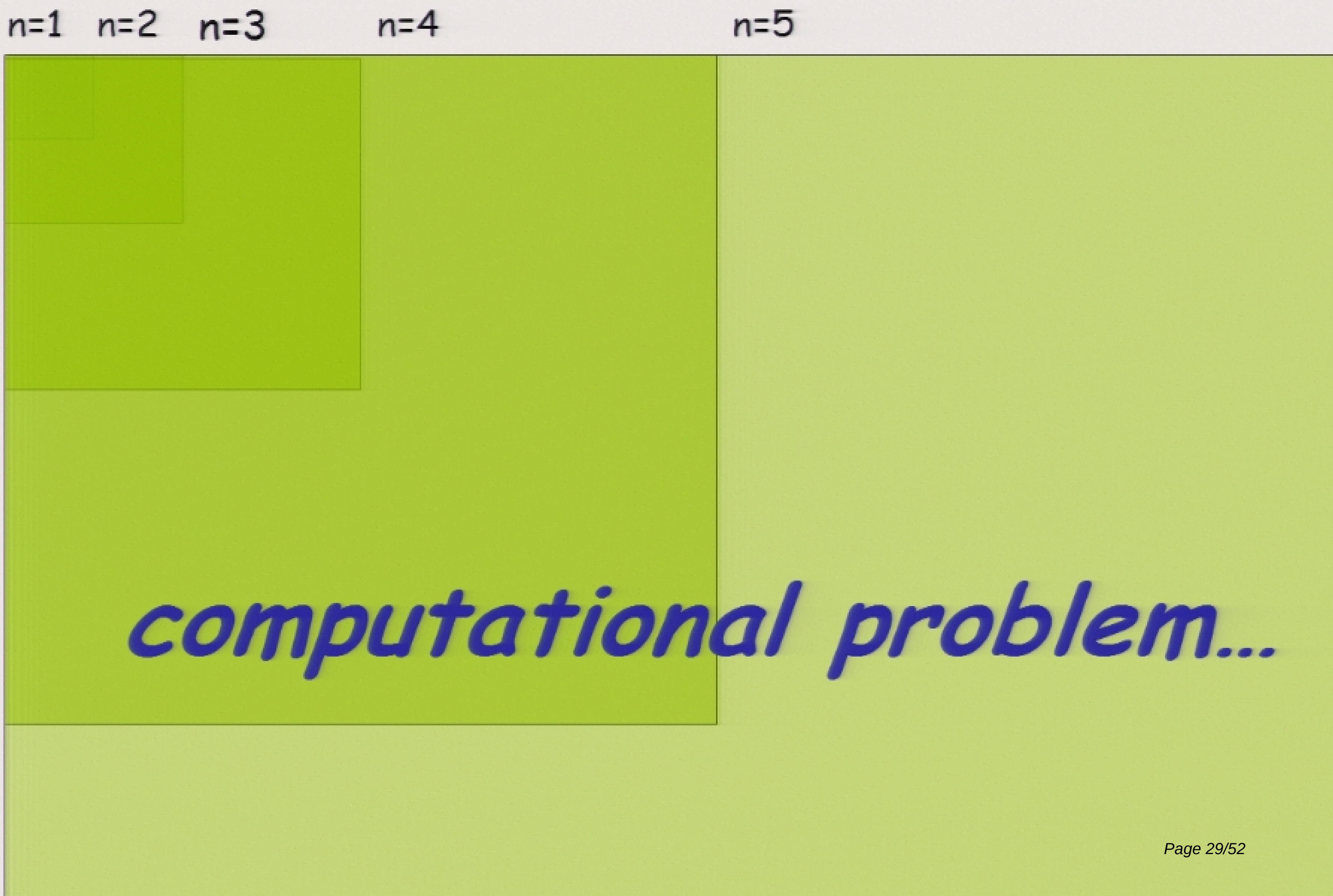
dimension of Hilbert space is exponentially large in n



