Lecture 7 — Biased or Balanced?

(the "Deutsch-Jozsa algorithm")

Recall:

- If $F:\{0,1\}^n \to \{0,1\}$ is easy to compute classically, "$\underbrace{\text{If } F(X_1,\ldots,X_n) \text{ Then Minus}}$" is easy quantumly.

Let $f(x) = \begin{cases} +1 \text{ if } F(x)=0 \\ -1 \text{ if } F(x)=1. \end{cases}$

Then dag for this instr. is

$$000 \xrightarrow{f(000)} 000$$

$$001 \xrightarrow{f(001)} 001$$

$$\vdots$$

$$111 \xrightarrow{f(111)} 111$$

- Add & Diff, Aug & Disp, Hadamard are all the same (if you allow unnormalized states)

$$\sqrt{2} \cdot \text{Aug\&Disp} = \text{Hadamard} = \sqrt{\tfrac{1}{2}} \cdot \text{Add\&Diff}$$

Example 5-qubit calc: Start with .8 ampl. on 11001

.6 ampl. on 11101

Say you do H on 3rd qubit:

[You should pair up all strings so they differ only in 3rd qubit — luckily, this has already happened!]

$$11\underbrace{(.8 \text{ on } 0,\ .6 \text{ on } 1)}01$$

0 w. ampl .8 → 0 w. ampl. $\sqrt{\frac{1}{2}}$

1 w. ampl. $\sqrt{\frac{1}{2}}$

1 w. ampl .6 → 0 w. ampl $\sqrt{\frac{1}{2}}$

1 w. ampl. $-\sqrt{\frac{1}{2}}$

$\therefore$ $.8\sqrt{\frac{1}{2}} + .6\sqrt{\frac{1}{2}} = 1.4\sqrt{\frac{1}{2}}$ ampl. on 11<u>0</u>01

$.8\sqrt{\frac{1}{2}} + .6(-\sqrt{\frac{1}{2}}) = .2\sqrt{\frac{1}{2}}$ ampl. on 11101

Say you *instead* did Add & Diff on 3rd qubit?

→ 1.4 ampl. on 11001

.2 ampl on 11101

} unnormalized

ex 2: Say you start at .8 ampl. on 10000
.6 ampl. on 00101

and do Add & Diff on 3rd qubit?

[Now these strings' "pairs" have 0 amplitude...]

$\left.\begin{matrix} .8 \text{ on } 10\underline{0}00 \\ 0 \text{ on } 10\underline{1}00 \end{matrix}\right\}$ Add & Diff (.8, 0) = (.8, .8)

→ .8 on 10000
.8 on 10100

[Note the ordering!]

$\left\{\begin{matrix} 0 \text{ on } 00\underline{0}01 \\ .6 \text{ on } 00\underline{1}01 \end{matrix}\right\}$ Add & Diff (0, .6) = (.6, −.6)

→ .6 on 00001
−.6 on 00101

Final (unnormalized) answer.

[OK! Review over! Time to do something cool!]

Quantum Trick #1:

Preparing the "uniform superposition".

Recipe: Starting from n qubits initialized to 0, do "H.A.T.!" → H on each qubit

[real name: "Hadamard Transform"]

