Introduction to Quantum Programming and Semantics

Lecture 11: Classical quantum circuits

Chris Heunen

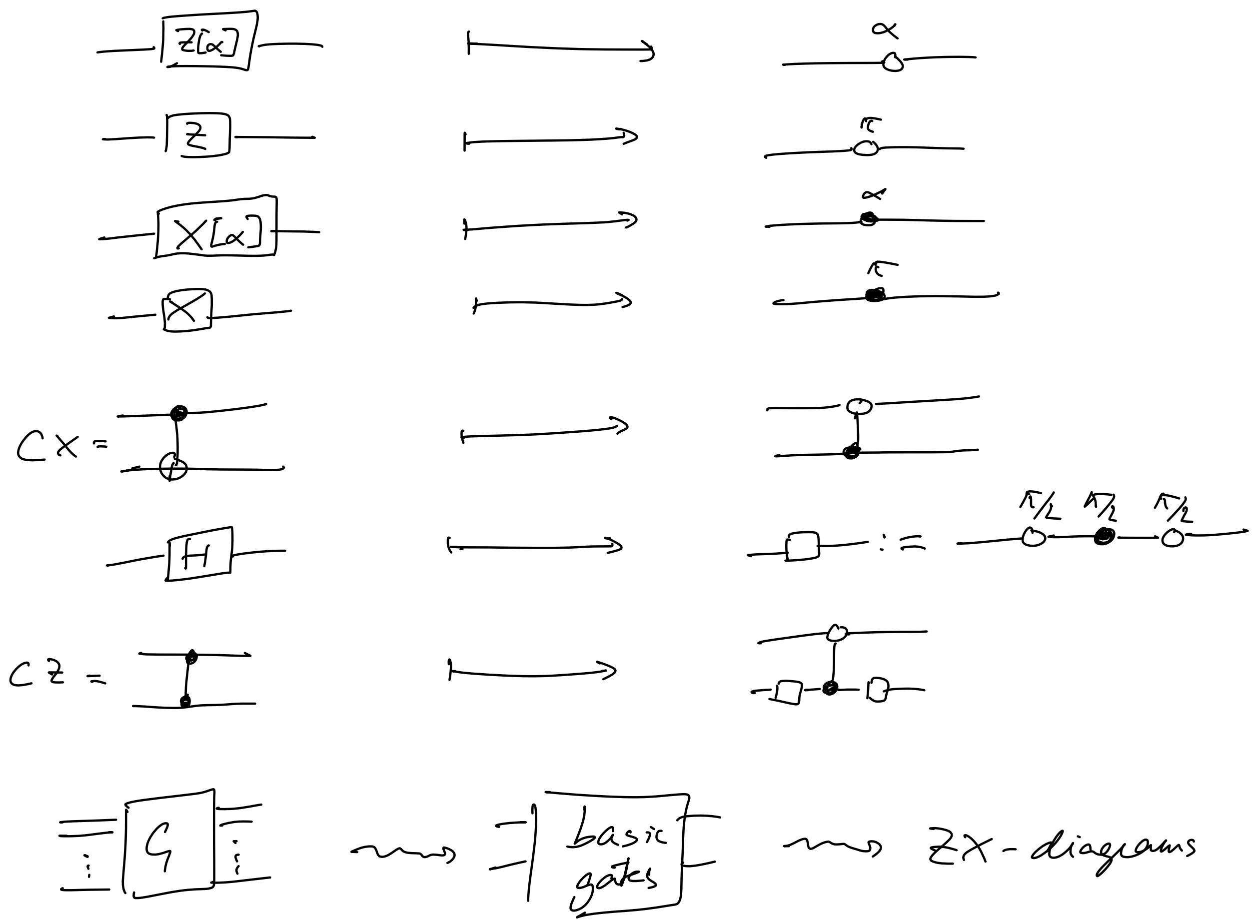
University of Edinburgh

Overview

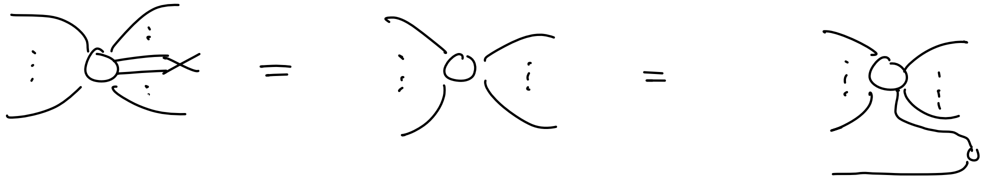
- CNOT circuits
- Bennett’s trick
- Phase-free ZX diagrams

