THEORY OF COMPUTATION

Turing machines

A (deterministic) <u>Turing machine</u> (TM) is a tuple

$$M = (Q, \Sigma, \Gamma, \delta)$$

where

Q is a finite set containing the elements

q_0 : start state
q_a : accept state
q_r : reject state.

Γ is a finite set containing

$\triangleright$: start symbol
$\sqcup$: blank symbol

and a subset Σ not containing $\sqcup$.

δ is a function (transition function)

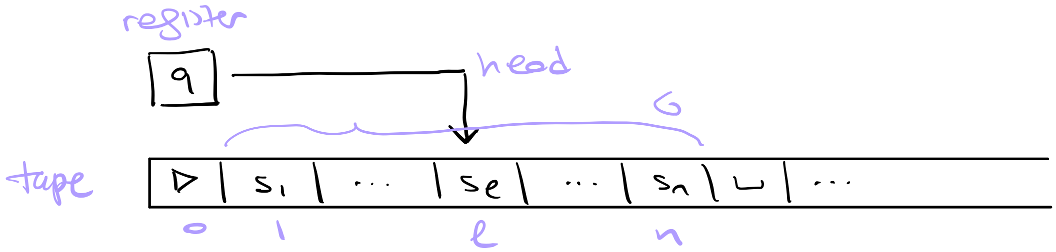
$$\delta : Q \times \Gamma \to Q \times \Gamma \times \{-1, 0, 1\}$$

such that

$$\delta(q, \triangleright) = (q', \triangleright, \Delta), \quad \Delta \neq -1.$$
$$\delta(q_a, \sigma) = (q_a, \sigma, 0),$$
$$\delta(q_r, \sigma) = (q_r, \sigma, 0).$$

Interpretation : Q : states
Γ : alphabet
Σ : input alphabet
δ : transition function

Representation of M as a tape:



Configuration at a given step is denoted by $\langle q, \sigma, \ell \rangle$ where

$$q \in Q$$
$$\ell \in \mathbb{N}$$
$$\sigma \in \Gamma^* = \bigsqcup_{k \in \mathbb{N}} \Gamma^k.$$

Initial configuration is given by

$$\langle q_0, \sigma, 0 \rangle$$

where $\sigma = s_1 \cdots s_n \in \Sigma^*$ is the input.

If $\langle q, \sigma, \ell \rangle$ is the configuration at step t then the configuration $\langle q', \sigma', \ell' \rangle$ at step $t+1$ is given by

$$\sigma' = s_1 \cdots s_{\ell-1} s'_\ell s_{\ell+1} \cdots s_n$$
$$\ell' = \ell + \Delta, \quad \Delta \in \{-1, 0, 1\},$$

where q', s'_ℓ, Δ are determined by

$$\delta(q, s_\ell) = (q', s'_\ell, \Delta).$$

We require M to satisfy one of the three possibilities:

1) M halts with q_r:

$$M(\sigma) = 0$$ (M rejects σ)

2) M halts with q_a:

$$M(\sigma) = 1$$ (M accepts σ)

3) M does not halt. (M loops)

If M halts then the final configuration is

$$\langle q, \tilde{\sigma}, \ell \rangle$$

where $q = q_a$ or q_r and

$\tilde{\sigma} = \vec{s}_1 \ldots \vec{s}_m$ is the output.

Let $\varphi_M : \Sigma^* \longrightarrow \Sigma^*$ denote the $\overbrace{\text{partial}}^{\text{defined on a subset}}$ function

$$\varphi_M(\sigma) = \tilde{\sigma}$$

if $M(\sigma) \in \{0,1\}$ and $\tilde{\sigma} \in \Sigma^*$.

A TM such that $M(\sigma) \in \{0,1\}$ for all $\sigma \in \Sigma^*$ is called a <u>decider</u>.

A function $f : \Sigma^* \longrightarrow \Sigma^*$ is <u>computable</u> if $\exists$ a decider TM M such that

$$f(\sigma) = \varphi_M(\sigma) \quad , \quad \forall \sigma \in \Sigma^*.$$

A subset $L \subset \Sigma^*$ is called a language.

To a TM M we can associate the language

$$L(M) = \{ \sigma \in \Sigma^* : M(\sigma) = 1 \}.$$

A language L is Turing-recognizable if $\exists$ TM $\underbrace{M}$ such that

not required to be decider

$$L = L(M).$$

A language is decidable if $\exists$ a decider TM M such that

$$L = L(M).$$

A function $P : \Sigma^* \longrightarrow \mathbb{B}$ is called a predicate.

A predicate can be identified with the language

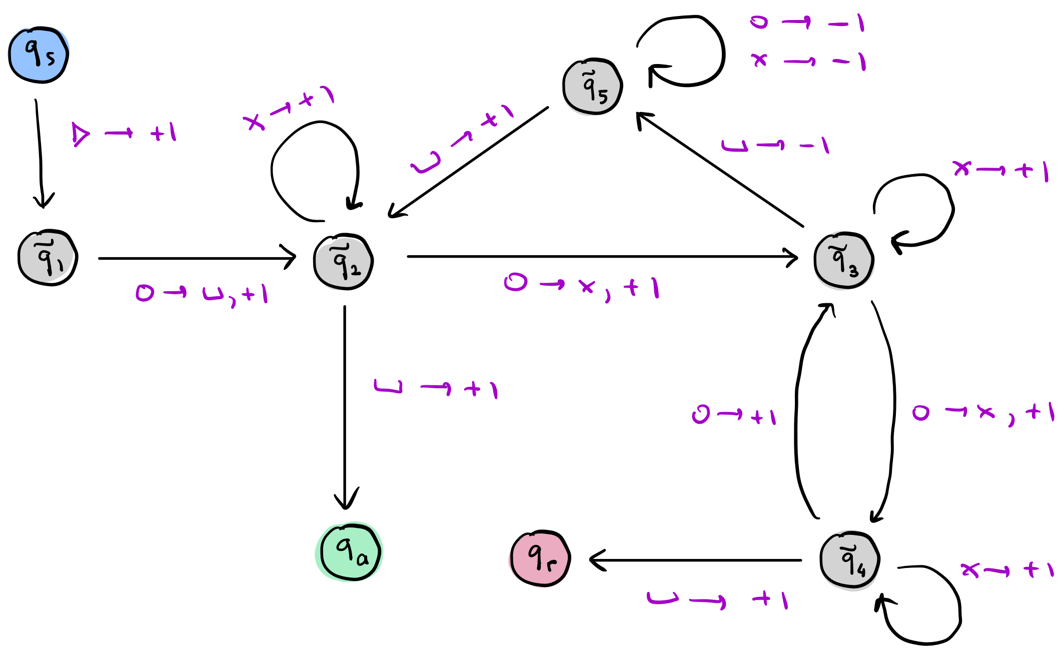
$$L(P) = \{ \sigma \in \Sigma^* : P(\sigma) = 1 \}$$

We say P is computable (or decidable) if $L(P)$ is decidable.

Ex: Let $\Sigma = \{0\}$.

$L = \{ 0^{2^n} : n \geq 0 \} \subset \Sigma^*$ is decidable:

We describe the TM that decides this language:



$\Gamma = \{ \triangleright, \sqcup, x \} \sqcup \{ 0 \}$.

For example:

$$\begin{aligned}
\langle q_s, 00, 0 \rangle &\mapsto \langle \tilde{q}_1, 00, 1 \rangle \\
&\mapsto \langle \tilde{q}_2, \sqcup 0, 2 \rangle \\
&\mapsto \langle \tilde{q}_3, \sqcup x, 3 \rangle \\
&\mapsto \langle \tilde{q}_5, \sqcup x, 2 \rangle \\
&\mapsto \langle \tilde{q}_5, \sqcup x, 1 \rangle \\
&\mapsto \langle q_2, \sqcup x, 2 \rangle \quad \text{(accept)}
\end{aligned}$$

HW: Test other inputs.

Universal Turing machine

To each TM M we will assign a bit string

$$M \longmapsto \langle M \rangle \in \mathbb{B}^*$$

in the following way:

We choose binary representation for the following additional characters

$$s \quad q \quad (\quad) \quad ,$$

Then

1) Encode elements of Q by

$$\tilde{q} \longmapsto q\, b_i \cdots b_0 \in \mathbb{B}^*$$

2) Encode elements of Γ by

$$\Sigma \ni \tilde{s} \longmapsto s\, b_j \cdots b_0 \in \mathbb{B}^*$$

and the special symbols by

$$0^{j-1} = \underbrace{0 \cdots 0}_{j-1}$$

$$\begin{aligned}
& \sqcup && \longmapsto && s\, 0^{j-1}\, 00 \\
\text{stay} \quad & 0 && \longmapsto && s\, 0^{j-1}\, 01 \\
\text{left move} \quad & -1 && \longmapsto && s\, 0^{j-1}\, 10 \\
\text{right move} \quad & +1 && \longmapsto && s\, 0^{j-1}\, 11
\end{aligned}$$

where i and j are large enough so that each element is represented by a distinct string.

3) Encoding δ:

$$\delta(\tilde{q}, \tilde{s}) = (\tilde{q}', \tilde{s}', \ell)$$

$$(q b_i \cdots b_0,\ s a_j \cdots a_0,\ q b_i' \cdots b_0',\ s a_j' \cdots a_0',\ \alpha)$$

where α is either $s\,0^{j-1}01$ or $s\,0^{j-1}10$ or $s\,0^{j-1}11$.

Let $\langle M \rangle \in \mathbb{B}^*$ denote the string obtained by concatenating strings in (3) separated by a comma in the lexicographic order.

We can associate a natural number:

$$M \longmapsto \langle M \rangle \longmapsto n \in \mathbb{N}$$

where the last map is

$$b_n \cdots b_0 \in \mathbb{B}^* \longmapsto \sum_{i=0}^{n} b_i 2^i .$$

universal TM

Theorem: There exists a TM U such that

$$U(\underbrace{\langle M, \sigma \rangle}_{\text{binary encoding}}) = M(\sigma) \quad \forall \text{ TM } M, \ \forall \sigma \in \Sigma^*.$$

Existence of U implies that the language

$$L_{TM} = \{ \langle M, \sigma \rangle : M \text{ is a TM such that } M(\sigma) = 1 \}$$

is Turing-recognizable: $L_{TM} = L(U)$.

Halting problem

No decider TM that can compute it.

Theorem: L_{TM} is undecidable.