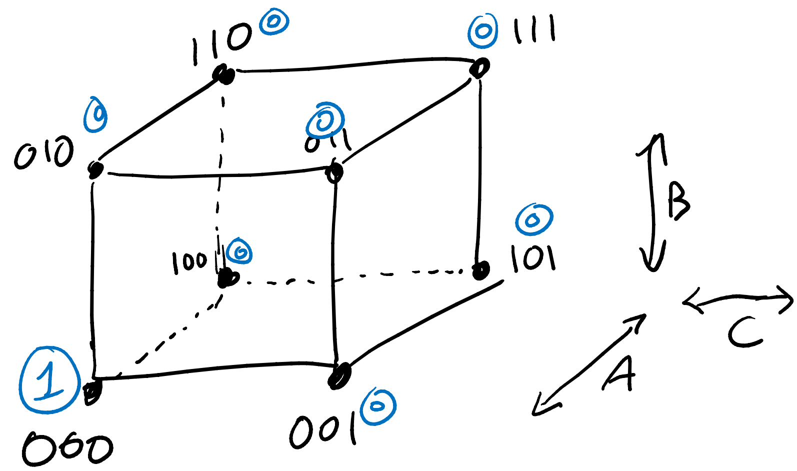
eg: $n=3$, Initially, A, B, C are 000

That is: 1 ampl on 000

0 ampl on 001, 010, ..., 111

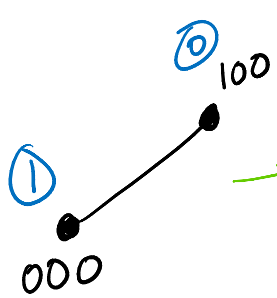
Result of....

$\boxed{\begin{array}{l}\text{H on A}\\ \text{H on B}\\ \text{H on C.}\end{array}}$?

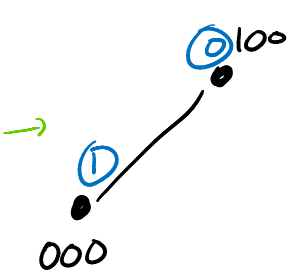


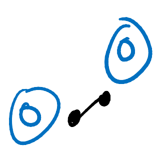
[Conceptually clearer to do Add & Diff, not H.]

Add & Diff on A? [Pair strings up according to only differing on A-bit]

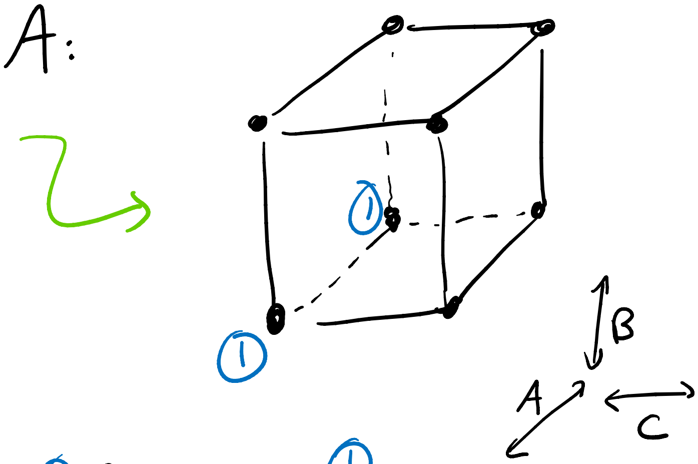


→ Add & Diff(1, 0) = (1, 1) →



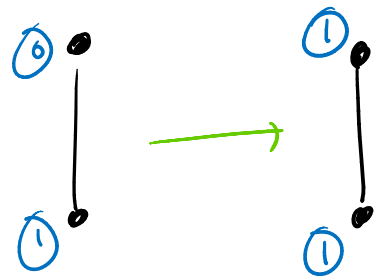
[All other edges:  Add & Diff(0,0) = (0,0) → no change.]

After Add&Diff on A:



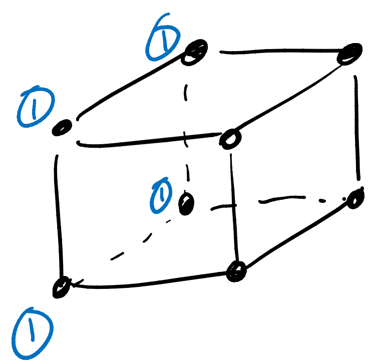
Now Add & Diff on B:

Two B (vertical) pairs like



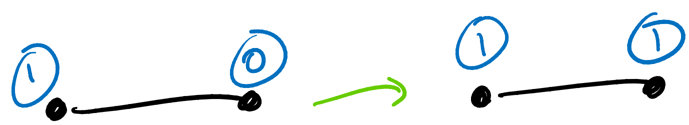
Other 6 pairs are Add&Diff (0,0) = (0,0).

