2- the complexity of the move search

we Know that all the possible candidates are $N_0 = 2^n$

now after making a guess we are getting k which is the grade of the guess

now after the guess each time we are fliping the bits of the candidates and making the list shorter the list is getting to

$$N_1 = \binom{n}{k} = \frac{n!}{k!(n-k)!} \simeq \frac{\sqrt{2\pi n}\left(\frac{n}{e}\right)^n}{\sqrt{2\pi k}\left(\frac{k}{e}\right)^k \sqrt{2\pi(n-k)}\left(\frac{n-k}{e}\right)^{n-k}}$$

and the max number of new candidates will be $k = \frac{n}{2} \rightarrow N_1$ is max

$$N_{e_{max}} = \frac{\sqrt{2\pi n}\left(\frac{n}{e}\right)^n}{2\pi \frac{n}{2}\left(\frac{n}{2e}\right)^n} = \sqrt{\frac{2}{n}} \frac{2^n}{\sqrt{n}}$$

now if we calculate the ratio of the change between N_0 and N_e we will get the complexity $\rightarrow \frac{N_0}{N_1} = \sqrt{\frac{\pi}{2}}\sqrt{n} \simeq \sqrt{n}$

3- as described in the solving_mastermind.pdf

In github repo you can see a detaily solving steps and there I defined the states as for example n=3

→ the states will be

$\underbrace{|000\rangle}_{n}\underbrace{|00\rangle}_{\text{score in compaing to the gues}}\underbrace{|000\rangle}_{\text{not-xor}}$

4- as described in the solving_mastemind.pdf look for usage examples over there

I am using 3 different quantum gates the first is the simple Hadamard gates that gives us all the possible candidates the superposition then I am using $\boxed{E}$ which takes in two states and does not-xor and then then it sums the obtained 1 bits

$|000\rangle|100\rangle|000\rangle \rightarrow \boxed{\text{not-xor}} \rightarrow |0010\rangle \xrightarrow[\text{of bits}]{\text{sum}} \boxed{1}$

$|010\rangle|000\rangle|000\rangle$

//

É gate

then there is gat $\boxed{P}$

gate $\boxed{P}$ is the gate that $|000\rangle|100\rangle|000\rangle$ changes the score of this candidate by fliping the first 0 bits in the first qubit correspanding to the zeros in the not-xor (3rd qubit) and this was in case if the difference in score of this candidate and the score of the guess is positive if the difference of the score was negative then we flip the first 1 bits

ok now lets design this gate

this gate takes 2 inputs:

first input is the difference

(guess score - candidat score)

secand input is the candidat example |000>|01>|010>

|ghi>

now if the difference first input is positive

then need to do the fallowing

we need set up n ancillary qubits and make

them |0> then we need to creat another

qubit and Initialize it to 0 ~~based on the maximum~~

(based on the difference number) like |00> = C

now we have ancillary gates anc0, anc1, anc2

we do CNOT(0, anc0), CNOT(1, anc1), CNOT(0, anc2)

|ghi> |ghi> |ghi>

→ anc = 1 anc1 = 0 anc2 = 1

now for each anci if anci == 1 and

C < difference

→ flip bit i in first qubit of input2

---

now for case of negative difference.

we do the same but for every anci == 0

then flip bit i in the first qubit

5- you cand demonstration of the gates in

solving_mastermind.pdf