Quantum Recursion and Second Quantisation

Mingsheng Ying

State Key Laboratory of Computer Science

Outline

Outline

IBM, Google, Intel, Microsoft building quantum computers

- IBM Q: 5 quantum bits (qubits)

IBM, Google, Intel, Microsoft building quantum computers

- IBM Q: 5 quantum bits (qubits)
- Google: quantum supremacy

Will you be quantum Alan Turing?

Model of quantum computation — Quantum Turing machine

[1] P. Benioff, The computer as a physical system: A microscopic quantum mechanical Hamiltonian model of computers as represented by Turing machines, *J. of Statistical Physics* 1980.
[2] I. Yu. Manin, *Computable and Noncomputable* (in Russian), Sov. Radio 1980.
[3] R. Feynman, Simulating physics with computers, *Int. J. of Theoretical Physics* 1982.
[4] D. Deutsch, Quantum theory, the Church-Turing principle and the universal quantum computer, *Proc. of the Royal Society of London A* 1985.

Outline

Mathematical Logic: Recursion

A long history in Mathematics!

Mathematical Logic: Recursion

A long history in Mathematics!

Recursive programming

Put forward and implement the recursive procedure as an ALGOL60 language construct

[5] E. W. Dijkstra, Recursive programming, *Numerische Mathematik* 1960.
[6] E. G. Daylight, Dijkstra's rallying cry for generalization: The advent of the recursive procedure, late 1950s - early 1960s, *The Computer J.* 2011.

Outline

Quantum programming primitive: Loops

[7] E. Bernstein and U. Vazirani, Quantum complexity theory, *SIAM J. on Computing* 1997

Quantum programming primitive: Loops

[7] E. Bernstein and U. Vazirani, Quantum complexity theory, *SIAM J. on Computing* 1997

Recursion in quantum programming

Recursive procedure in quantum programming language QPL

[8] P. Selinger, Toward a quantum programming language, *Mathematical Structures in Computer Science* 2004.

von Neumann's Hilbert space formalism of quantum mechanics

- State space of quantum system: a Hilbert space $\mathcal{H}$

von Neumann's Hilbert space formalism of quantum mechanics

- State space of quantum system: a Hilbert space $\mathcal{H}$
- Quantum states: density operator — an operator in $\mathcal{H}$: ρ is positive; $tr(\rho) = 1$.

von Neumann's Hilbert space formalism of quantum mechanics

- State space of quantum system: a Hilbert space $\mathcal{H}$
- Quantum states: density operator — an operator in $\mathcal{H}$: ρ is positive; $tr(\rho) = 1$.
- Dynamics of quantum system:

von Neumann's Hilbert space formalism of quantum mechanics

- State space of quantum system: a Hilbert space $\mathcal{H}$
- Quantum states: density operator — an operator in $\mathcal{H}$: ρ is positive; $tr(\rho) = 1$.
- Dynamics of quantum system:
  - Continuous time — Schrödinger equation

von Neumann's Hilbert space formalism of quantum mechanics

- State space of quantum system: a Hilbert space $\mathcal{H}$
- Quantum states: density operator — an operator in $\mathcal{H}$: ρ is positive; $tr(\rho) = 1$.
- Dynamics of quantum system:
  - Continuous time — Schrödinger equation
  - Discrete time —
    unitary operators (closed system): $UU^\dagger = U^\dagger U = I$.
    super-operators (open system): Operator in the space of operators: completely positive; trace-preserving

Solutions to recursive equations of quantum programs

Theorem:

1. The set of density operators in $\mathcal{H}$ with the Löwner order is a CPO

[9] M. S. Ying, R. Y. Duan, Y. Feng, Z. F. Ji, Predicate transformer semantics of quantum programs, in: *Semantic Techniques in Quantum Computation*, Cambridge Univ. Press 2010.

Solutions to recursive equations of quantum programs

Theorem:

1. The set of density operators in $\mathcal{H}$ with the Löwner order is a CPO
2. The set of super-operators in $\mathcal{H}$ is a CPO.

[9] M. S. Ying, R. Y. Duan, Y. Feng, Z. F. Ji, Predicate transformer semantics of quantum programs, in: *Semantic Techniques in Quantum Computation*, Cambridge Univ. Press 2010.

Solutions to recursive equations of quantum programs

Theorem:

1. The set of density operators in $\mathcal{H}$ with the Löwner order is a CPO
2. The set of super-operators in $\mathcal{H}$ is a CPO.

- finite-dimensional $\mathcal{H}$: P. Selinger (2004)

[9] M. S. Ying, R. Y. Duan, Y. Feng, Z. F. Ji, Predicate transformer semantics of quantum programs, in: *Semantic Techniques in Quantum Computation*, Cambridge Univ. Press 2010.

Solutions to recursive equations of quantum programs

Theorem:

1. The set of density operators in $\mathcal{H}$ with the Löwner order is a CPO
2. The set of super-operators in $\mathcal{H}$ is a CPO.