After doing B:



Now Add & Diff on C:

all pairs are



Conclusion: all amplitudes 1 !

Summary:

- Start with n qubits all 0
- Add & Diff on each
- Result: unnormalized state with amplitude 1 on all 2^n strings.
- Add & Diff added n factors of $\sqrt{2}$ over "proper" H instructions
  (normalized state: $\left(\sqrt{\frac{1}{2}}\right)^n$ ampl on all 2^n strings $\rightarrow$ "unif. superposition").

[[But let's leave it unnormalized; I prefer the unnormalized unif. superposition.]]

[[Compare with the "uniform probability distribution" on all n-bit strings: prob. $\left(\frac{1}{2}\right)^n$ on each. So this trick we saw is the quantum analogue of flipping n fair coins.]]

Say $n=3$, and we now add a new var, "Ans".

				x_1	x_2	x_3	Ans
unnormalized	ampl.	1	on	0	0	0	0
"	"	1	"	0	0	1	0
"	"	1	"	0	1	0	0
			⋮				
"	"	1		1	1	1	0

Now do "Add $\underline{\mathrm{Maj}_3}(x_1, x_2, x_3)$ to Ans".

[technically, this involves adding in a few "temp"/"ancilla" vars, but they all are init'd to 0 and end at 0, so we can ignore.]

Result:

				x_1	x_2	x_3	Ans
unnormalized	ampl.	1	on	0	0	0	**0**
"	"	1	"	0	0	1	**0**
"	"	1	"	0	1	0	**0**
"	"	1	"	0	1	1	**1**
			⋮				
"	"	1	"	1	1	1	**1**

[[Wow! By doing just 1 instruction, we computed all $2^n = 8$ answers to Maj_3! That's cool, right?!!]]

Not that cool.

Literally same picture in Probabilistic Computing.

What if we "Print All" now? [[Real term: "measure all"]]

normalized ampls: $\sqrt{\frac{1}{8}}$. $\left(\sqrt{\frac{1}{8}}\right)^2 = \frac{1}{8}$

Result:

			A	B	C	Ans
$\frac{1}{8}$	chance	of	0	0	0	Maj(000)
$\frac{1}{8}$	"	"	0	0	1	Maj(001)
			⋮			
$\frac{1}{8}$	"	"	1	1	1	Maj(111)

[[Literally same as if we had just probabilistically flipped fair coins for A, B, C, computed Maj_3, and printed.
NOT (yet) COOL.]]

[[Remember: to get coolness beyond Prob. Comp., we need minus signs. So how about...]]

Prep unif. superpos. on A, B, C.

Then do "If $Maj_3(A,B,C)$ Then Minus".

Result:

			A	B	C
unnormalized ampl.	+1	on	0	0	0
"	+1	"	0	0	1
"	+1	"	0	1	0
"	−1	"	0	1	1
"	+1	"	1	0	0
"	−1	"	1	0	1
"	−1	"	1	1	0
"	−1	"	1	1	1

[[Neat, like the "Minus World" truth table of Maj_3]]

Q: Cool yet?

A: Not really. If we "Print All" now....

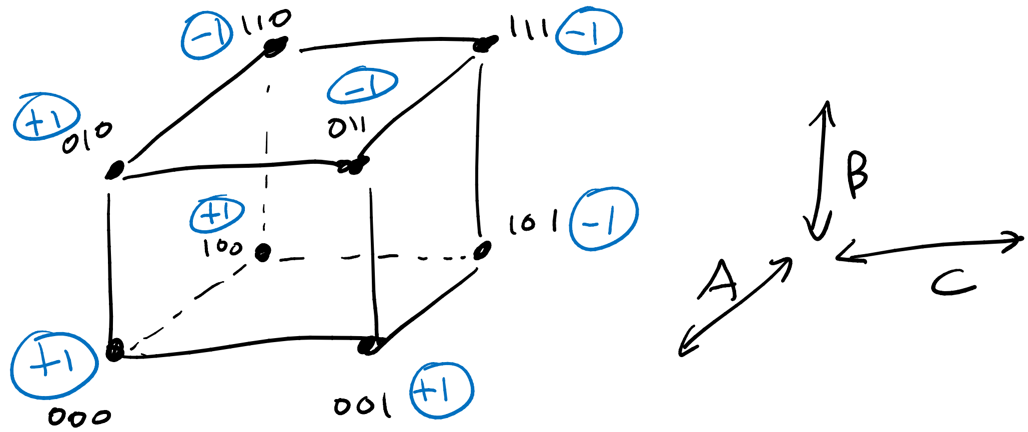
$\rightsquigarrow \left(\pm\sqrt{\tfrac{1}{8}}\right)^2 = \frac{1}{8}$, so we just see a random string!