straightforward classical simulation

$$\dim H_n = 2^n$$

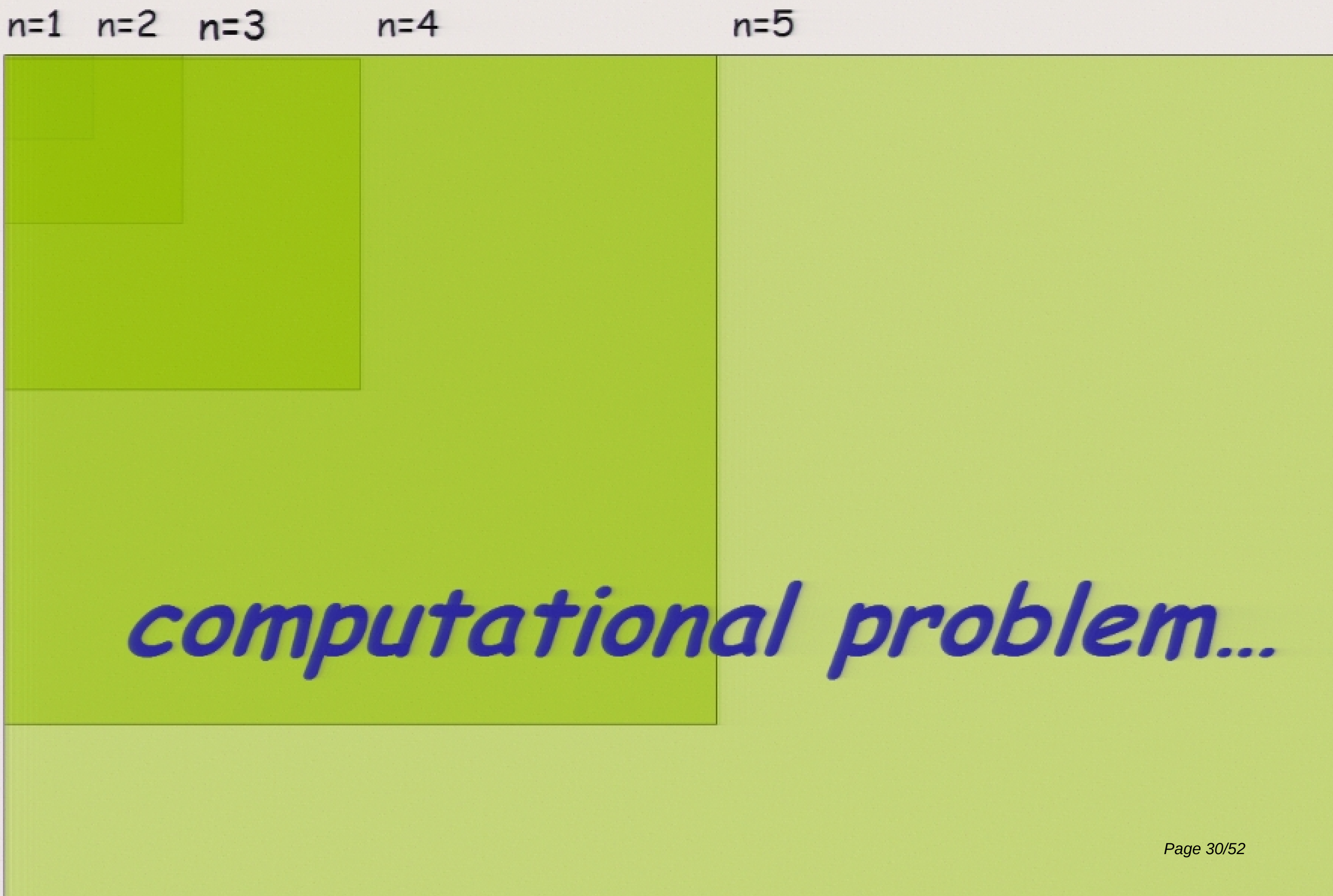
dimension of Hilbert space is exponentially large in *n*



straightforward classical simulation

$$\dim H_n = 2^n$$

dimension of Hilbert space is exponentially large in n

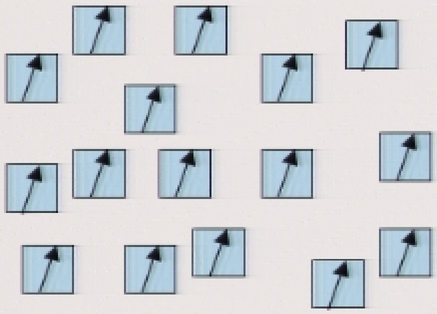


computational problem...

2 - Efficient Decomposition vs Efficient Simulation

2 - Efficient Decomposition vs Efficient Simulation

2 - Efficient Decomposition vs Efficient Simulation



n quantum systems

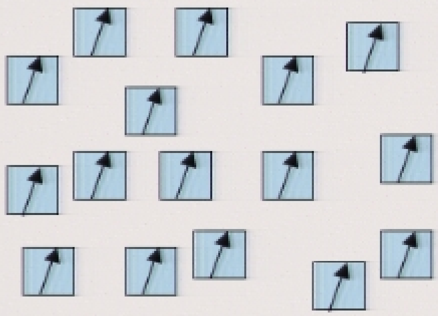
generic state

$$\left|\Phi_{(n)}\right\rangle = \sum_{i_1 \cdots i_n} C_{i_1 \cdots i_n} \left|i_1\right\rangle \cdots \left|i_n\right\rangle$$

2^n

parameters

2 - Efficient Decomposition vs Efficient Simulation



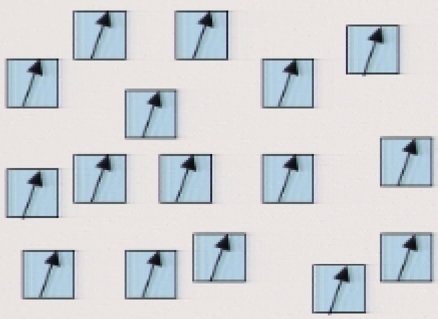
n quantum systems

generic state

$$\left|\Phi_{(n)}\right\rangle = \sum_{i_1\cdots i_n} C_{i_1\cdots i_n} \left|i_1\right\rangle \cdots \left|i_n\right\rangle$$

2^n parameters

2 - Efficient Decomposition vs Efficient Simulation

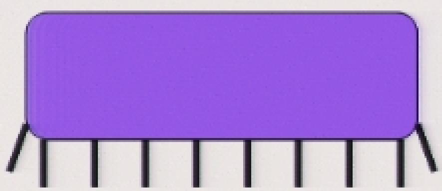


n quantum systems

generic state

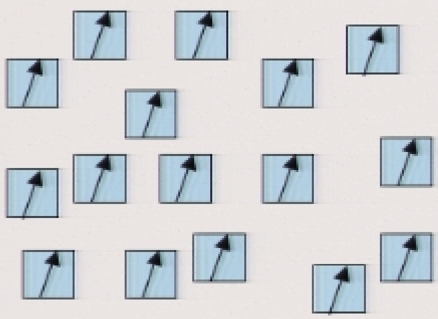
$$\left|\Phi_{(n)}\right\rangle = \sum_{i_1 \cdots i_n} C_{i_1 \cdots i_n} \left|i_1\right\rangle \cdots \left|i_n\right\rangle$$