- finite-dimensional $\mathcal{H}$: P. Selinger (2004)
- infinite-dimensional $\mathcal{H}$:

[9] M. S. Ying, R. Y. Duan, Y. Feng, Z. F. Ji, Predicate transformer semantics of quantum programs, in: *Semantic Techniques in Quantum Computation*, Cambridge Univ. Press 2010.

Outline

Selinger's slogan: "Quantum data, classical control"

Control flow is classical: branching is determined by the outcomes of quantum measurements.

Example: $M = \{M_0, M_1\}$ is a quantum measurement

$$\begin{array}{lcl} \textbf{if}\ M[q] = & 0 & \rightarrow P_0 \\ \square & 1 & \rightarrow P_1 \\ \textbf{fi} & & \end{array}$$

Selinger's slogan: "Quantum data, classical control"

Control flow is classical: branching is determined by the outcomes of quantum measurements.

Example: $M = \{M_0, M_1\}$ is a quantum measurement

$$\begin{array}{lll} \textbf{if}\ M[q] = & 0 \rightarrow P_0 \\ \square & 1 \rightarrow P_1 \\ \textbf{fi} & \end{array}$$

"Quantum data, quantum control"

Functional quantum programming language QML, its categorical semantics

[10] T. Altenkirch and J. Grattage, A functional quantum programming language, *LICS* 2005.

How to define quantum control?

[11] Y. Aharonov, J. Anandan, S. Popescu and L. Vaidman, Superpositions of time evolutions of a quantum system and quantum time-translation machine, *Plysical Review Letters* 1990.
[12] A. Ambainis, E. Bach, A. Nayak, A. Vishwanath and J. Watrous, One-dimensional-quantum walks, *STOC* 2001.
[13] D. Aharonov, A. Ambainis, J. Kempe and Vazirani, Quantum walks on graphs, *STOC* 2001.

How to define quantum control?

[11] Y. Aharonov, J. Anandan, S. Popescu and L. Vaidman, Superpositions of time evolutions of a quantum system and quantum time-translation machine, *Plysical Review Letters* 1990.
[12] A. Ambainis, E. Bach, A. Nayak, A. Vishwanath and J. Watrous, One-dimensional-quantum walks, *STOC* 2001.
[13] D. Aharonov, A. Ambainis, J. Kempe and Vazirani, Quantum walks on graphs, *STOC* 2001.

Introduce an external quantum coin c!

- state Hilbert space $\mathcal{H}_c = \text{span}\{|0\rangle, |1\rangle\}$
- U_0 and U_1 two unitary transformations on a quantum system q - state Hilbert space $\mathcal{H}_q$.

$$\begin{aligned} &\textbf{qif}\ [c]\ |0\rangle \rightarrow U_0[q] \\ &\qquad\ \ \square\ |1\rangle \rightarrow U_1[q] \\ &\textbf{fiq} \end{aligned}$$

Semantics of quantum case statement

- An unitary operator U in $\mathcal{H}_c \otimes \mathcal{H}_q$ - state Hilbert space of the composed system of coin c and principal system q:

$$U|0,\psi\rangle = |0\rangle U_0|\psi\rangle, \quad U|1,\psi\rangle = |1\rangle U_1|\psi\rangle$$

Semantics of quantum case statement

- An unitary operator U in $\mathcal{H}_c \otimes \mathcal{H}_q$ - state Hilbert space of the composed system of coin c and principal system q:

$$U|0,\psi\rangle = |0\rangle U_0|\psi\rangle, \quad U|1,\psi\rangle = |1\rangle U_1|\psi\rangle$$

- Quantum coin: superposition of $|0\rangle, |1\rangle$ —— $\alpha|0\rangle + \beta|1\rangle$.

Semantics of quantum case statement

- An unitary operator U in $\mathcal{H}_c \otimes \mathcal{H}_q$ - state Hilbert space of the composed system of coin c and principal system q:

$$U|0,\psi\rangle = |0\rangle U_0|\psi\rangle, \quad U|1,\psi\rangle = |1\rangle U_1|\psi\rangle$$

- Quantum coin: superposition of $|0\rangle, |1\rangle$ —— $\alpha|0\rangle + \beta|1\rangle$.
- Matrix representation:

$$U = |0\rangle\langle 0| \otimes U_0 + |1\rangle\langle 1| \otimes U_1 = \begin{pmatrix} U_0 & 0 \\ 0 & U_1 \end{pmatrix}.$$

Quantum Choice

- W a unitary operator in the coin's state Hilbert space $\mathcal{H}_c$.

[14] A. McIver and C. Morgan, *Abstraction, Refinement and Proof for Probabilistic Systems*, Springer 2005.

Quantum Choice

- W a unitary operator in the coin's state Hilbert space $\mathcal{H}_c$.
- Quantum choice of $U_0[q]$ and $U_1[q]$ with coin-tossing $W[c]$:

$$U_0[q] \oplus_{W[c]} U_1[q] \stackrel{\text{def}}{=} W[c];\ \mathbf{qif}\ [c]\ |0\rangle \rightarrow U_0[q]$$
$$\square\ |1\rangle \rightarrow U_1[q]$$
$$\mathbf{fiq}$$

[14] A. McIver and C. Morgan, *Abstraction, Refinement and Proof for Probabilistic Systems*, Springer 2005.