⟦Yes, we got minus amplitudes, but we still haven't done the really cool trick of making them "interfere"/cancel. How to do that? Add them back together! We're going to Had each qubit again!⟧

Biggest (arguably, <u>only</u>) trick in all Quantum Computing:

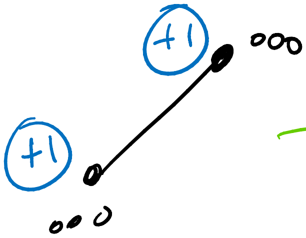
$$\boxed{\text{DO THE HADAMARD TRANSFORM (H.A.T.! H on all qubits) ON THE SIGN-COMPUTED VERSION OF F.}}$$

⟦It's nicer to do Aug & Disp rather than H on each qubit. The extra n factors of $\sqrt{\frac{1}{2}}$ also cancel the extra n $\sqrt{2}$'s from prepping the uniform superposition. Nice!⟧

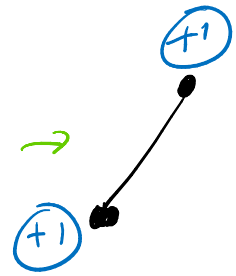


± 1: unnormalized ampl. after sign-computing Maj_3 on unif. superpos.

Avg & Disp. on A:

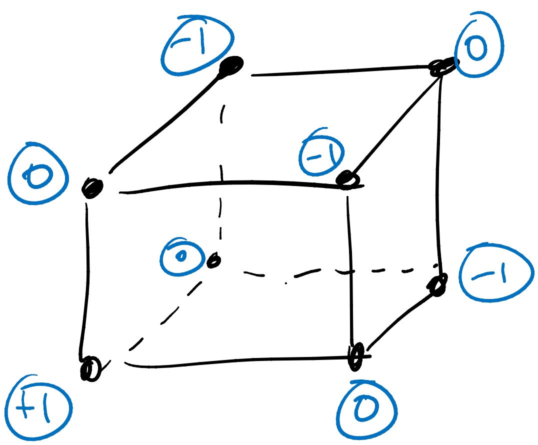


$\to$ Avg&Disp$(+1,+1) = (+1, 0)$ $\to$



Since also Avg&Disp$(+1,-1) = (0, +1)$

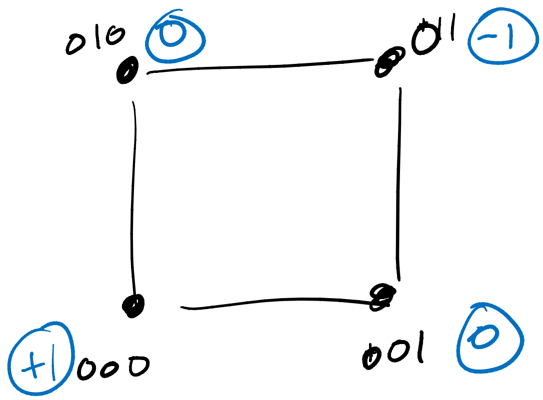
$(-1,-1) = (-1, 0)$, result is

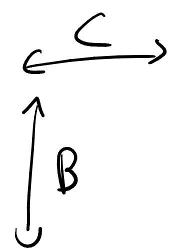


[Going to be a bit of a pain to continue. Let's just try to focus on a simple question...]