Proof: Suppose that H is a decider for L_{TM}. Then

$$H(\langle M, G\rangle) = \begin{cases} \perp & \text{if } M(G) = 1 \\ 0 & \text{if } M(G) \in \{0, \text{loop}\}. \end{cases}$$

Use H to construct another TM D:

On input $\langle M\rangle$ run $H(\langle M, M\rangle)$

- accept if H rejects. $\Leftrightarrow$ $M(G) = 1$.
- reject if H accepts. $\Leftrightarrow$ $M(G) \in \{0, \text{loop}\}$

We have a contradiction:

$$D(\langle D\rangle) = \begin{cases} \perp & D(\langle D\rangle) = 0 \\ 0 & D(\langle D\rangle) = 1. \end{cases}$$

Therefore H does not exist.

This is called the diagonalization argument:

	$\langle M_0\rangle$	$\langle M_1\rangle$	$\langle M_2\rangle$	$\dots$	$\langle D\rangle$
M_0	0	1	1	$\dots$	
M_1	0	1	0		
M_2	0	0	0		
$\vdots$				$\ddots$	
D	$\perp$	0	$\perp$	$\dots$	?

$\boxtimes$

Nondeterministic TM

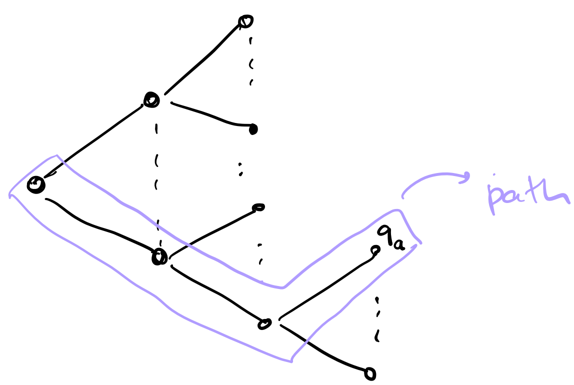
A nondeterministic TM N is a triple (Γ, Q, δ) where Γ, Q as before and

$$\delta : Q \times \Gamma \longrightarrow \underbrace{P}_{\text{collection of subsets}}(Q \times \Gamma \times \{-1, 0, +1\})$$

that is

$$\delta(q, s) = \{(\bar{q}_1, \bar{s}_1, \bar{\ell}_1), \dots, (\bar{q}_n, \bar{s}_n, \bar{\ell}_n)\}.$$

Computation branches out like a tree:



We write

$$N(\sigma) = \begin{cases} 1 & \text{if a path reaches } q_a \\ 0 & \text{if every path halts without reaching } q_a \end{cases}$$

Theorem: Computational power of nondeterministic TM is the same as the computational power of a deterministic TM.

(Both kinds compute the same class of functions.)

Circuits

Let $\mathcal{A}$ be a set of Boolean functions:

$$\mathcal{A} = \{ f_j : \mathbb{B}^{r_j} \longrightarrow \mathbb{B} \}.$$

A <u>circuit</u> C over $\mathcal{A}$ consists of

1) input variables $x_1, \dots, x_n$
2) auxilary variables $y_1, \dots, y_m$

$$y_j = f_j(u_1, \dots, u_{r_j})$$

where $f_j \in \mathcal{A}$ and

$$u_i = \begin{cases} x_1, \dots, x_n \\ \quad \text{or} \\ y_k \quad \text{where} \quad k < j. \end{cases}$$

y_m is the output of the circuit.

Representation as directed acyclic graph:

Directed graph $G = (V, E)$ consists of

V: a set of vertices, and

$E \subset V \times V$: a set of directed edges.

Acyclic means that there are no cycles.

The vertex set is partitioned as follows:

$$V = \underbrace{\{x_0, \dots, x_n\}}_{\text{sources}} \sqcup \underbrace{\{f_1, \dots, f_m\}}_{\text{gates}} \sqcup \underbrace{\{y_m\}}_{\text{sink}}$$

C computes $f: \mathbb{B}^n \longrightarrow \mathbb{B}$ if

$$\underbrace{C(a_1, \dots, a_n)}_{y_m} = f(a_1, \dots, a_n) \quad \forall a_i \in \mathbb{B}.$$

Ex: $\mathcal{A} = \{\wedge, \vee, \neg\}$

a) NOT : $\mathbb{B} \longrightarrow \mathbb{B}$, $x \longmapsto \neg x$

$$\begin{array}{ccc} 0 & \mapsto & 1 \\ 1 & & 0 \end{array}$$

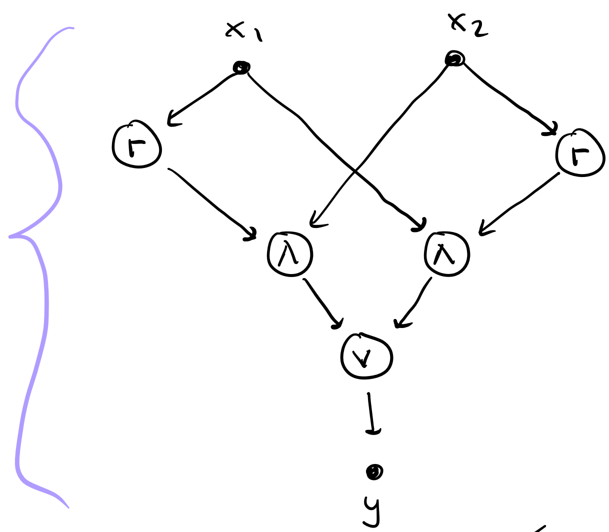
b) OR : $\mathbb{B}^2 \longrightarrow \mathbb{B}$, $(x, y) \longmapsto x \vee y$ (disjunction)

$$\begin{array}{ccc} 00 & & 0 \\ 01 & \mapsto & 1 \\ 10 & & 1 \\ 11 & & 1 \end{array}$$

c) AND : $\mathbb{B}^2 \longrightarrow \mathbb{B}$ $(x, y) \longmapsto x \wedge y$ (conjunction)

$$\begin{array}{ccc} 00 & & 0 \\ 01 & \mapsto & 0 \\ 10 & & 0 \\ 11 & & 1 \end{array}$$

C :



$$C(x_1, x_2) = XOR(x_1, x_2) = \begin{cases} 1 & x_1 \neq x_2 \\ 0 & x_1 = x_2 \end{cases}$$

Normal forms

A function of the form

$$f : \mathbb{B}^n \longrightarrow \mathbb{B}^m$$

$$f(x_1, \dots, x_n) = (y_1, \dots, y_m)$$

is called a <u>logic gate</u>.

Such a function amounts to a family of Boolean functions:

$$f_i : \mathbb{B}^n \longrightarrow \mathbb{B} \qquad i = 1, 2, \dots, m.$$

$$f_i(x_1, \dots, x_n) = y_i.$$

<u>D</u>isjunctive normal form (DNF):

Any Boolean function $f : \mathbb{B}^n \to \mathbb{B}$ can be written as a disjunction of conjunctions of literals (x_i or $\neg x_i$).

<u>Pro</u>of: Let

$$S = \{ u = (u_1, \dots, u_n) \in \mathbb{B}^n : f(u_1, \dots, u_n) = 1 \}.$$

Then we have

$$f(x_1, \dots, x_n) = \bigvee_{u \in S} \delta_u(x_1, \dots, x_n)$$

where

$$\delta_u(x) = \begin{cases} 1 & x_i = u_i \quad \forall i \\ 0 & \text{otherwise.} \end{cases}$$

Each δ_u can be written as

$$\delta_u(x_1, \dots, x_n) = \text{NOT}^{u_1}(x_1) \wedge \cdots \wedge \text{NOT}^{u_n}(x_n),$$

where

$$\text{NOT}^a(x) = \begin{cases} \neg x & a = 0 \\ x & a = 1 \end{cases} = \begin{cases} 1 & a = x \\ 0 & a \neq x. \end{cases}$$

$\boxtimes$

<u>Ex</u>: 1) NAND : $\mathbb{B}^2 \to \mathbb{B}$ $\quad (x_1, x_2) \mapsto \neg(x_1 \wedge x_2)$

$$S = \{ (0,0), (0,1), (1,0) \}$$

$$\text{NAND}(x_1, x_2) = (\neg x_1 \wedge \neg x_2) \vee (\neg x_1 \wedge x_2) \vee (x_1 \wedge \neg x_2)$$

2) XOR : $\mathbb{B}^2 \to \mathbb{B}$ $\quad (x_1, x_2) \mapsto x_1 \oplus x_2$

$$S = \{ (0,1), (1,0) \}$$

$$\text{XOR}(x_1, x_2) = (\neg x_1 \wedge x_2) \vee (x_1 \wedge \neg x_2).$$

<u>C</u>onjunctive normal form (CNF)

Let $f : \mathbb{B}^n \to \mathbb{B}$ be a Boolean function.
Write $g = \neg f$ in DNF

$$g(x_1, \ldots, x_n) = \bigvee_{u \in S} \delta_u (x_1, \ldots, x_n)$$

We will use the following De Morgan's identities

$$x \wedge y = \neg(\neg x \vee \neg y)$$
$$x \vee y = \neg(\neg x \wedge \neg y).$$

Applying $\neg$ to g we obtain

$$f(x_1, \ldots, x_n) = \neg g(x_1, \ldots, x_n)$$
$$= \neg \Big(\bigvee_{u \in S} \delta_u (x_1, \ldots, x_n) \Big)$$

$$= \Big(\bigwedge_{u \in S} \neg \underbrace{S_u(x_1, \dots, x_n)}_{\neg\left(\bigwedge_i \mathrm{NOT}^{u_i}(x_i)\right)} \Big)$$

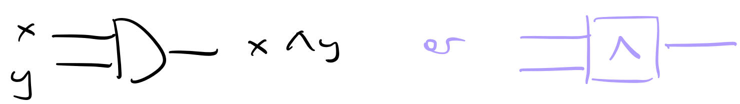
$$= \bigwedge_{u \in S} \bigvee_i \underbrace{\neg \, \mathrm{NOT}^{u_i}(x_i)}_{\text{literals}} \qquad \square$$