Quantum Choice

- W a unitary operator in the coin's state Hilbert space $\mathcal{H}_c$.
- Quantum choice of $U_0[q]$ and $U_1[q]$ with coin-tossing $W[c]$:

$$U_0[q] \oplus_{W[c]} U_1[q] \stackrel{\text{def}}{=} W[c];\ \mathbf{qif}\ [c]\ |0\rangle \rightarrow U_0[q] \\ \square\ |1\rangle \rightarrow U_1[q] \\ \mathbf{fiq}$$

- Compare with probabilistic choice!

[14] A. McIver and C. Morgan, *Abstraction, Refinement and Proof for Probabilistic Systems*, Springer 2005.

A more general quantum case statement

$$\begin{aligned} &\textbf{qif } [c]\ |0\rangle \rightarrow P_0 \\ &\qquad\ \square\ |1\rangle \rightarrow P_1 \\ &\textbf{fiq} \end{aligned}$$

- P_0, P_1 include quantum measurements.

[15] Chapter 6 of M. S. Ying, *Foundations of Quantum Programming*, Elsevier - Morgan Kaufmann 2016.

A more general quantum case statement

$$\begin{aligned} &\textbf{qif } [c] \ |0\rangle \rightarrow P_0 \\ &\qquad \square \ |1\rangle \rightarrow P_1 \\ &\textbf{fiq} \end{aligned}$$

- P_0, P_1 include quantum measurements.
- How to define the semantics?

[15] Chapter 6 of M. S. Ying, *Foundations of Quantum Programming*, Elsevier - Morgan Kaufmann 2016.

Outline

One-dimensional quantum walk

- One-dimensional random walk — a particle moves on a line marked by integers $\mathbb{Z}$; at each step it moves one position left or right, depending on the flip of a (fair) coin.

One-dimensional quantum walk

- One-dimensional random walk — a particle moves on a line marked by integers $\mathbb{Z}$; at each step it moves one position left or right, depending on the flip of a (fair) coin.
- Hadamard walk — a quantum variant of one-dimensional random walk.

One-dimensional quantum walk

- One-dimensional random walk — a particle moves on a line marked by integers $\mathbb{Z}$; at each step it moves one position left or right, depending on the flip of a (fair) coin.
- Hadamard walk — a quantum variant of one-dimensional random walk.
- state Hilbert space $\mathcal{H}_d \otimes \mathcal{H}_p$:

One-dimensional quantum walk

- One-dimensional random walk — a particle moves on a line marked by integers $\mathbb{Z}$; at each step it moves one position left or right, depending on the flip of a (fair) coin.
- Hadamard walk — a quantum variant of one-dimensional random walk.
- state Hilbert space $\mathcal{H}_d \otimes \mathcal{H}_p$:
  - $\mathcal{H}_d = \text{span}\{|L\rangle, |R\rangle\}$, L, R indicate the direction Left and Right.

One-dimensional quantum walk

- One-dimensional random walk — a particle moves on a line marked by integers $\mathbb{Z}$; at each step it moves one position left or right, depending on the flip of a (fair) coin.
- Hadamard walk — a quantum variant of one-dimensional random walk.
- state Hilbert space $\mathcal{H}_d \otimes \mathcal{H}_p$:
  - $\mathcal{H}_d = \text{span}\{|L\rangle, |R\rangle\}$, L, R indicate the direction Left and Right.
  - $\mathcal{H}_p = \text{span}\{|n\rangle : n \in \mathbb{Z}\}$, n indicates the position marked by integer n.

One-dimensional quantum walk

- One step of Hadamard walk — $U = T(H \otimes I)$:

One-dimensional quantum walk

- One step of Hadamard walk — $U = T(H \otimes I)$:
  - Translation T — a unitary operator in $\mathcal{H}_d \otimes \mathcal{H}_p$:

$$T|L,n\rangle = |L,n-1\rangle, \quad T|R,n\rangle = |R,n+1\rangle$$

One-dimensional quantum walk

- One step of Hadamard walk — $U = T(H \otimes I)$:
  - Translation T — a unitary operator in $\mathcal{H}_d \otimes \mathcal{H}_p$:

$$T|L,n\rangle = |L,n-1\rangle, \quad T|R,n\rangle = |R,n+1\rangle$$

  - Hadamard transform in the direction space $\mathcal{H}_d$:

$$H = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$$

One-dimensional quantum walk

- One step of Hadamard walk — $U = T(H \otimes I)$:
  - Translation T — a unitary operator in $\mathcal{H}_d \otimes \mathcal{H}_p$:

$$T|L,n\rangle = |L,n-1\rangle, \quad T|R,n\rangle = |R,n+1\rangle$$

  - Hadamard transform in the direction space $\mathcal{H}_d$:

$$H = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$$

- Hadamard walk —— repeated applications of operator W.