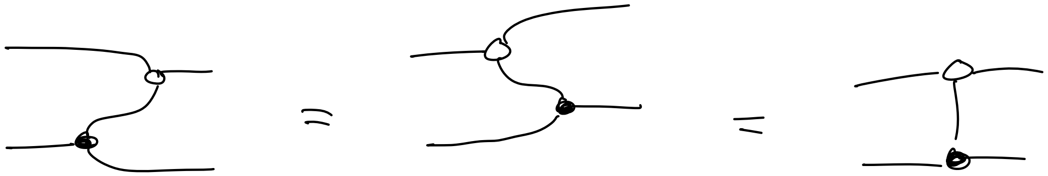
dictionary

circuit $\longrightarrow$ ZX-diagram



ZX diagrams have extreme 'only connectivity matters':





ZX calculus is a tool to reason about quantum circuits. Solve problems like:

- synthesis: given a high-level description of a computation/unitary, find a circuit that implements it

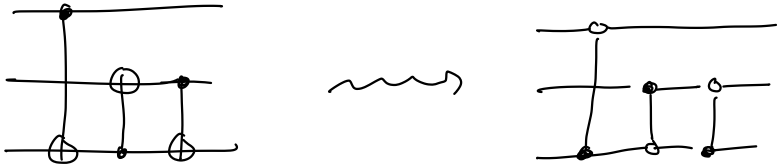
- optimisation: given a circuit C that implements U, find a 'smaller' C' that also implements U.

- (classical) simulation: given C that implements U, and an input $|x\rangle$, (not given quantum hardware)
  - compute measurement probabilities for $U|\psi\rangle$ ("strong")
  - sample measurement outcome for $U|\psi\rangle$ ("weak")

CNOT circuits

Parity

any CNOT circuit induces a phase-free ZX-diagram



what about converse?

def: a function of form $f(x_1, \dots, x_n) = x_{i_1} \oplus \cdots \oplus x_{i_n}$ is called "parity function"

XOR

can think about this as linear algebra over $\mathbb{F}_2 = \{0,1\}$,

$x \cdot y = x \text{ AND } y$

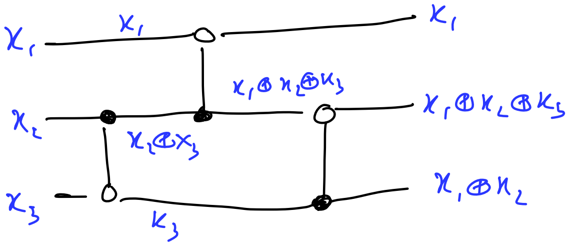
$x + y = x \text{ XOR } y$

e.g. $$(1\ 0\ 1\ 1)\begin{pmatrix} a \\ b \\ c \\ d \end{pmatrix} = a \oplus c \oplus d$$

$$\underbrace{\begin{pmatrix} 1 & 0 & 1 & 1 \\ 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 \end{pmatrix}}_{\text{"parity matrix"}} \begin{pmatrix} a \\ b \\ c \\ d \end{pmatrix} = \begin{pmatrix} a \oplus c \oplus d \\ b \oplus c \\ a \oplus d \\ d \end{pmatrix}$$

can turn CNOT into parity matrix:

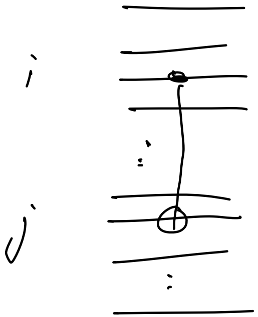
Parity matrix

$C =$ 

$$C|x_1, x_2 x_3\rangle = |x_1, x_1 \oplus x_2 \oplus x_3, x_1 \oplus x_2\rangle$$

$$\begin{pmatrix} 1 & 0 & 0 \\ 1 & 1 & 1 \\ 1 & 1 & 0 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} x_1 \\ x_1 \oplus x_2 \oplus x_3 \\ x_1 \oplus x_2 \end{pmatrix}$$

more generally:



gives parity matrix $$E^{ij} = \begin{array}{cl} \begin{array}{ccccc} & i & & & \\ & \downarrow & & & \end{array} & \\ \begin{pmatrix} 1 & & & & \\ & 1 & & 0 & \\ & & \ddots & & \\ & 1 & & \ddots & \\ 0 & & & & 1 \end{pmatrix} & \leftarrow j \end{array}$$

elementary matrix operation

CNOT circuit synthesis

- start with a parity matrix P, empty circuit C
- do Gauss-Jordan elimination on columns of P

  whenever apply E^{ij}, append CNOT to C

  with control i target j

  swap two rows, use

- repeat until P eliminated

then C implements P

Bennett’s trick

Reversible computing

Where do circuits come from?