<u>Representing circuits using wires</u>

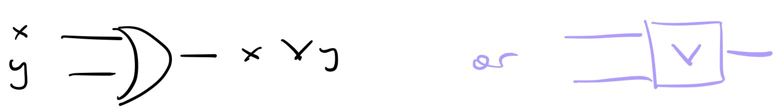
A circuit computing f can be represented using "wires" and "gates":

a) $\mathrm{NOT} : \mathbb{B} \longrightarrow \mathbb{B}$

b) $\mathrm{AND} : \mathbb{B}^2 \longrightarrow \mathbb{B}$



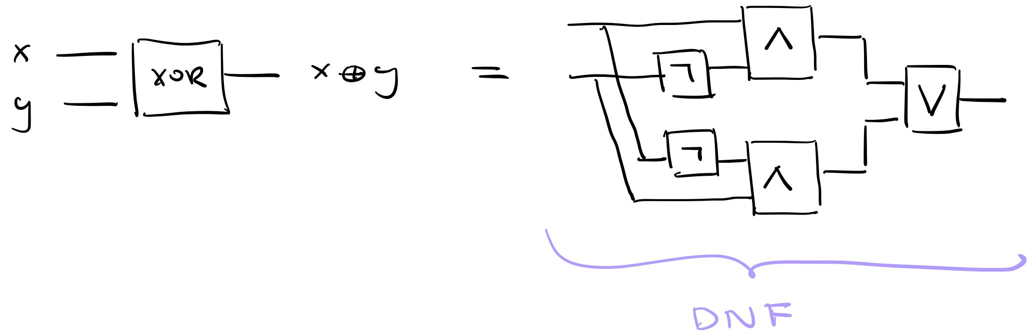
c) $\mathrm{OR} : \mathbb{B}^2 \longrightarrow \mathbb{B}$



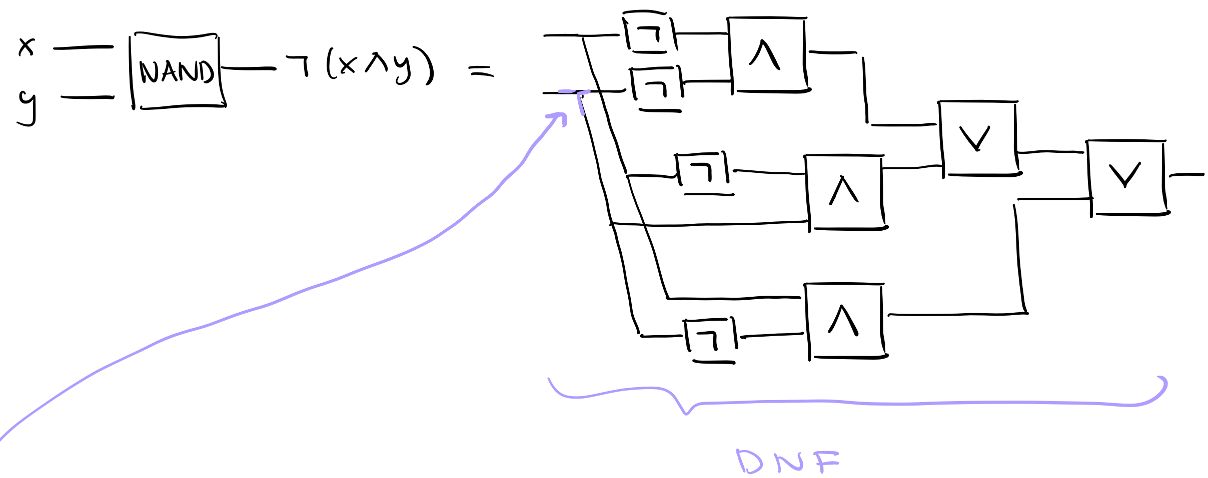
These basic gates can be composed to obtain more complicated circuits.

Ex :

1)

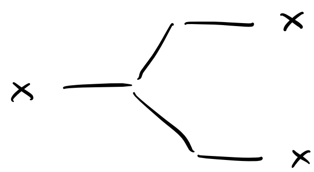


2)

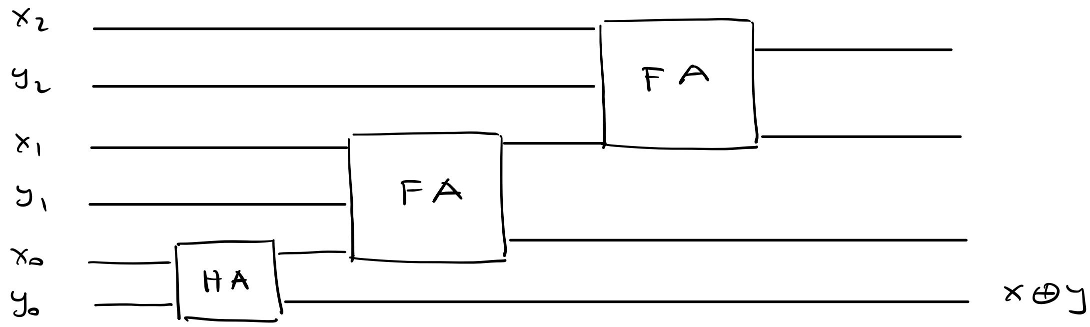


In the circuits we are using a new gate splitting a bit into two copies :

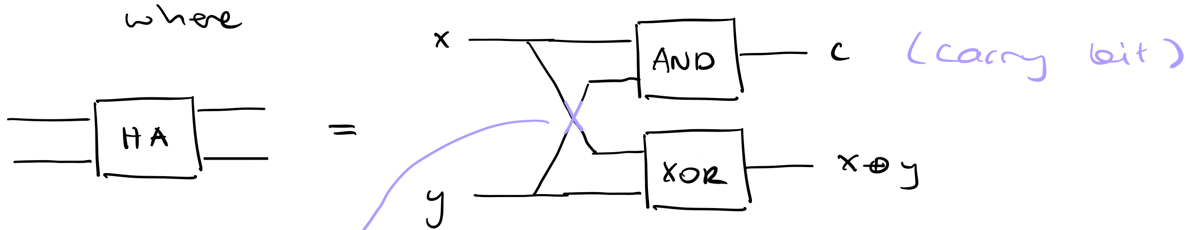
$$\text{FANOUT} : \mathbb{B} \longrightarrow \mathbb{B}^2 \qquad x \longmapsto (x, x)$$

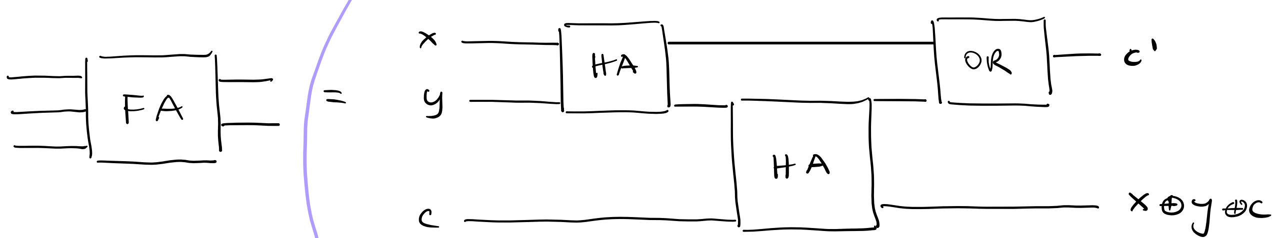


3) Adding $x = x_2 x_1 x_0$ and $y = y_2 y_1 y_0$:



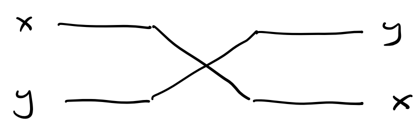
where





(SWAP)

CROSSOVER : $\mathbb{B}^2 \longrightarrow \mathbb{B}^2$



Universal gates

A set $\mathcal{A} = \{ f_j : \mathbb{B}^{n_j} \longrightarrow \mathbb{B} \}$ is called universal if any $f : \mathbb{B}^n \longrightarrow \mathbb{B}$ $(n \geq 1)$ can be computed using a circuit over $\mathcal{A}$.

Theorem: $\mathcal{A} = \{ \vee, \wedge, \neg \}$ is universal.

Proof: Given $f : \mathbb{B}^n \longrightarrow \mathbb{B}$ use DNF to write it as

$$f(x_1, \dots, x_n) = \bigvee_{u \in S} \delta_u(x_1, \dots, x_n)$$

and translate this into a circuit:

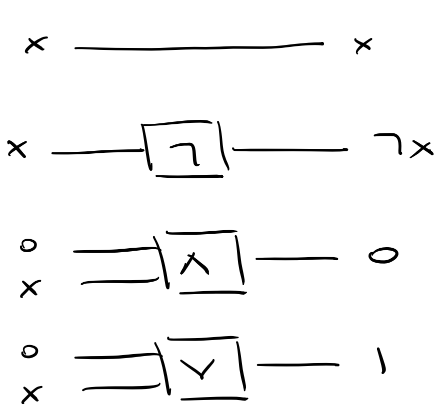
a) Let $f : \mathbb{B} \longrightarrow \mathbb{B}$ $(n = 1)$. There are 4 possibilities:

i) $0 \mapsto 0$, $1 \mapsto 1$:

ii) $0 \mapsto 1$, $1 \mapsto 0$:

iii) $0 \mapsto 0$, $1 \mapsto 0$

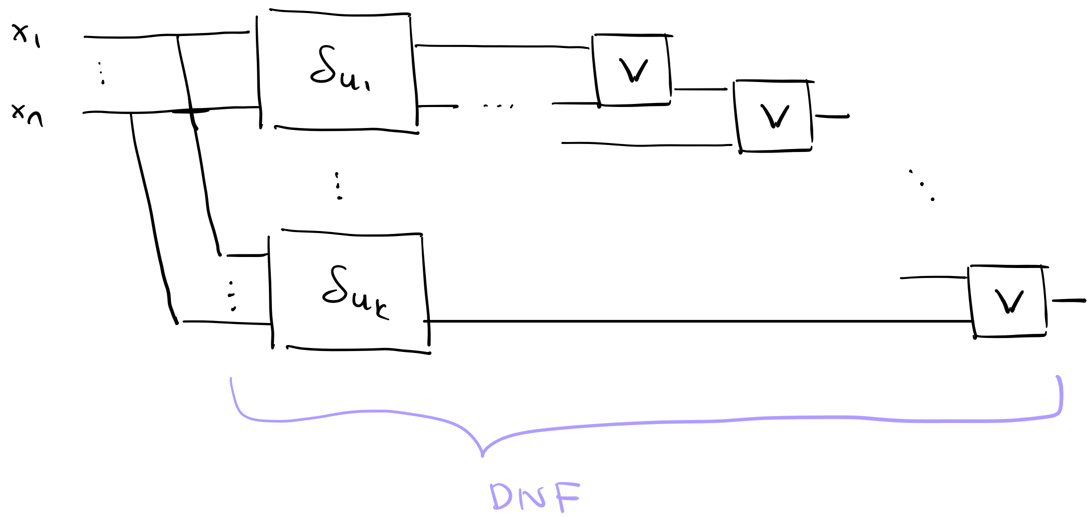
iv) $0 \mapsto 1$, $1 \mapsto 1$



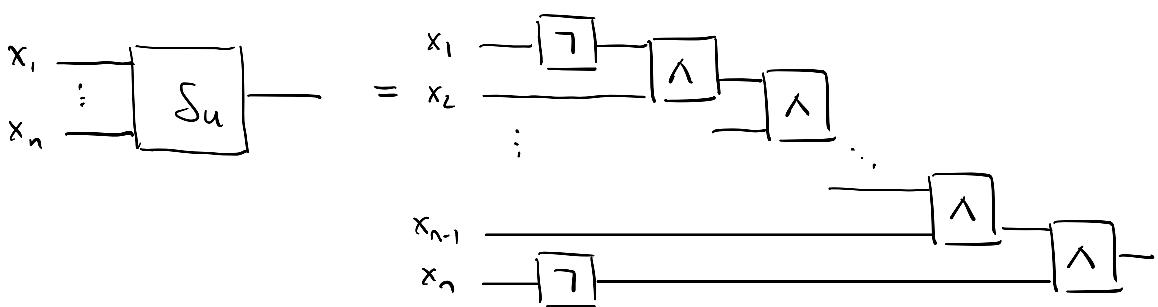
b) The circuit for an arbitrary

$$f : \mathbb{B}^n \longrightarrow \mathbb{B} \quad (n \geq 2)$$

looks like



For example $u = (0, 1, 1, \dots, 1, 1, 0)$



Therefore $\{\neg, \wedge, \vee\}$ is universal (provided that wires, ancilla bits, FANOUT's are available) $\blacksquare$

Other universal gates (wires, ancillas, FANOUT's provided)

1) $\mathcal{A} = \{ \neg, \wedge \}$ since

$$x_1 \vee x_2 = \neg (\neg x_1 \wedge \neg x_2).$$

2) $\mathcal{A} = \{ \neg, \vee \}$ since

$$x_1 \wedge x_2 = \neg (\neg x_1 \vee \neg x_2)$$

3) $\mathcal{A} = \{ \wedge, \oplus \}$

4) $\mathcal{A} = \{ \text{NAND} \}$ $\quad \text{NAND}(x,y) = \neg(x \wedge y)$

5) $\mathcal{A} = \{ \text{NOR} \}$ $\quad \text{NOR}(x,y) = \neg(x \vee y)$

HW: 3, 4, 5 are universal.

Turing machine vs Circuit model

A predicate $f : \mathbb{B}^* \to \mathbb{B}$ gives a sequence of Boolean functions

$$f_n : \mathbb{B}^n \to \mathbb{B}$$

$$f_n(x_1, \dots, x_n) = f(x_1 \cdots x_n) \qquad n = 1, 2, \dots$$

Each f_n can be computed by a circuit C_n.
Therefore $\{C_i\}_{i=1}^{\infty}$ can be used to compute f.
On the other hand, there are predicates that cannot be computed by a TM.

Uniform circuit family

A family of circuits $\{C_n\}_{n=1}^{\infty}$ is called uniform if there exists a TM M such that

$$\varphi_M : \mathbb{B}^* \longrightarrow \mathbb{B}^*$$ (function computed by M)

$$\varphi_M(\underbrace{\langle n \rangle}_{\text{bit string}}) = \underbrace{\langle C_n \rangle}_{\text{bit string}}$$

Theorem: The class of functions computable by a TM is the same as the class of functions computable by a uniform circuit family.

Church - Turing thesis

The class of functions computable by a TM is the same as the class of functions computable by an algorithm.

running on a physically realizable computation device

Complexity classes

Let f and g be functions $\mathbb{N} \to \mathbb{R}_{\geq 0}$.
We say $f(n)$ is in the class of functions $O(g(n))$ if there exists $c, n_0 \in \mathbb{N}$ such that

$$f(n) \leq c\, g(n) \quad \forall\, n \geq n_0.$$

For short we say $f(n)$ is $O(g(n))$ and write $f(n) = O(g(n))$.

Ex: 1) If $f(n)$ is a polynomial of degree k then $f(n)$ is $O(n^{\ell})$ for $\ell \geq k \geq 0$:

First observe that

$$n^k \leq n^{\ell} \quad \text{for} \quad n \geq 1.$$

Let $f(n) = \sum_{i=0}^{k} \alpha_i n^i$.

Then

$$f(n) \leq n^k \sum_{i=0}^{k} |\alpha_i| \quad \text{for} \quad n \geq 1.$$

2) $\log n$ is $O(n^k)$ for any $k > 0$.

HW: prove (2).

We say $f(n)$ is $\Omega(g(n))$ if there exists $c, n_0 \in \mathbb{N}$ such that

$$c\, g(n) \leq f(n) \quad \forall\, n \geq n_0.$$

We say $f(n)$ is $\Theta(g(n))$ if $f(n)$ is both $O(g(n))$ and $\Omega(g(n))$.

Let M be a decider TM.
The time complexity of M is the function

$$t_M : \mathbb{N} \longrightarrow \mathbb{N}$$

defined by

$t_M(n)$: the maximum number of steps M performs on any input of length n.

We say M is a $t_M(n)$ time TM.
Let N be a decider non-deterministic TM.
Time complexity of N:

$$t_N : \mathbb{N} \longrightarrow \mathbb{N}$$

where

$t_N(n)$: the maximum number of steps N performs on any branch of its computation on any input of length n.

We say N is a $t_N(n)$ time non-determinstic TM.

Let $t : \mathbb{N} \longrightarrow \mathbb{R}_{\geq 0}$ be a function.
The time complexity class $\mathrm{Time}(t(n))$ is defined by

$$\mathrm{Time}(t(n)) = \{ L : L \text{ is a language decided by an } O(t(n)) \text{ time deterministic TM} \}$$

The non-deterministic time complexity class:

$$\mathrm{NTime}(t(n)) = \{ L : L \text{ is decided on a } O(t(n)) \text{ time nondeterministic TM} \}$$

P vs NP

P is the class of languages decidable in polynomial time on a deterministic TM:

$$P = \bigcup_{k \geq N} \mathrm{Time}(n^k).$$

Nondeterministic polynomial time complexity:

$$NP = \bigcup_{k \geq N} \mathrm{NTime}(n^k).$$

Unsolved problem: Is $P = NP$?
(Common belief is that $P \neq NP$)

A verifier for a language $L \subset \Sigma^*$ is a TM V such that

$$L = \{ G : V \text{ accepts } \langle G, w \rangle \text{ for some } w \in \Sigma^* \}$$

witness

Time complexity of V is measured as a function of the length of G.

<u>Theorem</u>: $L \in NP \iff L$ has a polynomial time verifier.

<u>Proof</u>:

The sequence of non-deterministic choices made by an accepting computational branch can be seen as a witness, and vice versa. $\square$

Problems that are in P:

Graphs can be encoded as a list of vertices and edges. Another way is to use the adjacency matrix. We assume that the size of the encoding is polynomial in the number of vertices.

1) PATH(G) = { <G, s, t> : G = (V, E) is a directed graph that has a directed path from vertex s to vertex t }.

Algorithm: Repeat the following until no new vertices are marked:

Scan all the edges of G.

For an edge (a, b), mark b if a is marked.

This step runs |V| times.

2) RELPRIME = { <x, y> : x and y relatively prime }

$\gcd(x,y) = 1$

Algorithm: Use Euclid's algorithm to find the greatest common divisor:

$$x = q_0 y + r_0$$
$$y = q_1 r_0 + r_1$$
$$r_0 = q_2 r_1 + r_2$$
$$\cdots$$
$$r_{N-2} = q_N r_{N-1} + r_N$$
$$r_{N-1} = \gcd(x,y).$$

HW: This step runs $\min\{2\log x, 2\log y\}$ times.

Problems in NP:

1) A Hamiltonian path in a directed graph is a directed path that goes through each vertex exactly once.

HAMPATH = $\{\langle G, s, t\rangle$: G is a directed graph with a Hamiltonian path from s to $t\}$

2) FACTORING = $\{\langle x, \ell\rangle : \ell < x$ and $\exists\, 1 < k < \ell$ such that k divides $x\}$

Verification can be done in polynomial time by long division.

3) A Boolean formula is an expression ϕ involving Boolean variables $\{x_i\}$ and operations $\{\wedge, \vee, \neg\}$.

E.g. $\phi = (\neg x_1 \wedge x_2) \vee (x_1 \wedge \neg x_3)$.

A Boolean formula is a circuit where each auxiliary variable, except the last one, is used exactly once.

A Boolean formula is satisfiable if there exists $(a_1, \ldots, a_n) \in \mathbb{B}^n$ such that

$$\phi(a_1, \ldots, a_n) = 1.$$

SAT = $\{\langle\phi\rangle : \phi$ is a satisfiable Boolean formula$\}$

Theorem [Cook – Levin]

$$SAT \in P \iff P = NP.$$

Reducibility

or efficiently reducible

A language L_1 is (polynomial time) reducible to another language L_2 if there exists a polynomial time TM M such that $\varphi_M : \Sigma^* \longrightarrow \Sigma^*$ is a function and

$$G \in L_1 \iff \varphi_M(G) \in L_2$$

$t_M(n) = O(n^k)$

A language $L \in NP$ is NP-complete if any other language in NP is reducible to L.

Restatement of Cook – Levin theorem:

SAT is NP-complete.

Also HAMPATH is NP-complete.

BPP

A probabilistic TM is a nondeterministic TM $M = (\Gamma, Q, \delta)$ such that

- M either accepts or rejects:
$$M(\sigma) \in \{0,1\}$$
- $$\delta(q,\sigma) = \underbrace{\{ \delta_0(q,\sigma) , \delta_1(q,\sigma)\}}_{\in P(Q \times \Gamma \times \{\pm 1, 0\})}$$

and at each step of computation a fair coin flip decides whether to apply δ_0 or δ_1.