One-dimensional quantum walk — a different view

- Define the left and right translation operators T_L and T_R in the position space $\mathcal{H}_p$:

$$T_L|n\rangle = |n-1\rangle, \quad T_R|n\rangle = |n+1\rangle$$

One-dimensional quantum walk — a different view

- Define the left and right translation operators T_L and T_R in the position space $\mathcal{H}_p$:

$$T_L|n\rangle = |n-1\rangle, \quad T_R|n\rangle = |n+1\rangle$$

- Then the translation operator T is a quantum case statement:

$$\begin{aligned} T = \mathbf{qif}\ [d]\ & |L\rangle \rightarrow T_L[p] \\ & \square\ |R\rangle \rightarrow T_R[p] \\ \mathbf{fiq} & \end{aligned}$$

One-dimensional quantum walk — a different view

- Define the left and right translation operators T_L and T_R in the position space $\mathcal{H}_p$:

$$T_L|n\rangle = |n-1\rangle, \quad T_R|n\rangle = |n+1\rangle$$

- Then the translation operator T is a quantum case statement:

$$\begin{aligned} T = \textbf{qif}\ [d]\ & |L\rangle \rightarrow T_L[p] \\ \square\ & |R\rangle \rightarrow T_R[p] \\ \textbf{fiq} & \end{aligned}$$

- The single-step walk operator U is a quantum choice:

$$T_L[p] \oplus_{H[d]} T_R[p]$$

Unidirectional Recursive Hadamard Walk

- The walk first runs the coin-tossing Hadamard operator $H[d]$, and then a quantum case statement:

Unidirectional Recursive Hadamard Walk

- The walk first runs the coin-tossing Hadamard operator $H[d]$, and then a quantum case statement:
  - if the "direction coin" d is in state $|L\rangle$ then the walker moves one position left;

Unidirectional Recursive Hadamard Walk

- The walk first runs the coin-tossing Hadamard operator $H[d]$, and then a quantum case statement:
  - if the "direction coin" d is in state $|L\rangle$ then the walker moves one position left;
  - if d is in state $|R\rangle$ then it moves one position right, followed by *a procedure behaving as the recursive walk itself*.

Unidirectional Recursive Hadamard Walk

- The walk first runs the coin-tossing Hadamard operator $H[d]$, and then a quantum case statement:
  - if the "direction coin" d is in state $|L\rangle$ then the walker moves one position left;
  - if d is in state $|R\rangle$ then it moves one position right, followed by *a procedure behaving as the recursive walk itself*.
- The walk —— a recursive program X declared by the recursive equation:

$$X \Leftarrow T_L[p] \oplus_{H[d]} (T_R[p]; X)$$

Bidirectional Recursive Hadamard Walk

- The walk first runs the coin-tossing Hadamard operator $H[d]$ and then a quantum case statement:

Bidirectional Recursive Hadamard Walk

- The walk first runs the coin-tossing Hadamard operator $H[d]$ and then a quantum case statement:
  - if the direction coin d is in state $|L\rangle$ then the walker moves one position left, followed by *a procedure behaving as the recursive walk itself*;

Bidirectional Recursive Hadamard Walk

- The walk first runs the coin-tossing Hadamard operator $H[d]$ and then a quantum case statement:
  - if the direction coin d is in state $|L\rangle$ then the walker moves one position left, followed by *a procedure behaving as the recursive walk itself*;
  - if d is in state $|R\rangle$ then it moves one position right, also followed by *a procedure behaving as the recursive walk itself*.

Bidirectional Recursive Hadamard Walk

- The walk first runs the coin-tossing Hadamard operator $H[d]$ and then a quantum case statement:
  - if the direction coin d is in state $|L\rangle$ then the walker moves one position left, followed by *a procedure behaving as the recursive walk itself*;
  - if d is in state $|R\rangle$ then it moves one position right, also followed by *a procedure behaving as the recursive walk itself*.
- The walk — a program X (or Y) declared by the equation:

$$X \Leftarrow (T_L[p]; X) \oplus_{H[d]} (T_R[p]; X)$$

A More Interesting Recursive Quantum Walk

- Let $n \geq 2$. A variant of unidirectional recursive quantum walk:

$$X \Leftarrow ((T_L[p]; X) \oplus_{H[d]} (T_R[p]; X)); (T_L[p] \oplus_{H[d]} T_R[p])^n$$

A More Interesting Recursive Quantum Walk

- Let $n \geq 2$. A variant of unidirectional recursive quantum walk:

$$X \Leftarrow ((T_L[p]; X) \oplus_{H[d]} (T_R[p]; X)); (T_L[p] \oplus_{H[d]} T_R[p])^n$$

How to solve these quantum recursive equations?

Syntactic Approximation

- A recursive program X declared by equation

$$X \Leftarrow F(X)$$

Syntactic Approximation

- A recursive program X declared by equation

$$X \Leftarrow F(X)$$

- Syntactic approximations:

$$\begin{cases} X^{(0)} = \textbf{Abort}, \\ X^{(n+1)} = F[X^{(n)}/X] \text{ for } n \geq 0. \end{cases}$$

Program $X^{(n)}$ is the nth syntactic approximation of X.