One source: classical computations

$f:\{0,1\}^n \longrightarrow \{0,1\}^n \quad \rightsquigarrow \quad (\mathbb{C}^2)^{\otimes n} \xrightarrow{U_f} (\mathbb{C}^2)^{\otimes n}$

$|x\rangle \longmapsto |f(x)\rangle$

$f:\{0,1\}^n \longrightarrow \{0,1\} \quad \underset{\text{Bennett's trick}}{\rightsquigarrow} \quad (\mathbb{C}^2)^{\otimes n+1} \xrightarrow{U_f} (\mathbb{C}^2)^{\otimes n+1}$

$|x,y\rangle \longmapsto |x, y\oplus f(x)\rangle$

e.g.

$NOT : \{0,1\} \longrightarrow \{0,1\} \quad \rightsquigarrow \quad X : \begin{array}{l} |0\rangle \longmapsto |1\rangle \\ |1\rangle \longmapsto |0\rangle \end{array}$

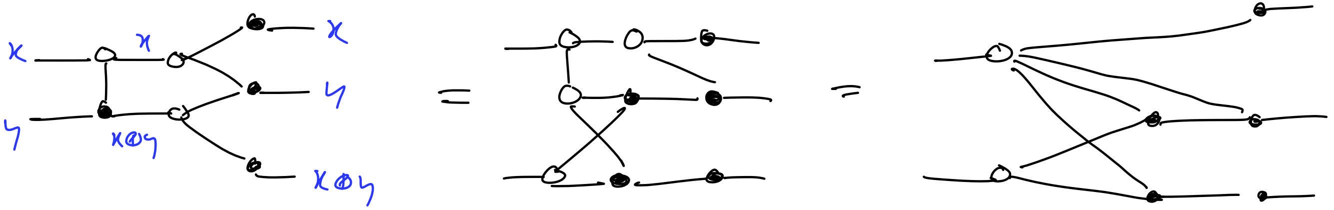
$CNOT : \{0,1\}^2 \longrightarrow \{0,1\}^2 \quad \rightsquigarrow \quad CX : |x,y\rangle \longmapsto |x, x\oplus y\rangle$

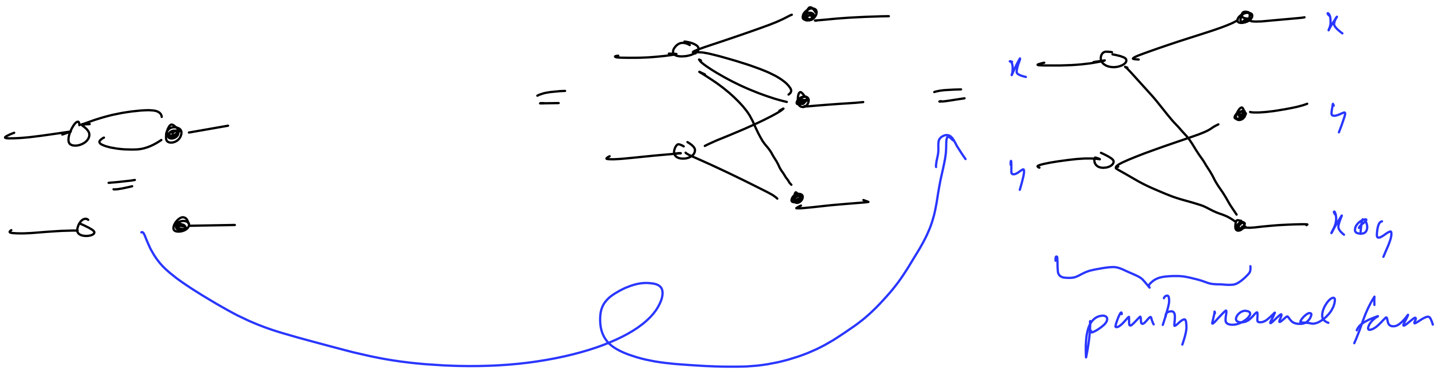
Phase-free ZX diagrams

Parity normal form

parity matrix

$$\begin{pmatrix} 1 & 0 \\ 0 & 1 \\ 1 & 1 \end{pmatrix} \rightsquigarrow$$





Reduction to parity form

unitary phase-free ZX-diagram $\rightsquigarrow$ parity normal form $\rightsquigarrow$ parity matrix $\rightsquigarrow$ CNOT circuit

unitary phase-free ZX-diagram $\longleftarrow$ dictionary $\longleftarrow$ CNOT circuit

Summary:

- ZX diagrams are (complexity-theoretically) hard to rewrite algorithmically
- Special class of CNOT diagrams easier, comes down to parity checking
- Bennett's trick makes classical functions reversible, useful for oracles
- Special class of phase-free ZX diagrams easier