We define the probability that M accepts σ:

$$p(M(\sigma)=1) = \sum_b p(b)$$

where b runs over accepting branches and

$$p(b) = 2^{-k}, \quad k: \text{number of coin flips.}$$

Probability that M rejects:

$$p(M(\sigma)=0) = 1 - p(M(\sigma)=1).$$

Given a language L, let $P_L : \Sigma^* \longrightarrow \mathbb{B}$ denote the predicate defined by

$$P_L(\sigma) = \begin{cases} 1 & \sigma \in L \\ 0 & \sigma \notin L. \end{cases}$$

We define success probability / error probability

- $p_{succ}(M)$: probability that $M(\sigma) = P_L(\sigma)$
- $p_{err}(M) = 1 - p_{succ}(M)$.

M <u>decides</u> L with error probability $\epsilon < 1/2$ if

$$p_{err}(M) \leq \epsilon.$$

<u>BPP</u> is the class of languages decided by a probabilistic polynomial time TM with error probability of $1/3$. → regarded as nondeterministic TM

bounded-error probabilistic polynomial time

By definition $P \subset BPP$. However, the relationship to NP is not known.

<u>Ex</u>: $PRIMES = \{ \langle n \rangle : n \text{ is prime} \}$

Recent progress: $PRIMES \in P$ (2002)

<u>Amplification lemma</u>

Let M be a probabilistic polynomial time TM that decides L with error probability $\epsilon_M < 1/2$. Given a polynomial $p(n)$, there exists a probabilistic polynomial time TM N that decides L with error probability

$$\epsilon_N \leq 2^{-p(n)}.$$

Strong Church-Turing thesis

Any model of computation can be simulated on a probabilistic TM with at most a polynomial increase in the number of steps.

efficient simulation

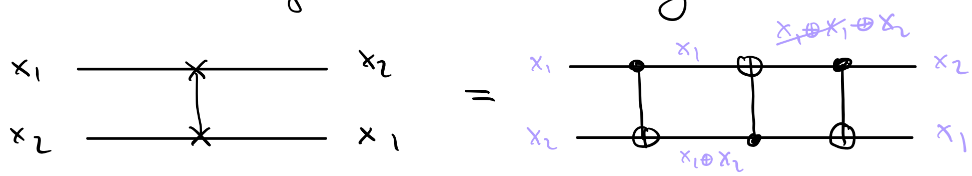
Reversible computation

A logic gate $f: \mathbb{B}^n \longrightarrow \mathbb{B}^n$ is called <u>reversible</u> if f is a permutation.

Ex: 1) NOT : $\mathbb{B} \longrightarrow \mathbb{B}$ $\quad x \longmapsto x+1$

2) CNOT : $\mathbb{B}^2 \longrightarrow \mathbb{B}^2$ $\quad (x_1, x_2) \longmapsto (x_1, x_2 \oplus x_1)$

Another useful reversible gate:

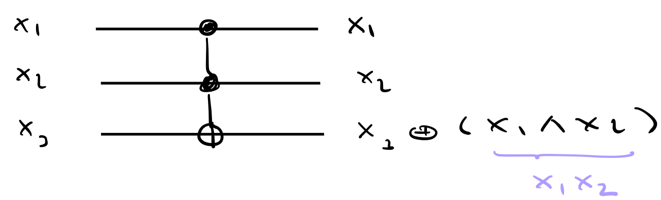


This is called SWAP (or CROSSOVER).

3) Toffoli gate:

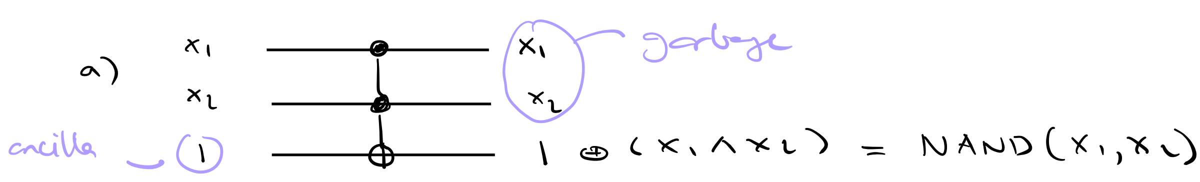
CCNOT : $\mathbb{B}^3 \longrightarrow \mathbb{B}^3$

HW: Reversible gates of the form $\mathbb{B}^2 \to \mathbb{B}^2$ are not universal.

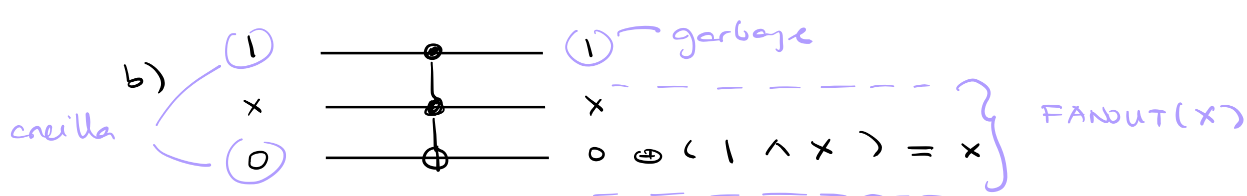


Using the Toffoli gate we can implement:

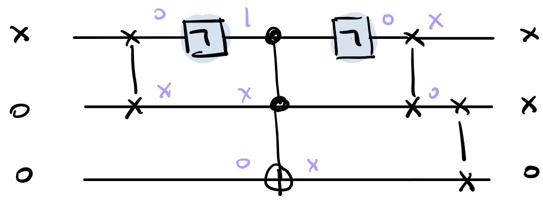
a)



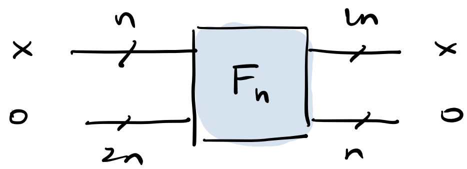
b)

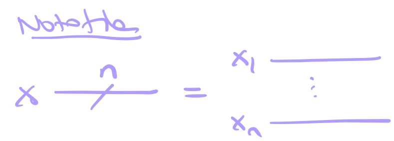


Using NOT & CNOT we can implement



In general, we can implement





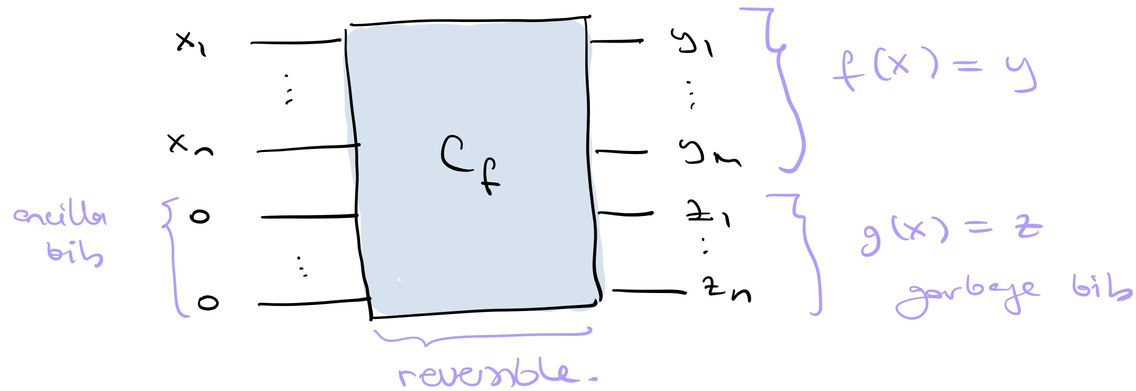
Note that if ancilla bits are used and garbage bits are ignored the universal gate set {NAND, FANOUT} can be implemented using the Toffoli gates.

Pro : Any logic gate $f : \mathbb{B}^n \to \mathbb{B}^m$ can be implemented as a circuit over {NOT, CNOT, CCNOT} which after ignoring the ancilla and garbage bits (both given by a sequence of 0's) corresponds to the reversible gate:

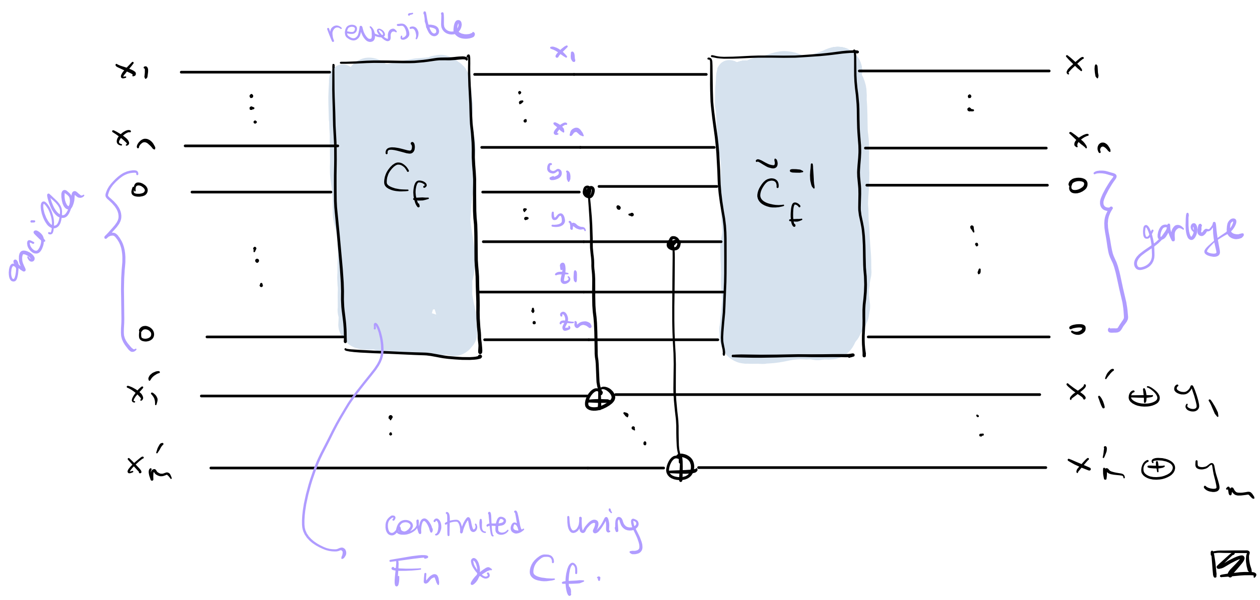
$$\tilde{f} : \mathbb{B}^{n+m} \longrightarrow \mathbb{B}^{n+m}$$

$$\tilde{f}(x, x') = (x, x' \oplus f(x)).$$

Proof: First express f as a circuit over $\{\text{NAND}, \text{FANOUT}\}$. Then replace each elementary gate with its implementation using the Toffoli gate. Using $\{\text{NOT}, \text{CNOT}\}$ we have



We can arrange the garbage bits to be independent of the input x:



$\boxtimes$

HW: Count the ancillas needed & extra gates to implement a circuit in a reversible way.