Syntactic Approximation

- A recursive program X declared by equation

$$X \Leftarrow F(X)$$

- Syntactic approximations:

$$\begin{cases} X^{(0)} = \textbf{Abort}, \\ X^{(n+1)} = F[X^{(n)}/X] \text{ for } n \geq 0. \end{cases}$$

  Program $X^{(n)}$ is the nth syntactic approximation of X.
- Semantics $[\![X]\!]$ of X is the limit

$$[\![X]\!] = \lim_{n\to\infty} [\![X^{(n)}]\!]$$

Example - Unidirectional Recursive Hadamard Walk

$$X^{(0)} = \mathbf{abort},$$

$$X^{(1)} = T_L[p] \oplus_{H[d]} (T_R[p]; \mathbf{abort}),$$

$$X^{(2)} = T_L[p] \oplus_{H[d]} (T_R[p]; T_L[p] \oplus_{H[d_1]} (T_R[p]; \mathbf{abort})),$$

$$X^{(3)} = T_L[p] \oplus_{H[d]} (T_R[p]; T_L[p] \oplus_{H[d_1]} (T_R[p]; T_L[p] \oplus_{H[d_2]} (T_R[p]; \mathbf{abort}))),$$

....

Example - Unidirectional Recursive Hadamard Walk

$$X^{(0)} = \textbf{abort},$$
$$X^{(1)} = T_L[p] \oplus_{H[d]} (T_R[p]; \textbf{abort}),$$
$$X^{(2)} = T_L[p] \oplus_{H[d]} (T_R[p]; T_L[p] \oplus_{H[d_1]} (T_R[p]; \textbf{abort})),$$
$$X^{(3)} = T_L[p] \oplus_{H[d]} (T_R[p]; T_L[p] \oplus_{H[d_1]} (T_R[p]; T_L[p] \oplus_{H[d_2]} (T_R[p]; \textbf{abort}))),$$

....

Observations

- Continuously introduce new coin to avoid variable conflict.
- Variables $d, d_1, d_2, ...$ denote identical particles.

Example - Unidirectional Recursive Hadamard Walk

$$
\begin{aligned}
X^{(0)} &= \textbf{abort}, \\
X^{(1)} &= T_L[p] \oplus_{H[d]} (T_R[p]; \textbf{abort}), \\
X^{(2)} &= T_L[p] \oplus_{H[d]} (T_R[p]; T_L[p] \oplus_{H[d_1]} (T_R[p]; \textbf{abort})), \\
X^{(3)} &= T_L[p] \oplus_{H[d]} (T_R[p]; T_L[p] \oplus_{H[d_1]} (T_R[p]; T_L[p] \oplus_{H[d_2]} (T_R[p]; \textbf{abort}))),
\end{aligned}
$$

....

Observations

- Continuously introduce new coin to avoid variable conflict.
- Variables $d, d_1, d_2, ...$ denote identical particles.
- The number of the coin particles that are needed in running the recursive walk is unknown beforehand.

Example - Unidirectional Recursive Hadamard Walk

$$X^{(0)} = \textbf{abort},$$
$$X^{(1)} = T_L[p] \oplus_{H[d]} (T_R[p]; \textbf{abort}),$$
$$X^{(2)} = T_L[p] \oplus_{H[d]} (T_R[p]; T_L[p] \oplus_{H[d_1]} (T_R[p]; \textbf{abort})),$$
$$X^{(3)} = T_L[p] \oplus_{H[d]} (T_R[p]; T_L[p] \oplus_{H[d_1]} (T_R[p]; T_L[p] \oplus_{H[d_2]} (T_R[p]; \textbf{abort}))),$$

....

Observations

- Continuously introduce new coin to avoid variable conflict.
- Variables $d, d_1, d_2, ...$ denote identical particles.
- The number of the coin particles that are needed in running the recursive walk is unknown beforehand.
- We need to deal with *quantum systems where the number of particles of the same type may vary*.

Outline

Fock Spaces

- ***The principle of symmetrisation***: the states of n identical particles are either completely symmetric or completely antisymmetric with respect to the permutations of the particles. [bosons - symmetric; fermions - antisymmetric]

Fock Spaces

- ***The principle of symmetrisation***: the states of n identical particles are either completely symmetric or completely antisymmetric with respect to the permutations of the particles. [bosons - symmetric; fermions - antisymmetric]
- Let $\mathcal{H}$ be the state Hilbert space of one particle.