2^n parameters



2^n parameters

2 - Efficient Decomposition vs Efficient Simulation

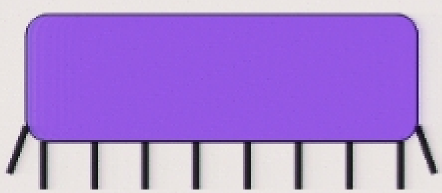


n quantum systems

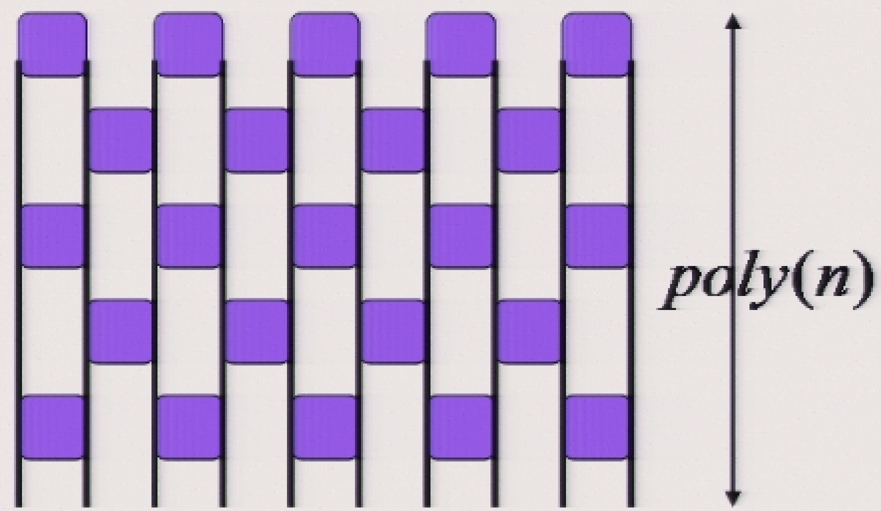
generic state

$$\left|\Phi_{(n)}\right\rangle = \sum_{i_1 \cdots i_n} C_{i_1 \cdots i_n} \left|i_1\right\rangle \cdots \left|i_n\right\rangle$$

2^n parameters

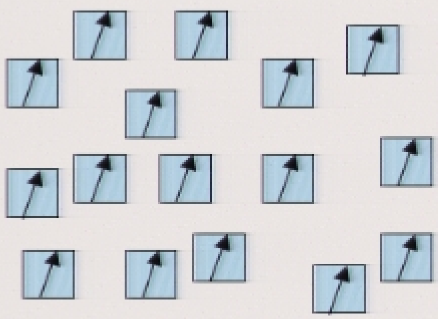


2^n parameters



poly(n) parameters

2 - Efficient Decomposition vs Efficient Simulation

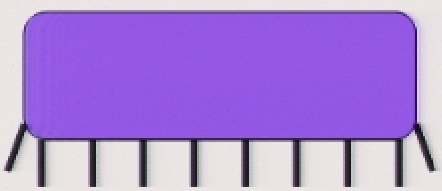


n quantum systems

generic state

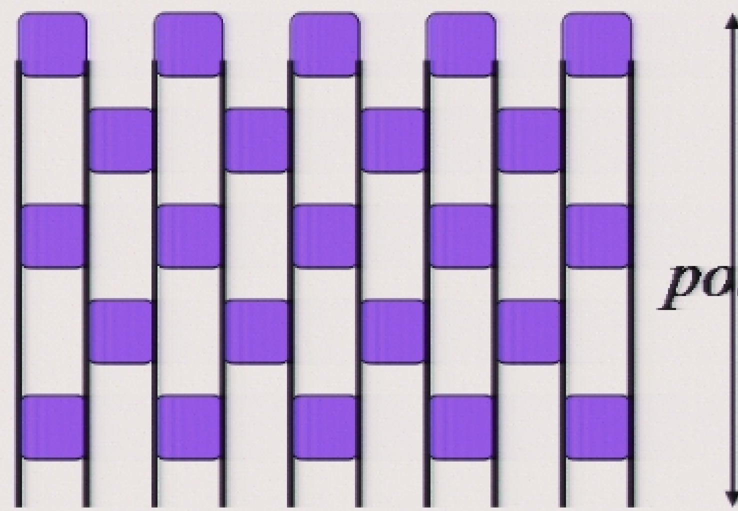
$$\left|\Phi_{(n)}\right\rangle = \sum_{i_1 \cdots i_n} C_{i_1 \cdots i_n} \left|i_1\right\rangle \cdots \left|i_n\right\rangle$$