Quantum computation

<u>Lem</u>: Any $U \in U(\mathbb{C}^2)$ can be written as

$$U = e^{i\alpha} R_z(\beta) R_y(\gamma) R_z(\delta).$$

<u>Proof</u>: Recall that we can write

$$U = e^{i\alpha} \underbrace{\begin{pmatrix} a & b \\ c & d \end{pmatrix}}_{\text{determinant} = 1}.$$

The condition $U^{-1} = U^{+}$ implies that

$$\begin{pmatrix} d & -b \\ -c & a \end{pmatrix} = \begin{pmatrix} \bar{a} & \bar{c} \\ \bar{b} & \bar{d} \end{pmatrix},$$

i.e., $d = \bar{a}$ and $b = -\bar{c}$.

This gives

$$U = e^{i\alpha} \underbrace{\begin{pmatrix} a & -\bar{c} \\ c & \bar{a} \end{pmatrix}}_{\det = 1}, \qquad \underbrace{|a|^2 + |c|^2}_{} = 1.$$

We can write $|a| = \cos \gamma/2$ and $|c| = \sin \gamma/2$ for some $\gamma \in [0, 2\pi)$.

Then

$$U = e^{i\alpha} \begin{pmatrix} e^{i\theta} \cos \gamma/2 & -e^{-i\varphi} \sin \gamma/2 \\ e^{i\varphi} \sin \gamma/2 & e^{-i\theta} \cos \gamma/2 \end{pmatrix}$$

Choosing β, δ such that

$$\theta = -(\beta + \delta)/2 \quad \text{and} \quad \varphi = (\beta - \delta)/2$$

we obtain

$$U = e^{i\alpha} \begin{pmatrix} e^{-i\beta/2} & 0 \\ 0 & e^{i\beta/2} \end{pmatrix} \begin{pmatrix} \cos \gamma/2 & -\sin \gamma/2 \\ \sin \gamma/2 & \cos \gamma/2 \end{pmatrix} \begin{pmatrix} e^{-i\delta/2} & 0 \\ 0 & e^{i\delta/2} \end{pmatrix}$$

$\blacksquare$

Cor: $U = e^{i\alpha} AXBXC$ where $A, B, C \in U(\mathbb{C}^2)$ such that $ABC = \mathbb{1}$.

Proof: Let

$$A = R_z(\beta)\, R_y(\gamma/2)$$
$$B = R_y(-\gamma/2)\, R_z(-(\delta+\beta)/2)$$
$$C = R_z((\delta-\beta)/2).$$

Then $ABC = \mathbb{1}$ and

$$AXBXC = A(X R_y(-\gamma/2) \overset{XX}{\vee} R_z(-(\delta+\beta)/2) X) C$$
$$= A\, R_y(\gamma/2)\, R_z((\delta+\beta)/2)\, C$$
$$= R_z(\beta)\, R_y(\gamma)\, R_z(\delta)$$

HW: $X R_z(\theta) X = R_z(-\theta)$ and $X R_y(\theta) X = R_y(-\theta)$

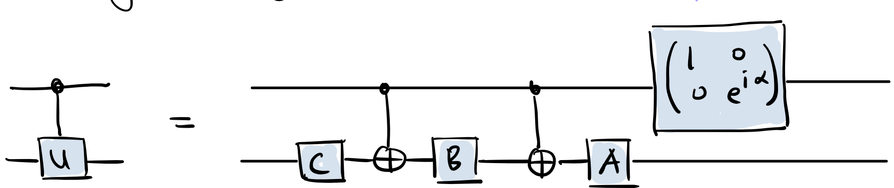
$\blacksquare$

Using this result we can implement

$$C(U) : (\mathbb{C}^2)^{\otimes 2} \longrightarrow (\mathbb{C}^2)^{\otimes 2}$$
$$|ct\rangle \longmapsto |c\rangle U^c |t\rangle$$

as the following circuit:

$$|c\rangle e^{ic\alpha} |t\rangle = \begin{cases} |0\rangle|t\rangle \\ e^{i\alpha}|1\rangle|t\rangle \end{cases}$$



$$\underbrace{AX^cBX^cC}_{} = \begin{cases} \mathbb{1} & c=0 \\ e^{i\alpha} U & c=1 \end{cases}$$

That is, $C(U)$ can be implemented by single unitaries and CNOT's.

c_J

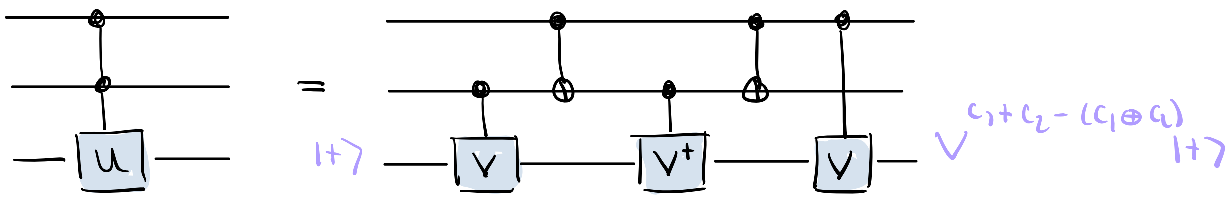
teknik

A two-qubit controlled unitary

$$C^2(U) : (\mathbb{C}^2)^{\otimes 2} \otimes \mathbb{C}^2 \longrightarrow (\mathbb{C}^2)^{\otimes 2} \otimes \mathbb{C}^2$$

$$|c_1 c_2\rangle\, |t\rangle \longmapsto |c_1 c_2\rangle\, U^{c_1 \cdot c_2} |t\rangle$$

can be implemented as follows:



where $V \in U(\mathbb{C}^2)$ such that $V^2 = U$.

<u>Ex</u>: When $U = X$ (Toffoli gate) use

$$V = (1 - i)\, \frac{\mathbb{1} + iX}{2}.$$

<u>Lem</u>: A multi-qubit controlled unitary

$$C^n(U) : (\mathbb{C}^2)^{\otimes n+1} \longrightarrow (\mathbb{C}^2)^{\otimes n+1}$$

$$|c_1 \cdots c_n\rangle\, |t\rangle \longmapsto |c_1 \cdots c_n\rangle\, U^{c_1 \cdots c_n} |t\rangle$$

can be implemented using $O(n)$ unitaries in $U(\mathbb{C}^2) \cup \{CNOT\}$.

<u>HW</u>: Prove this. Fig 4.10 in NC.

We write $C^n_i(U)$ if the target is the i-th qubit.

Universal quantum gates

Let $U, U' \in U(V)$.

We define the <u>error</u> when U' is implemented instead of U as follows:

$$E(U,U') = \max_{\substack{v \in V: \\ \|v\|=1}} \|(U-U')v\|.$$

<u>Lem</u>: Let $U_i, U_i' \in U(V)$ where $i=1,..,m$.
Then

$$E(U_m \cdots U_1, U_m' \cdots U_1') \leq \sum_{i=1}^{m} E(U_i, U_i').$$

<u>Proof</u>: It suffices to prove the result for $m=2$:

$$E(U_2U_1, U_2'U_1') = \max_{v:\|v\|=1} \{\|(U_2U_1 - U_2'U_1')v\|\}$$

$$= \|(U_2U_1 - U_2'U_1')u\| \quad \text{for some } u.$$

(exists by compactness of $P(\mathbb{C}^2) = S^2$) (closed + bounded)

$$= \|(U_2-U_2')U_1 u + U_2'(U_1-U_1')u\|$$

$$\leq \|(U_2-U_2)\underbrace{U_2 u}_{\text{unit vector}}\| + \underbrace{\|\underbrace{U_2'}_{\text{unitary matrix}}(U_1-U_1')u\|}_{\|(U_1-U_1')u\|}$$

triangle ineq.

$$\leq E(U_2, U_2') + E(U_1, U_1').$$

The general case follows by induction. $\boxtimes$

Let $SU((\mathbb{C}^2)^{\otimes n}) = \{U \in U((\mathbb{C}^2)^{\otimes n}) : \det U = 1\}$.

Theorem: $\mathcal{A}_Q = \{H, T, CNOT\}$ is universal for quantum computation:

Given $U \in SU((\mathbb{C}^2)^{\otimes n})$ and $\epsilon > 0$ there exists a circuit over $\mathcal{A}_Q$ that implements $U' \in SU((\mathbb{C}^2)^{\otimes n})$ such that

$$E(U, U') < \epsilon.$$

A unitary $U \in U(\mathbb{C}^d)$ is called <u>two-level</u> if

$$|\{ j : 1 \leq j \leq d \text{ and } U e_j = e_j \}| \geq d-2.$$

Lem: Let $U \in U(\mathbb{C}^d)$
There exists a set $\{U_i \in U(\mathbb{C}^d)\}_{i=1}^n$ of two-level unitaries such that

$$U = U_1 U_2 \cdots U_n.$$

Proof: For $d \leq 2$ the claim holds.
Assume $d \geq 3$ and write

$$U = \begin{pmatrix} a_{11} & \cdots & a_{1d} \\ \vdots & & \vdots \\ a_{d1} & \cdots & a_{dd} \end{pmatrix}$$

Claim: There exists unitaries $V_1, \dots, V_{d-1}$ such that

$$(V_{k-1} \cdots V_1 U)_{i1} = 0 \quad \text{for } 2 \leq i \leq k:$$

Suppose $V_1, \dots, V_k$ are constructed such that

$$(V_{k-1} \cdots V_1 U) = \begin{pmatrix} b_{11} & \cdots & b_{d1} \\ 0 & & \\ \vdots & & \vdots \\ 0 & & \\ b_{k+1,1} & & \\ \vdots & & \\ b_{d1} & \cdots & b_{dd} \end{pmatrix}.$$

If $b_{k+1,1} \neq 0$ then let

$$V_k = \begin{array}{cc} \begin{pmatrix} \bar{c} & 0 & \cdots & 0 & \bar{d} & 0 & \cdots & 0 \\ 0 & 1 & & & 0 & & & \vdots \\ \vdots & & \ddots & & \vdots & & & \\ 0 & & & 1 & 0 & & & \\ d & \cdots & & 0 & -c & 0 & & 0 \\ 0 & & & & 0 & 1 & & \vdots \\ \vdots & & & & \vdots & & \ddots & \\ 0 & \cdots & & & 0 & & \cdots & 1 \end{pmatrix} & \begin{matrix} \\ \\ \\ \\ - k+1 \\ \\ \\ \\ \end{matrix} \\ \begin{matrix} \qquad\qquad\qquad\qquad | \\ \qquad\qquad\qquad\qquad k+1 \end{matrix} & \end{array}$$

where

$$c = \frac{b_{11}}{\sqrt{|b_{11}|^2 + |b_{k+1,1}|^2}} \qquad d = \frac{b_{k+1,1}}{\sqrt{|b_{11}|^2 + |b_{k+1,1}|^2}}.$$

Otherwise, let $V_k = \mathbb{1}$.