Fock Spaces

- ***The principle of symmetrisation***: the states of n identical particles are either completely symmetric or completely antisymmetric with respect to the permutations of the particles. [bosons - symmetric; fermions - antisymmetric]
- Let $\mathcal{H}$ be the state Hilbert space of one particle.
- For each permutation π of $1, ..., n$, define the permutation operator P_π in $\mathcal{H}^{\otimes n}$:

$$P_\pi |\psi_1 \otimes ... \otimes \psi_n\rangle = |\psi_{\pi(1)} \otimes ... \otimes \psi_{\pi(n)}\rangle$$

Fock Spaces

- ***The principle of symmetrisation***: the states of n identical particles are either completely symmetric or completely antisymmetric with respect to the permutations of the particles. [bosons - symmetric; fermions - antisymmetric]
- Let $\mathcal{H}$ be the state Hilbert space of one particle.
- For each permutation π of $1, ..., n$, define the permutation operator P_π in $\mathcal{H}^{\otimes n}$:

$$P_\pi |\psi_1 \otimes ... \otimes \psi_n\rangle = |\psi_{\pi(1)} \otimes ... \otimes \psi_{\pi(n)}\rangle$$

- Define the symmetrisation and antisymmetrisation operators in $\mathcal{H}^{\otimes n}$:

$$S_+ = \frac{1}{n!} \sum_\pi P_\pi, \qquad S_- = \frac{1}{n!} \sum_\pi (-1)^\pi P_\pi$$

Fock Spaces

$v = +$ for bosons, $v = -$ for fermions.

- Symmetrisation or antisymmetrisation:

$$|\psi_1, ..., \psi_n\rangle_v = S_v|\psi_1 \otimes ... \otimes \psi_n\rangle.$$

Fock Spaces

$v = +$ for bosons, $v = -$ for fermions.

- Symmetrisation or antisymmetrisation:

$$|\psi_1, ..., \psi_n\rangle_v = S_v|\psi_1 \otimes ... \otimes \psi_n\rangle.$$

- State space of n bosons and that of fermions:

$$\mathcal{H}_v^{\otimes n} = S_v\mathcal{H}^{\otimes n} = \text{span}\{|\psi_1, ..., \psi_n\rangle_v : |\psi_1\rangle, ..., |\psi_n\rangle \text{ are in } \mathcal{H}\}$$

Fock Spaces

$v = +$ for bosons, $v = -$ for fermions.

- Symmetrisation or antisymmetrisation:

$$|\psi_1, ..., \psi_n\rangle_v = S_v|\psi_1 \otimes ... \otimes \psi_n\rangle.$$

- State space of n bosons and that of fermions:

$$\mathcal{H}_v^{\otimes n} = S_v\mathcal{H}^{\otimes n} = \text{span}\{|\psi_1, ..., \psi_n\rangle_v : |\psi_1\rangle, ..., |\psi_n\rangle \text{ are in } \mathcal{H}\}$$

- Introduce the vacuum state $|\mathbf{0}\rangle$:

$$\mathcal{H}_v^{\otimes 0} = \mathcal{H}^{\otimes 0} = \text{span}\{|\mathbf{0}\rangle\}.$$

Fock Spaces

$v = +$ for bosons, $v = -$ for fermions.

- Symmetrisation or antisymmetrisation:

$$|\psi_1, ..., \psi_n\rangle_v = S_v|\psi_1 \otimes ... \otimes \psi_n\rangle.$$

- State space of n bosons and that of fermions:

$$\mathcal{H}_v^{\otimes n} = S_v \mathcal{H}^{\otimes n} = \text{span}\{|\psi_1, ..., \psi_n\rangle_v : |\psi_1\rangle, ..., |\psi_n\rangle \text{ are in } \mathcal{H}\}$$

- Introduce the vacuum state $|\mathbf{0}\rangle$:

$$\mathcal{H}_v^{\otimes 0} = \mathcal{H}^{\otimes 0} = \text{span}\{|\mathbf{0}\rangle\}.$$

- The space of the states of variable particle number is the Fock space:

$$\mathcal{F}_v(\mathcal{H}) = \sum_{n=0}^{\infty} \mathcal{H}_v^{\otimes n}$$

Evolution in the Fock Spaces

- (discrete-time) evolution of one particle —— unitary operator U.

Evolution in the Fock Spaces

- (discrete-time) evolution of one particle —— unitary operator U.
- Evolution of n particles without mutual interactions is operator $\mathbf{U}$ in $\mathcal{H}^{\otimes n}$:

$$\mathbf{U}|\psi_1 \otimes ... \otimes \psi_n\rangle = |U\psi_1 \otimes ... \otimes U\psi_n\rangle$$

Evolution in the Fock Spaces

- (discrete-time) evolution of one particle —— unitary operator U.
- Evolution of n particles without mutual interactions is operator $\mathbf{U}$ in $\mathcal{H}^{\otimes n}$:

$$\mathbf{U}|\psi_1 \otimes ... \otimes \psi_n\rangle = |U\psi_1 \otimes ... \otimes U\psi_n\rangle$$

- Symmetrisation or antisymmetrisation:

$$\mathbf{U}|\psi_1, ..., \psi_n\rangle_v = |U\psi_1, ...U\psi_n\rangle_v.$$

Evolution in the Fock Spaces

- (discrete-time) evolution of one particle —— unitary operator U.
- Evolution of n particles without mutual interactions is operator $\mathbf{U}$ in $\mathcal{H}^{\otimes n}$:

$$\mathbf{U}|\psi_1 \otimes ... \otimes \psi_n\rangle = |U\psi_1 \otimes ... \otimes U\psi_n\rangle$$