2^n parameters



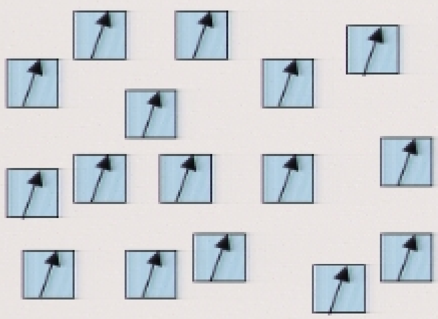
2^n parameters

inefficient decomposition



poly(n) parameters

2 - Efficient Decomposition vs Efficient Simulation

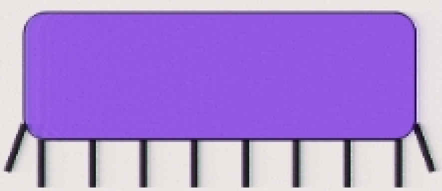


n quantum systems

generic state

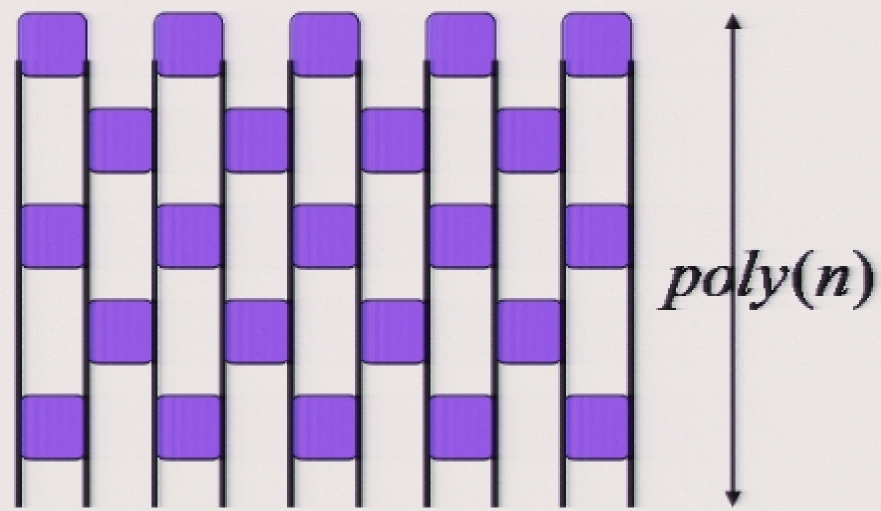
$$\left|\Phi_{(n)}\right\rangle = \sum_{i_1\cdots i_n} C_{i_1\cdots i_n}\left|i_1\right\rangle\cdots\left|i_n\right\rangle$$

2^n parameters



2^n parameters

inefficient decomposition



poly(n) parameters

efficient decomposition

Efficient decomposition

Efficient decomposition $\not\Longrightarrow$ Efficient simulation

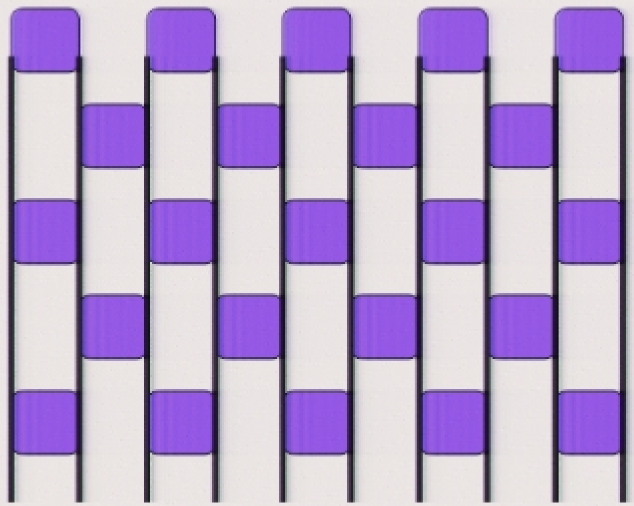
Computation of a scalar product:

$$\left\langle \Phi_{(n)} \middle| \Phi_{(n)} \right\rangle =$$

Efficient decomposition $\not\Rightarrow$ Efficient simulation

Computation of a scalar product:

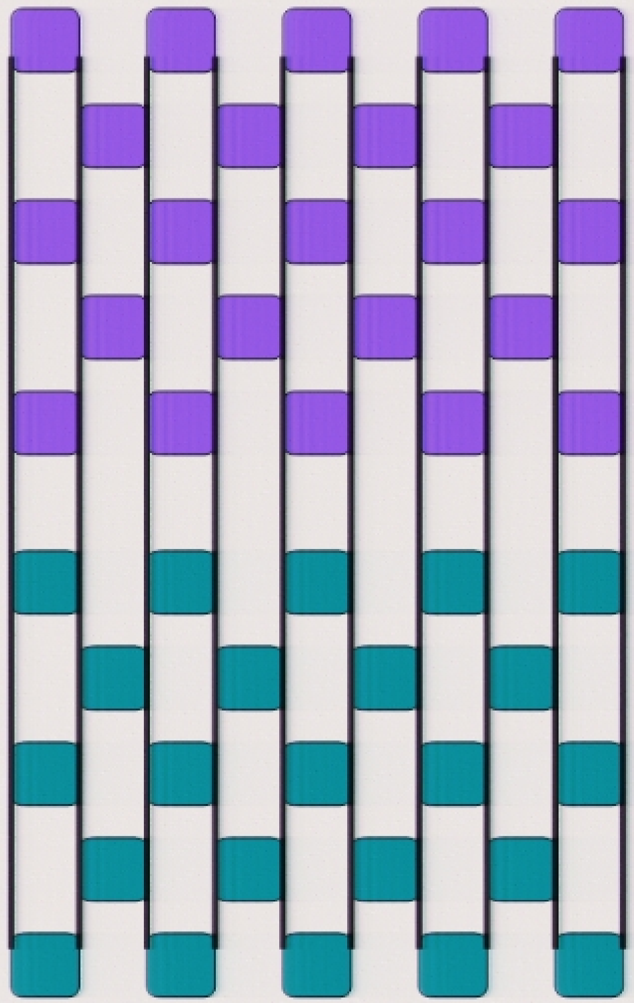
$$\left\langle \Phi_{(n)} \middle| \Phi_{(n)} \right\rangle =$$