Then

$$(V_k \cdots V_1 U)_{k+1,1} = d\, b_{1,1} - c\, b_{k+1,1} = 0.$$

Apply this to the remaining columns:

There exists two level unitaries $\{V_{i,j}\}$ such that

$$(V_{1,d-1}) \cdots (V_{d-1,1} \cdots V_{1,1})\, U = \mathbb{1}_{\mathbb{C}^d}.$$

Therefore

$$U = (V_{1,1}^{+} \cdots V_{d-1,1}^{+}) \cdots (V_{1,d-1}^{+}) \qquad \boxed{\checkmark}$$

<u>HW</u>: show V^{+} is two-level unitary if V is.

Note that the number of two-level unitary in the decomposition for U is at most

$$(d-1)+(d-2)+\cdots+1=\underbrace{d(d-1)/2}_{O(d^2)}.$$

Given bit strings σ and σ' of length n, the Hamming distance between σ and σ' is defined by

$$H(\sigma,\sigma')=\left|\{i : \sigma_i\neq\sigma'_i\}\right|.$$

Given a reversible logic gate $f:\mathbb{B}^n\longrightarrow\mathbb{B}^n$ we will write

$$V_f:(\mathbb{C}^2)^{\otimes n}\longrightarrow(\mathbb{C}^2)^{\otimes n}$$

for the unitary defined by

$$V_f|\sigma\rangle=|f(\sigma)\rangle$$

Implementing two-level unitaries

Let $U \in U((\mathbb{C}^2)^{\otimes n})$ be a two-level unitary satisfying

$$U|a_1 \cdots a_n\rangle = |a_1 \cdots a_n\rangle$$

for all basis vectors other than

$$|s_1 \cdots s_n\rangle \quad \text{and} \quad |t_1 \cdots t_n\rangle.$$

Consider a sequence $\{g_i\}_{i=1}^{m}$ of bit strings of length n such that

- $g_1 = s_1 \cdots s_n$
- $H(g_i, g_{i+1}) = 1 \quad \forall i$
- $g_m = t_1 \cdots t_n$.

Define $f_i : \mathbb{B}^n \to \mathbb{B}^n$ by

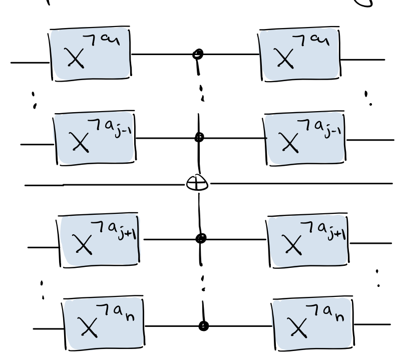
$$f_i(a_1 \cdots a_n) = \begin{cases} g_{i+1} & a_1 \cdots a_n = g_i \\ g_i & a_1 \cdots a_n = g_{i+1} \\ a_1 \cdots a_n & \text{otherwise.} \end{cases}$$

Then let

$$V = V_{f_{m-2}} \cdots V_{f_2} V_{f_1}.$$

Each V_{f_i} can be implemented as follows:

$C(a_1 \cdots a_n ; j) =$



where $g_i = a_1 \cdots a_n$ and $g_{i+1} = a_1 \cdots (\neg a_j) \cdots a_n$.

can there be

We have

* more vectors on the right

$$U|s_1 \dots s_n\rangle = \alpha |s_1 \dots s_n\rangle + \beta |t_1 \dots t_n\rangle$$
$$U|t_1 \dots t_n\rangle = \vartheta |s_1 \dots s_n\rangle + \gamma |t_1 \dots t_n\rangle .$$

Let b_i denote the bit in G_{m-2} that is different than G_{m-1}.

Then define $\tilde{U} \in U(\mathbb{C}^2)$:

$$\tilde{U}|b_i\rangle = \alpha |b_i\rangle + \beta |\neg b_i\rangle$$
$$\tilde{U}|\neg b_i\rangle = \vartheta |b_i\rangle + \gamma |\neg b_i\rangle .$$

Let $C_i^n(\tilde{U})$ denote the controlled gate with target the i-th qubit.

Then we have

$$U = V^{+} C_i^n(\tilde{U}) V.$$

<u>Pro</u> : A two-level unitary $U \in U((\mathbb{C}^2)^{\otimes n})$ can be implemented using $O(n^2)$ gates in $U(\mathbb{C}^2) \sqcup \{CNOT\}$.

<u>HW</u> : Verify the count.

A unitary $U \in U((\mathbb{C}^2)^{\otimes n})$ can be implemented using $O(n^2 4^n)$ gates in $U(\mathbb{C}^2) \sqcup \{CNOT\}$.

" $O(n^2 4^n) = O(n^2)\, O((2^n)^2)$ "

Let $\overline{\alpha} = \alpha - \lfloor \alpha \rfloor$ where $\lfloor \alpha \rfloor$ largest integer less than or equal to α.

Given $\epsilon > 0$ chose $n \in \mathbb{N}$ s.t. $1/n < \epsilon$.

Then $\exists\, i, j, r \in \mathbb{N}$:

$$\left| \underbrace{\overline{i\alpha} - \overline{j\alpha}}_{\underbrace{(i-j)}_{p}\alpha + \underbrace{\lfloor j\alpha \rfloor - \lfloor i\alpha \rfloor}_{q}} \right| < 1/n \qquad \underbrace{1 \leq i, j \leq n+1}_{\text{This is by the Pigeonhole principle:}}$$

n+1 numbers in
[0,1) & n intervals

Approximating single qubit unitaries

Pro: Let $\vartheta \in [0,1)$ be irrational.
Then $\{ e^{2\pi i \vartheta n} : n \in \mathbb{N} \}$ is dense in $U(\mathbb{C})$.

Proof: Given $\epsilon > 0$ there exists $n, m \in \mathbb{Z}$ such that

$$|n\vartheta - m| < \epsilon.$$ (Dirichlet approx. thm)

That is, the distance between $e^{2\pi i \vartheta n}$ and $e^{2\pi i m} = 1$ is less than ϵ.
Since ϑ is irrational we have $e^{2\pi i \vartheta n} \neq 1$.
Then given $\varphi \in [0,1)$ there exists $k \in \mathbb{N}$ such that the distance between $e^{2\pi i \varphi}$ and $e^{2\pi i \vartheta n k}$ is less than ϵ. $\boxtimes$

We will apply this result to the group of rotations in $\mathbb{R}^2$, since this group can be identified with $U(\mathbb{C})$.
Recall that

$$R_{\hat{n}}(\vartheta) = e^{-i \vartheta \hat{n} \cdot \hat{\sigma}} = \cos\frac{\vartheta}{2} \mathbb{1} - \sin\frac{\vartheta}{2} (\hat{n} \cdot \hat{\sigma}).$$

Let $SU(\mathbb{C}^d) = \{ U \in U(\mathbb{C}^d) : \det U = 1 \}$

Lem: Let $\hat{n}$ and $\hat{m}$ be orthogonal unit vectors in $\mathbb{R}^3$. Then any $U \in SU(\mathbb{C}^2)$ can be written as

$$U = R_{\hat{n}}(\beta)\, R_{\hat{m}}(\gamma)\, R_{\hat{n}}(\delta)$$

for some $\beta, \gamma, \delta \in [0, 2\pi)$.

Proof: We proved this result for $\hat{n} = \hat{z}$ and $\hat{m} = \hat{y}$.

There exists a rotation that maps $\hat{n}$ to $\hat{z}$ and $\hat{m}$ to $\hat{y}$. Let us write $Rot_{\hat{k}}(\theta)$ for this rotation.

Observe that

$$R_{\hat{k}}(\theta)\, R_{\hat{n}}(\beta)\, R_{\hat{k}}(\theta)^{\dagger}$$

$$= R_{\hat{k}}(\theta)\left(\cos\frac{\beta}{2}\,\mathbb{1} - i\sin\frac{\beta}{2}\,(\hat{n}\cdot\hat{\sigma})\right)R_{\hat{k}}(\theta)^{\dagger}$$

$$= \cos\frac{\beta}{2}\,\mathbb{1} - i\sin\frac{\beta}{2}\,\Big(\underbrace{Rot_{\hat{k}}(\theta)(\hat{n})}_{\hat{z}}\cdot\hat{\sigma}\Big).$$

$$= R_{\hat{z}}(\beta).$$

Then given U first express as

$$R_{\hat{k}}(\theta)^{\dagger}\, U\, R_{\hat{k}}(\theta) = R_{\hat{z}}(\beta)\, R_{\hat{y}}(\gamma)\, R_{\hat{z}}(\delta)$$

which gives

$$U = R_{\hat{k}}(\theta)\left(R_{\hat{z}}(\beta)\, R_{\hat{y}}(\gamma)\, R_{\hat{z}}(\delta)\right) R_{\hat{k}}(\theta)^{\dagger}$$

$$= R_{\hat{n}}(\beta)\, R_{\hat{m}}(\gamma)\, R_{\hat{n}}(\delta) \qquad \blacksquare$$

Lem: Given $U \in SU(\mathbb{C}^2)$ and $\epsilon > 0$ there exists a unitary V given by a product of H & T gates such that $E(U, V) < \epsilon$.

Proof: Let $\hat{n}$ and $\hat{m}$ be two orthogonal unit vectors in $\mathbb{R}^3$.