- Symmetrisation or antisymmetrisation:

$$\mathbf{U}|\psi_1, ..., \psi_n\rangle_v = |U\psi_1, ...U\psi_n\rangle_v.$$

- Extend to the Fock spaces $\mathcal{F}_v(\mathcal{H})$ and $\mathcal{F}(\mathcal{H})$:

$$\mathbf{U}\left(\sum_{n=0}^{\infty} |\Psi(n)\rangle\right) = \sum_{n=0}^{\infty} \mathbf{U}|\Psi(n)\rangle$$

Creation and Annihilation of Particles

- Transitions between states of different particle numbers.

Creation and Annihilation of Particles

- Transitions between states of different particle numbers.
- Creation operator $a^*(\psi)$ in $\mathcal{F}_v(\mathcal{H})$:

$$a^*(\psi)|\psi_1, ..., \psi_n\rangle_v = \sqrt{n+1}|\psi, \psi_1, ..., \psi_n\rangle_v$$

  Add a particle in the individual state $|\psi\rangle$ to the system of n particles without modifying their respective states.

Creation and Annihilation of Particles

- Transitions between states of different particle numbers.
- Creation operator $a^*(\psi)$ in $\mathcal{F}_v(\mathcal{H})$:

$$a^*(\psi)|\psi_1, ..., \psi_n\rangle_v = \sqrt{n+1}|\psi, \psi_1, ..., \psi_n\rangle_v$$

  Add a particle in the individual state $|\psi\rangle$ to the system of n particles without modifying their respective states.
- Annihilation operator $a(\psi)$ — the Hermitian conjugate of $a^*(\psi)$:

$$a(\psi)|\mathbf{0}\rangle = 0,$$

$$a(\psi)|\psi_1, ..., \psi_n\rangle_v = \frac{1}{\sqrt{n}} \sum_{i=1}^{n} (v)^{i-1} \langle\psi|\psi_i\rangle |\psi_1, ..., \psi_{i-1}, \psi_{i+1}, ..., \psi_n\rangle_v$$

  Decrease the number of particles by one unit, while preserving the symmetry of the state.

Outline

Example - Unidirectional Recursive Hadamard Walk

Semantics of the recursive Hadamard walk:

$$[\![X]\!] = \left[\sum_{i=0}^{\infty}\left(\bigotimes_{j=0}^{i-1}|R\rangle_{d_j}\langle R| \otimes |L\rangle_{d_i}\langle L|\right) \otimes T_L T_R^i\right](\mathbf{H} \otimes I)$$

- An operator in

$$\mathcal{F}_v(\mathcal{H}_d) \otimes \mathcal{H}_p \to \mathcal{F}(\mathcal{H}_d) \otimes \mathcal{H}_p.$$

Example - Unidirectional Recursive Hadamard Walk

Semantics of the recursive Hadamard walk:

$$[\![X]\!] = \left[\sum_{i=0}^{\infty}\left(\bigotimes_{j=0}^{i-1}|R\rangle_{d_j}\langle R| \otimes |L\rangle_{d_i}\langle L|\right)\otimes T_L T_R^i\right](\mathbf{H}\otimes I)$$

- An operator in

$$\mathcal{F}_v(\mathcal{H}_d)\otimes\mathcal{H}_p \to \mathcal{F}(\mathcal{H}_d)\otimes\mathcal{H}_p.$$

- The sign v is $+$ or $-$, depending on using bosons or fermions to implement the direction coins $d, d_1, d_2, \ldots$.

Principal System Semantics

- Each state $|\Psi\rangle$ in Fock space $\mathcal{F}_v(\mathcal{H}_d)$ induces mapping:

$$[\![X, \Psi]\!]_p : \text{pure states} \to \text{partial density operators in } \mathcal{H}_p$$
$$[\![X, \Psi]\!]_p(|\psi\rangle) = tr_{\mathcal{F}(\mathcal{H}_d)}(|\Phi\rangle\langle\Phi|)$$

where $|\Phi\rangle = [\![X]\!](|\Psi\rangle \otimes |\psi\rangle)$

Principal System Semantics

- Each state $|\Psi\rangle$ in Fock space $\mathcal{F}_v(\mathcal{H}_d)$ induces mapping:

$$[\![X, \Psi]\!]_p : \text{pure states} \rightarrow \text{partial density operators in } \mathcal{H}_p$$
$$[\![X, \Psi]\!]_p(|\psi\rangle) = tr_{\mathcal{F}(\mathcal{H}_d)}(|\Phi\rangle\langle\Phi|)$$

  where $|\Phi\rangle = [\![X]\!](|\Psi\rangle \otimes |\psi\rangle)$

- $[\![X, \Psi]\!]_p$ is called the principal system semantics of X with coin initialisation $|\Psi\rangle$.