Efficient decomposition $\not\Rightarrow$ Efficient simulation

Computation of a scalar product:

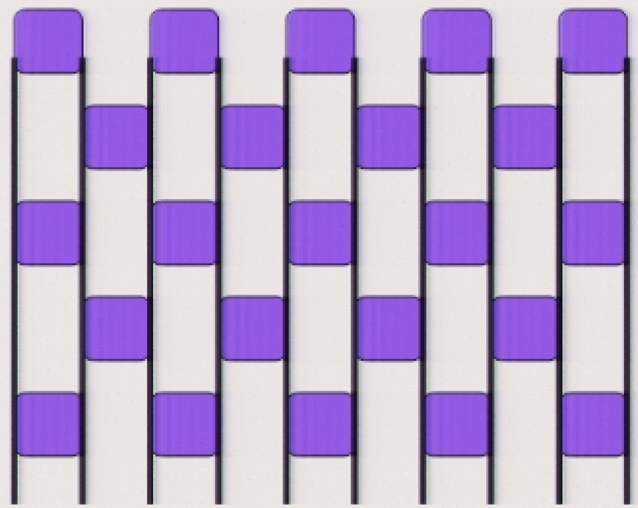
$$\left\langle \Phi_{(n)} \middle| \Phi_{(n)} \right\rangle =$$



Efficient decomposition $\not\longrightarrow$ Efficient simulation

Computation of a scalar product:

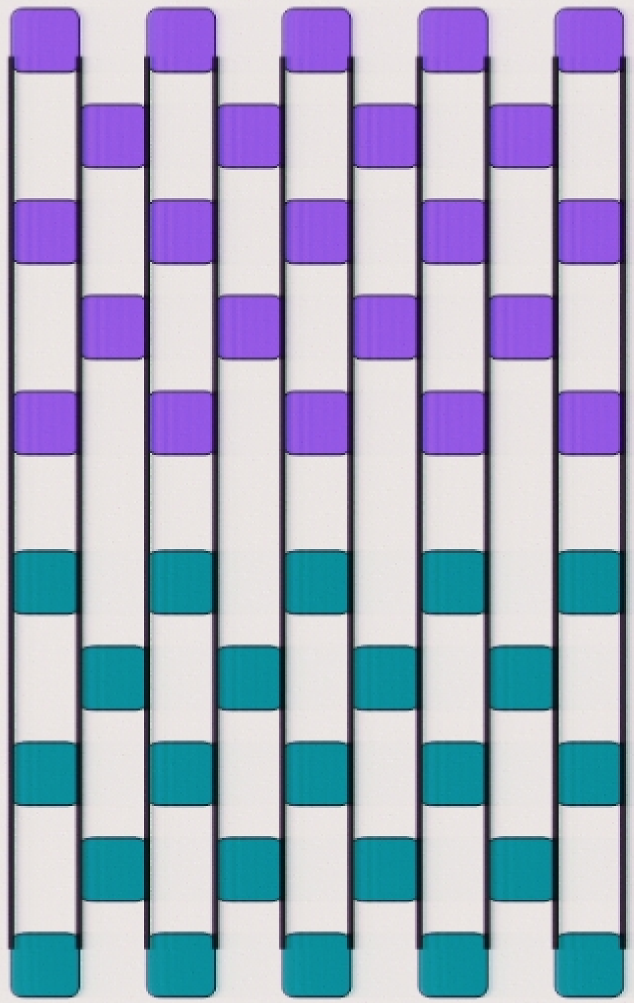
$$\left\langle \Phi_{(n)} \middle| \Phi_{(n)} \right\rangle =$$



Efficient decomposition ⇸ Efficient simulation

Computation of a scalar product:

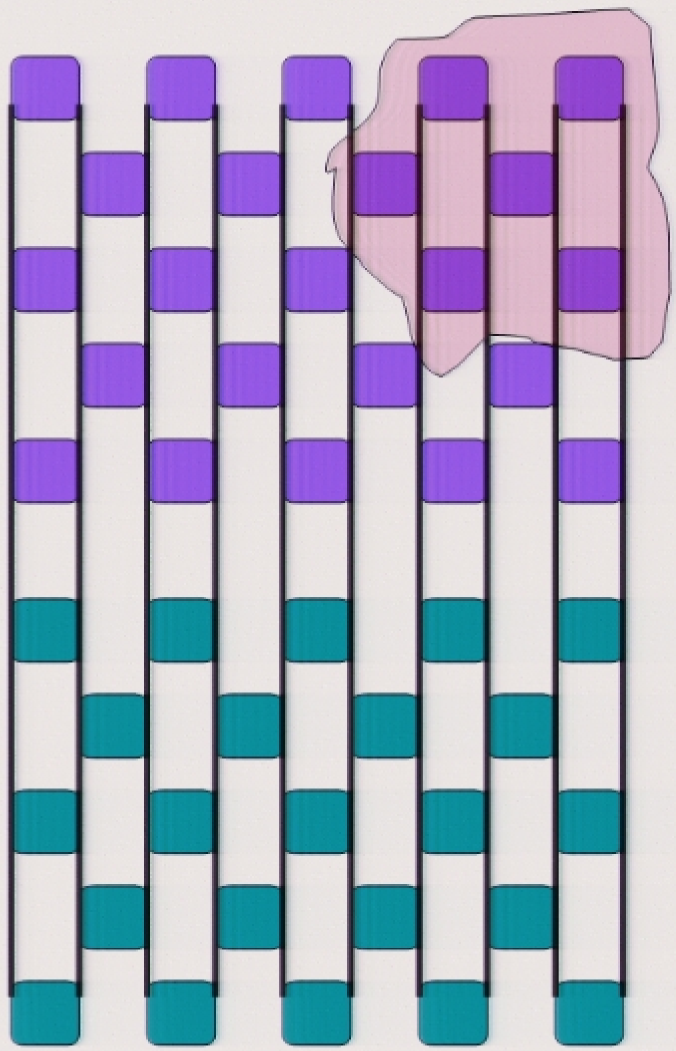
$$\left\langle \Phi_{(n)} \middle| \Phi_{(n)} \right\rangle =$$