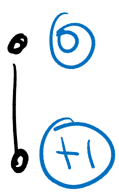
Q: Well, what's just final ampl. on
00……0?

[[If we only worry about that, we can ignore the "back face" in preceding diagram.]]

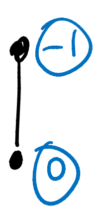




Now Avg & Disp on B. [[If only caring about final ampl. on 00…0, can ignore "right side".]]

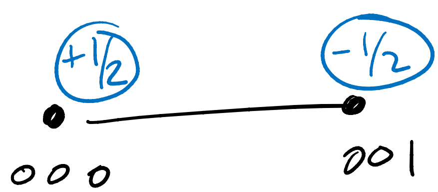


$\longrightarrow$ AvgWhoCares$(+1, 0) = (+\frac{1}{2}, ?)$



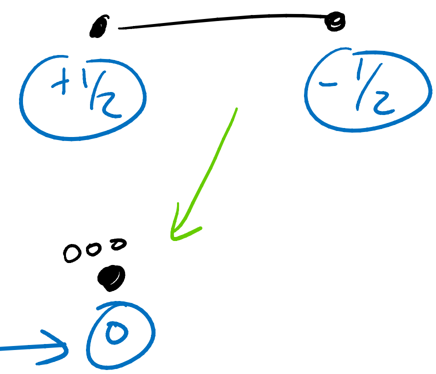
$\longrightarrow$ AvgWhoCares$(0, -1) = (-\frac{1}{2}, ?)$

Result:



[[I hope you can tell that if you're only working out ampl. on 00...0, you just keep averaging, and you end up with ...]]

Avg & Diff on C:



OVERALL AVG OF ALL $2^n = 8$ entries in F's "minus world" truth table!

[[Cancellation occurred! This *is* cool!]]

Theorem: After prepping unif. superpos, sign-computing $F: \{0,1\}^n \to \{0,1\}$, and doing Had. Transform, normalized ampl. on 00....0 is: avg. of all F's ± 1 values.

[[Remark: the other ampls. have some meaning too, but we'll get to that later on...]]

ex: In Maj_3 example, final state is

$$(0 \cdot |000\rangle) + \tfrac{1}{2}|001\rangle + \tfrac{1}{2}|010\rangle + \tfrac{1}{2}|100\rangle - \tfrac{1}{2}|111\rangle.$$

Question: When is ampl. on $00\cdots0$ exactly 0?

Answer: When F is exactly "<u>balanced</u>":

truth table is exactly half 0, half 1.

[in minus world: +1 , −1]

Now let's "Print All".

F balanced $\Rightarrow$ $\mathrm{Ampl}[00\cdots0] = 0$

$\Rightarrow$ $\mathrm{Prob}[\text{printing } 00\cdots0] = 0^2 = 0.$

F "<u>biased</u>" (<u>not</u> balanced) $\Rightarrow$ $\mathrm{Ampl}[00\cdots0] \neq 0$

$\Rightarrow$ $\mathrm{Prob}[\text{printing } 00\cdots0] = (\text{nonzero})^2 > 0$

strictly positive

Conclusion: Say $F: \{0,1\}^n \to \{0,1\}$ easily computed on a classical computer.

Then there is an easy quantum program C with this behavior:

- If F's truth table balanced,
  C never prints 000...0
- If F's truth table biased,
  C prints 000...0 with prob. > 0.

Is this cool? Yes!

[This behavior is not practically useful at all.
But... thanks to obscure computational complexity work from ≈ 1990 (before any quantum algs existed...) we know...]

Thm: If we had the same behavior but with a classical alg. C, then...

$$P^{\Sigma_2} = NP^{\Sigma_2}$$

[Some equally unbelievable "poly-time hierarchy" generalization of P=NP.]

[So quantum comps can do this weird "bias busting" task, classical comps. can't (unless $P^{\Sigma_2} = NP^{\Sigma_2}$!!!)