Then any $U \in U(\mathbb{C}^2)$ can be expressed as

$$U = R_{\hat{n}}(\beta)\, R_{\hat{m}}(\gamma)\, R_{\hat{n}}(\delta).$$ (HW)

Claim: We have

$$T^{-1} H T H = e^{-i\vartheta \hat{n}\cdot\hat{\sigma}}$$

where
$$\hat{n} = \frac{1}{\sqrt{1+\cos^2 \pi/8}} (\cos \pi/8, -\sin \pi/8, -\cos \pi/8)$$

ϑ is an irrational multiple of π:

We begin by computing

$$T^{-1}(H T H) = e^{-i\pi/8} R_z(-\pi/4)\, e^{i\pi/8} R_x(\pi/4)$$

$$= (\cos \pi/8\, \mathbb{1} + i \sin \pi/8\, Z)(\cos \pi/8\, \mathbb{1} - i \sin \pi/8\, X)$$

$$= \underbrace{\cos^2 \pi/8\, \mathbb{1}}_{\substack{\cos \vartheta/2 \\ \text{for some } \vartheta.}} - i \underbrace{\left[\cos \pi/8\, (X - Z) - \sin \pi/8\, Y\right]}_{(\cos \pi/8,\, -\sin \pi/8,\, -\cos \pi/8)\cdot\hat{\sigma}} \sin \pi/8$$

$$= \cos \vartheta/2\, \mathbb{1} - i\, \hat{\lambda}\cdot\hat{\sigma} \sin \pi/8$$

where $\hat{\lambda} = (\cos \pi/8, -\sin \pi/8, -\cos \pi/8)$ and $\vartheta \in [0, 2\pi)$ is such that $\cos \vartheta/2 = \cos^2 \pi/8$.

double angle: $\cos^2 x = (1+\cos 2x)/2$

We have

$$\cos\frac{\theta}{2} = \cos^2\frac{\pi}{8} = \frac{1+\cos\pi/4}{2} = \frac{1+1/\sqrt{2}}{2} = \frac{\sqrt{2}+1}{2\sqrt{2}}$$

and

$$|\hat{\alpha}| = \sqrt{\cos^2\pi/8 + \sin^2\pi/8 + \cos^2\pi/8}$$
$$= \sqrt{1+\cos^2\pi/8} = \sqrt{\frac{3\sqrt{2}+1}{2\sqrt{2}}}.$$

Note that

$$\sin\frac{\pi}{8}\,|\hat{\alpha}| = \sqrt{\frac{4-2\sqrt{2}}{8}}\sqrt{\frac{3\sqrt{2}+1}{2\sqrt{2}}} = \frac{1}{2\sqrt{2}}\sqrt{\frac{10\sqrt{2}-8}{2\sqrt{2}}}$$
$$= \frac{\sqrt{5-2\sqrt{2}}}{2\sqrt{2}} = \sqrt{1-\frac{3+2\sqrt{2}}{8}} = \sqrt{1-\cos^2\frac{\theta}{2}} = \sin\frac{\theta}{2}.$$

Therefore

$$T^{-1}(HTH) = \cos\theta/2\ \mathbb{1} - i\ \hat{\alpha}\cdot\hat{\sigma}\ \sin\pi/8$$
$$= \cos\theta/2\ \mathbb{1} - i\ \hat{n}\cdot\hat{\sigma}\ |\hat{\alpha}|\sin\pi/8$$
$$= R_{\hat{n}}(\theta) \quad \text{where}$$

- $\cos\frac{\theta}{2} = \frac{\sqrt{2}+1}{2\sqrt{2}}$ ← θ is an irrational multiple of π.
- $\hat{n} = \left(\frac{3\sqrt{2}+1}{2\sqrt{2}}\right)^{-1/2}\hat{\alpha}.$

Let $R_{\hat{m}}(\theta) = H^{-1/2} R_{\hat{n}}(\theta) H^{1/2}$.
Observe that $\hat{n}$ and $\hat{m}$ are orthogonal in $\mathbb{R}^3$ since $\hat{p} = \left(\frac{1}{\sqrt{2}}, 0, \frac{1}{\sqrt{2}}\right)$ is orthogonal to $\hat{n}$ and $H^{-1/2}$ is rotation about $\hat{p}$ by angle $-\pi/2$.

Given $U = R_{\hat{n}}(\beta) R_{\hat{m}}(\gamma) R_{\hat{n}}(\delta)$ and $\epsilon > 0$ let $k_1, k_2, k_3 > 0$ be such that

$$\left.\begin{aligned} E(R_{\hat{n}}(\beta), R_{\hat{n}}(\vartheta)^{k_1}) &< \epsilon/3 \\ E(R_{\hat{m}}(\gamma), R_{\hat{m}}(\vartheta)^{k_2}) &< \epsilon/3 \\ E(R_{\hat{n}}(\delta), R_{\hat{n}}(\vartheta)^{k_3}) &< \epsilon/3 \end{aligned}\right\} \text{by Pro. above}$$

Then

$$E(R_{\hat{n}}(\beta) R_{\hat{m}}(\gamma) R_{\hat{n}}(\delta), R_{\hat{n}}(\vartheta)^{k_1} R_{\hat{m}}(\vartheta)^{k_2} R_{\hat{n}}(\vartheta)^{k_3}) < \epsilon$$

by the additivity of the error. $\boxtimes$

This finishes the proof of the universality of the gates $\mathcal{A}_Q = \{H, T, CNOT\}$.

Remark: This proof diverges from NC and can be found in BMP+99.

NC uses

$$R_{\hat{n}}(\vartheta) = THTH \quad \text{and}$$

$$R_{\hat{m}}(\vartheta) = H \, R_{\hat{n}}(\vartheta) \, H.$$

But then $\hat{n}$ & $\hat{m}$ are not orthogonal but just nonparallel. Given two non-parallel unit vectors $\hat{n}$ & $\hat{m}$ and $U \in SU(\mathbb{C}^2)$ we can write

$$U = R_{\hat{n}}(\beta_1) R_{\hat{m}}(\gamma_1) \cdots R_{\hat{n}}(\beta_k) R_{\hat{m}}(\gamma_k)$$

for some k that only depends on the non-parallel vectors $\hat{n}$ and $\hat{m}$ (not on U).

Theorem [Solovay – Kitaev]
Let $\mathcal{A} \subset SU(\mathbb{C}^2)$ be a finite set of elements such that

- if $A \in \mathcal{A}$ then $A^{\dagger} \in \mathcal{A}$.
- $\langle \mathcal{A} \rangle$ is dense in $SU(\mathbb{C}^2)$.

For $\ell \geq 1$ let

$$\mathcal{A}_\ell = \{ A_1 \cdots A_n \;:\; A_i \in \mathcal{A} \;\&\; n \leq \ell \}.$$

Given $U \in SU(\mathbb{C}^2)$ and $\epsilon > 0$ taking $\ell = O(\log^c(1/\epsilon))$ (where $c \approx 4$) there exists $A \in \mathcal{A}_\ell$ such that

$$E(U, A) < \epsilon.$$

Let $U \in SU((\mathbb{C}^2)^{\otimes n})$ be implemented by a circuit consisting of m gates in $SU(\mathbb{C}^2) \cup \{CNOT\}$.
Given $\epsilon > 0$ there exists a circuit over

$$\tilde{\mathcal{A}}_Q = \{ H, T, T^{\dagger}, CNOT \}$$

implementing U' using $O(m \log^c(m/\epsilon))$ gates such that $E(U, U') < \epsilon$.

HW: Prove this. Approximate each gate by error ϵ/m and use additivity of error.

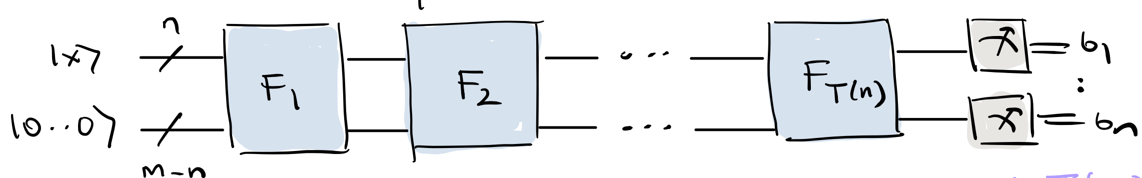
Quantum computational complexity

Let $T : \mathbb{N} \to \mathbb{N}$ be a function.

A predicate $f : \mathbb{B}^* \to \mathbb{B}$ is computable in <u>quantum $T(n)$-time</u> if there exists a deterministic TM that outputs a description of the gates

$$F_1, F_2, \ldots, F_{T(n)} \in \mathcal{A}_Q$$

such that the quantum circuit



outputs $(-1)^{f(x)}$ on the first qubit with probability $\geq 1-\epsilon$ where $\epsilon = 1/3$.

For $b \in \mathbb{B}$ the probability $p(b)$ of observing $(-1)^b$ on the first qubit is

$$p(b) = \sum_{b_2, \ldots, b_m} \underbrace{p(b\, b_1 \ldots b_m)}_{|\langle b\, b_2 \ldots b_m | \psi \rangle|^2}$$

where $|\psi\rangle = |x\rangle \otimes |0 \ldots 0\rangle$.

The language $L = f^{-1}(1)$ is <u>decidable</u> in quantum $T(n)$-time if f is computable in quantum $T(n)$-time.

BQP

Bounded-error quantum polynomial time:

BQP = { L : L is decidable in quantum q(n)-time for some polynomial q }.

Alternative description of BPP:

BPP = { L : ∃ q(n)-time deterministic TM V for some polynomial q such that
$V(\langle x, w\rangle) = P_L(x)$ with probability $\geq 1-\epsilon$ where $\epsilon = 1/3$ }.

witness: $w \in \mathbb{B}^m$, $m \leq q(n)$.

Recall that $P_L : \mathbb{B}^* \to \mathbb{B}$ is the predicate such that $P_L^{-1}(1) = L$.

Circuit complexity:

The size of a circuit is the number of the gates it consists of.

The size complexity of a circuit family $\{C_n\}_{n\in\mathbb{N}}$ is the function $f : \mathbb{N} \to \mathbb{N}$ where

$$f(n) = \text{size of } C_n.$$

The circuit complexity of L is the size complexity of a minimal (in size) circuit family deciding L.

$(b \in L, |b| = n \iff C_n(b) = 1)$

Lem: Let $t : \mathbb{N} \to \mathbb{N}$ be a function.
If $L \in \mathrm{Time}(t(n))$ then L has circuit complexity $O(t(n)^2)$.

Cor: $BPP \subseteq BQP$.

Proof: Replace the verifier V with a circuit of size $O(q(n)^2)$, which can be implemented as a reversible circuit of polynomial size. $\boxtimes$

We will show that

$$\text{FACTORING} \in BQP.$$

It is believed that

$$\text{FACTORING} \notin BPP.$$