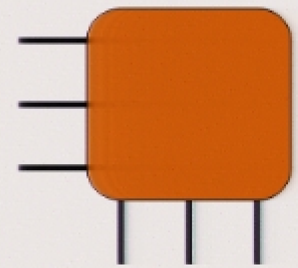


Efficient decomposition ⇏ Efficient simulation

Computation of a scalar product:

$$\left\langle \Phi_{(n)} \middle| \Phi_{(n)} \right\rangle =$$

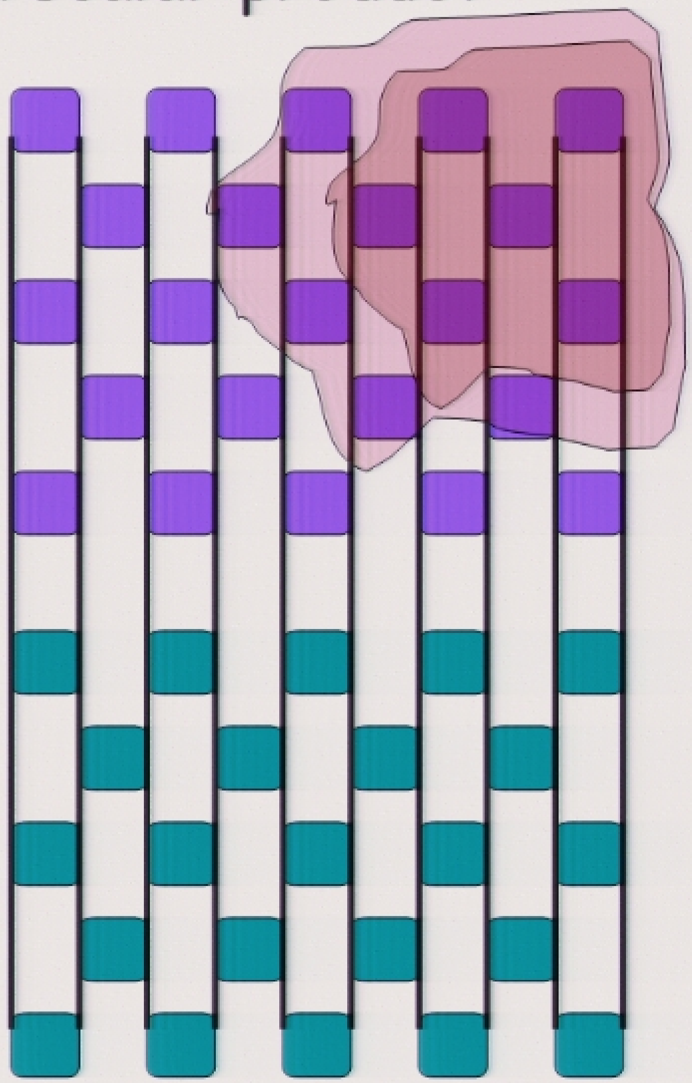


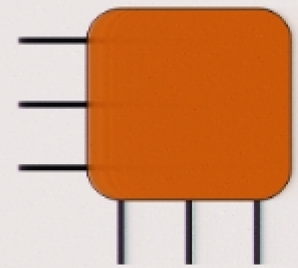


Efficient decomposition ⇏ Efficient simulation

Computation of a scalar product:

$$\left\langle \Phi_{(n)} \middle| \Phi_{(n)} \right\rangle =$$

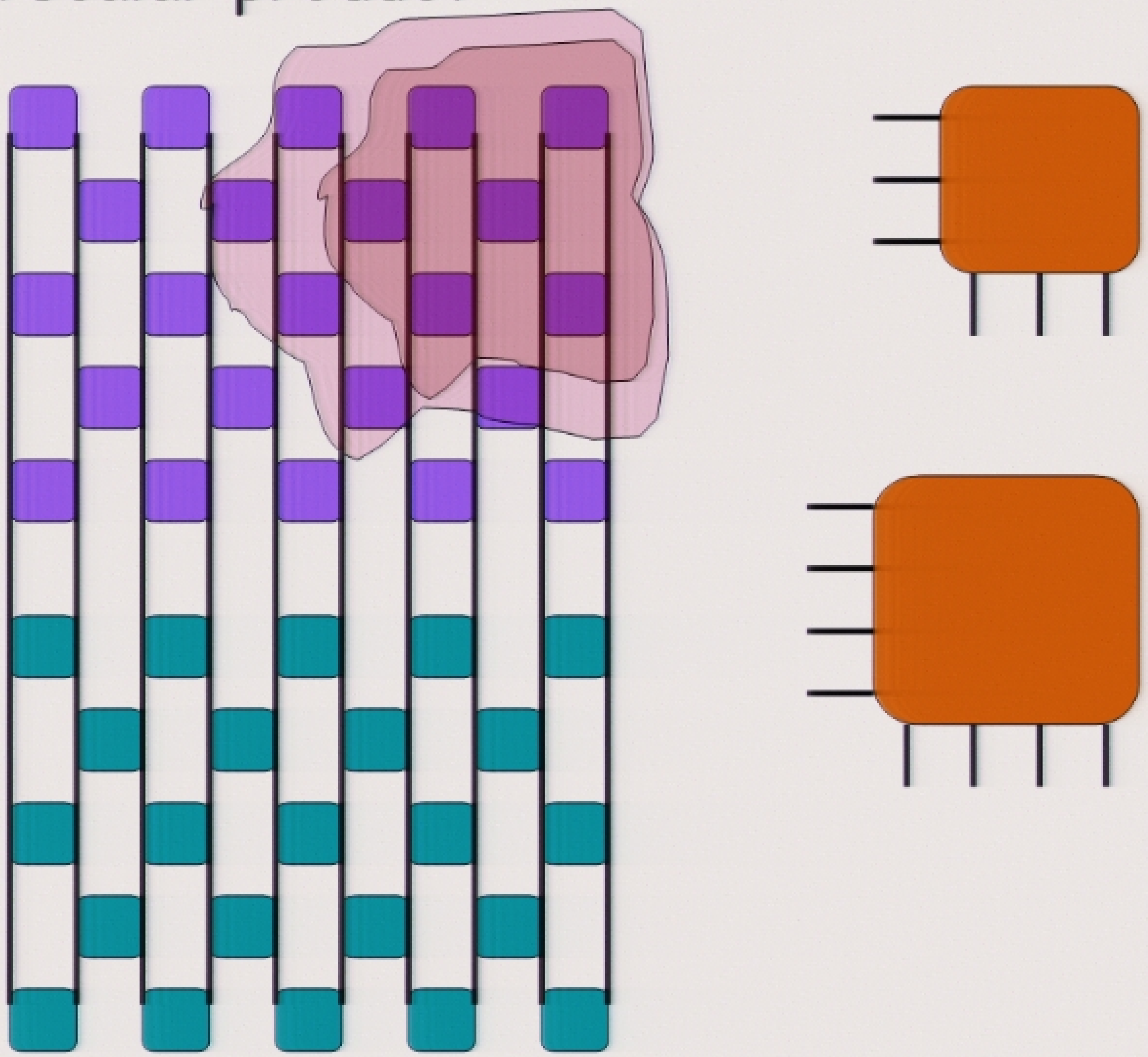




Efficient decomposition $\not\Rightarrow$ Efficient simulation

Computation of a scalar product:

$$\left\langle \Phi_{(n)} \middle| \Phi_{(n)} \right\rangle =$$



Efficient decomposition

and

Efficient simulation

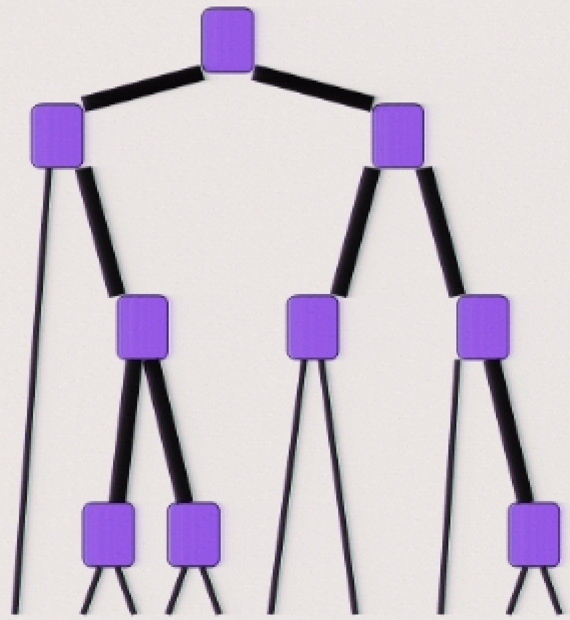
Efficient decomposition

and

Efficient simulation

Example:

Tree of small tensors

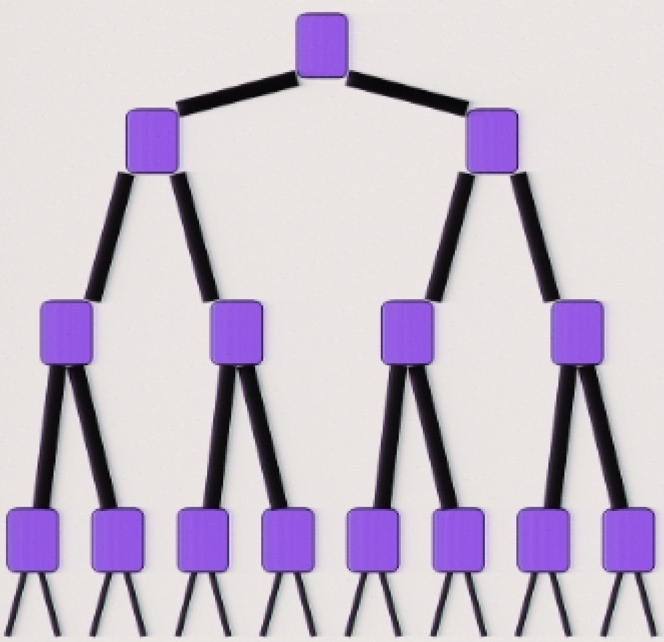


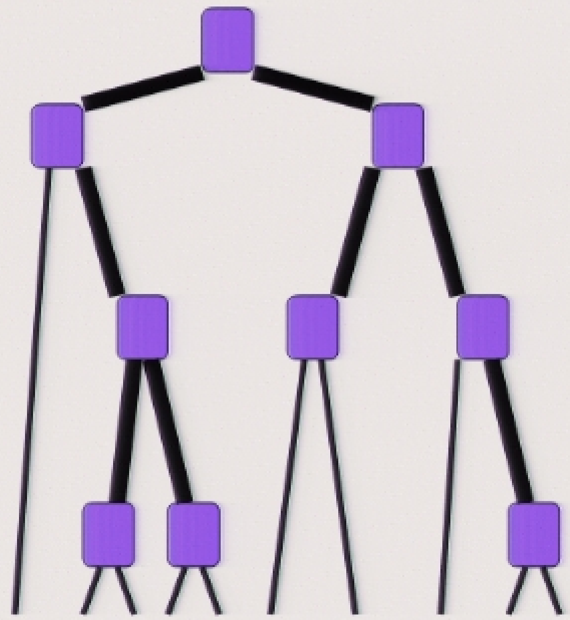
Efficient decomposition

and

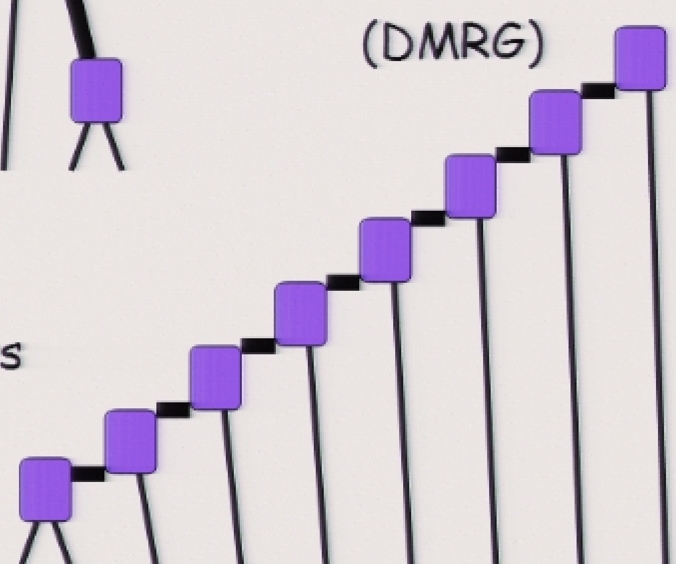
Efficient simulation

Example:
Tree of small tensors





$O(n)$ parameters

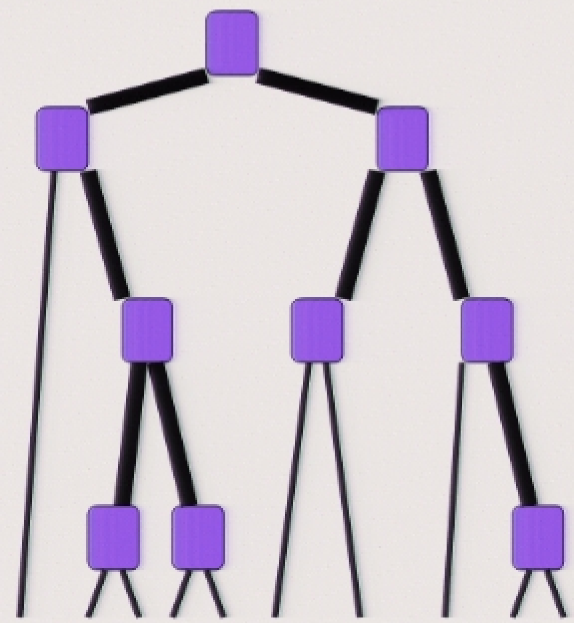


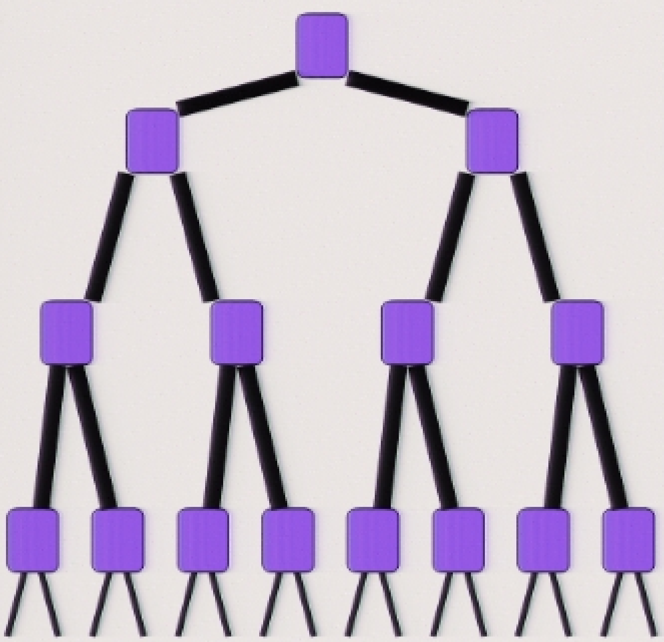
Efficient decomposition

and

Efficient simulation

Example:
Tree of small tensors





$O(n)$ parameters

(DMRG)