Example - Bidirectional Recursive Quantum Walk

$$\begin{cases} X \Leftarrow T_L[p] \oplus_{H[d]} (T_R[p]; Y), \\ Y \Leftarrow (T_L[p]; X) \oplus_{H[d]} T_R[p] \end{cases}$$

- Coherent state of bosons in the symmetric Fock space $\mathcal{F}_+(\mathcal{H})$ over $\mathcal{H}$:

$$|\psi\rangle_{\text{coh}} = \exp\left(-\frac{1}{2}\langle\psi|\psi\rangle\right) \sum_{n=0}^{\infty} \frac{[a^*(\psi)]^n}{n!} |\mathbf{0}\rangle$$

[16] Chapter 7 of M. S. Ying, *Foundations of Quantum Programming*, Elsevier - Morgan Kaufmann 2016.

Example - Bidirectional Recursive Quantum Walk

$$\begin{cases} X \Leftarrow T_L[p] \oplus_{H[d]} (T_R[p]; Y), \\ Y \Leftarrow (T_L[p]; X) \oplus_{H[d]} T_R[p] \end{cases}$$

- Coherent state of bosons in the symmetric Fock space $\mathcal{F}_+(\mathcal{H})$ over $\mathcal{H}$:

$$|\psi\rangle_{\mathrm{coh}} = \exp\left(-\frac{1}{2}\langle\psi|\psi\rangle\right) \sum_{n=0}^{\infty} \frac{[a^*(\psi)]^n}{n!} |\mathbf{0}\rangle$$

- The walk starts from position 0 and the coins are initialised in the coherent states of bosons corresponding to $|L\rangle$:

$$\begin{aligned} [\![X, L_{\mathrm{coh}}]\!]_p(|0\rangle) &= \frac{1}{\sqrt{e}}\left(\sum_{k=0}^{\infty} \frac{1}{2^{2k+1}}|-1\rangle\langle -1| + \sum_{k=0}^{\infty} \frac{1}{2^{2k+2}}|2\rangle\langle 2|\right) \\ &= \frac{1}{\sqrt{e}}\left(\frac{2}{3}|-1\rangle\langle -1| + \frac{1}{3}|2\rangle\langle 2|\right). \end{aligned}$$

[16] Chapter 7 of M. S. Ying, *Foundations of Quantum Programming*, Elsevier - Morgan Kaufmann 2016

Outline

Quantum programming theory

- Imperative quantum programs
  [1] M. S. Ying, Floyd-Hoare logic for quantum programs, *TOPLAS* 2011.
  [2] M. S. Ying, S. G. Ying and X. D. Wu, Invariants of quantum programs: characterisations and generation, *POPL* 2017.

Quantum programming theory

- Imperative quantum programs
  [1] M. S. Ying, Floyd-Hoare logic for quantum programs, *TOPLAS* 2011.
  [2] M. S. Ying, S. G. Ying and X. D. Wu, Invariants of quantum programs: characterisations and generation, *POPL* 2017.
- Functional quantum programs
  [1] M. Pagani, P. Selinger and B. Valiron, Applying quantitative semantics to higher-order quantum computing, *POPL* 2014.
  [2] S. Staton, Algebraic effects, linearity, and quantum programming languages, *POPL* 2015.

Quantum programming theory

- Imperative quantum programs
  [1] M. S. Ying, Floyd-Hoare logic for quantum programs, *TOPLAS* 2011.
  [2] M. S. Ying, S. G. Ying and X. D. Wu, Invariants of quantum programs: characterisations and generation, *POPL* 2017.
- Functional quantum programs
  [1] M. Pagani, P. Selinger and B. Valiron, Applying quantitative semantics to higher-order quantum computing, *POPL* 2014.
  [2] S. Staton, Algebraic effects, linearity, and quantum programming languages, *POPL* 2015.
- Concurrent quantum programs
  [1] S. J. Gay and R. Nagarajan, Communicating quantum processes, *POPL* 2005.
  [2] Y. Feng, R. Y. Duan and M. S. Ying, Bisimulations for quantum processes, *POPL* 2011 or *TOPLAS* 2012.

Quantum programming languages

- LIQUi|>, Microsoft, 2015.

Quantum programming languages

- LIQUi|>, Microsoft, 2015.
- Quipper, *PLDI* 2013.

Quantum computing startups mushrooming!

Quantum programming languages

- LIQUi|>, Microsoft, 2015.
- Quipper, *PLDI* 2013.
- Scaffold — ScaffCC, Princeton, UCSB, IBM, 2013 - 15.

Quantum computing startups mushrooming!

You will be quantum Bill Gates!

Quantum programming languages

- LIQUi | >, Microsoft, 2015.
- Quipper, *PLDI* 2013.
- Scaffold — ScaffCC, Princeton, UCSB, IBM, 2013 - 15.

Quantum computing startups mushrooming!

- D-Wave Systems, Rigetti Computing, IonQ, Cambridge Quantum Computing Ltd, 1QBit, QC Ware, ...

You will be quantum Bill Gates!

THANK YOU!