Quantum Mechanics Enabled Genetic Programming

***Abstract*—Quantum Computing (QC) has often been touted as an esoteric and terrifying field of computing research. However, the possible advantages offered by the inherent quantum fundamentals beseeches extensive additional ventures into this field. Just as QP offers exciting ideas in the field of computing, Genetic Programming (GP) offers an application oriented optimization route. GP uses Darwinian theories to maintain a set of candidate solutions, apply multiple operations on the candidates and eventually declare a global winner. In this paper, we combine QC and a flavour of GP to create a new interdisciplinary front of computational intelligence.**

***Index Terms*—component, formatting, style, styling, insert**

I. Introduction

Majority of the problems faced in any aspect of computer science, in one way or another, involves a catch-22 of multi-functional optimization. Entire fields have been dedicated to solving a generic version of this issue. The most common method of dealing with optimization problems is to be in collaboration with another interdisciplinary frontier acting in the capacity of a helper function.

A. Computing Methodologies

Most form of computational algorithms in the present day, are executed on a conventional computer. The fundamental notation used for differentiating Classical Computing (CC) and QC is the basic unit of information. While classical computers use bits 0 and 1, quantum computers use "one of" two computational basis states. The label awarded to these states is "bra-ket" notation i.e, state $|0\rangle$ or $|1\rangle$. Bits are assigned states 0 or 1 deterministically and independently. However, qbuits can exist in a superposition state of $\alpha_0|0\rangle$ + $\alpha_1|1\rangle$, where $\alpha 0$ and $\alpha 1$ are complex numbers, such that $|\alpha_0|^2$ + $|\alpha_1|^2$ = 1.

B. Why go Quantum?

The field of applied quantum mechanics is still unexplored for the best part. However, there are certain applications in which QC outperforms CC. Consider Shor's algorithm and RSA, both of which are used for encryption. Basically, the encryption of any form of data transmitted over the Internet relies immensley on factorization of a huge number. This process is extremley ardous for a non-quantum computer, with the best known factoring technique requiring an amount of time proportional to $2^{n^{\frac{1}{3}}\log(n)^{\frac{2}{3}}}$, where *n* is the number of digits in the number to be factored. Meanwhile Shor's Quantum algorithm requires time proportional to only $n^2 log(n) log(log(n))$.

QC takes advantage of major quantum phenomena such as superposition, quantum entanglment, principle of uncertainity among others, for improving existing searchc and optimization techniques. Humans inherently stick to definitive physical concepts such as deterministic state transition, state duality and temporal static behaviour of particles.

According to Moore's Law, the size of computational units shrinks at an exponential rate as the number of transistors on a chipset increases every year. Even after accounting for physical constraints, a time will come when operations will be conducted on an atomic scale. On such a level, atomic forces overpower particle physics naturally. As a result, understanding QC and applying them in a virtual landscape to get a sense of their possibilities is a must.

C. Using Evolutionary Algorithms

Evolutionary Algorithms (EA) encompass a wide array of research ideas stemming from general principles of genetics and the theory of evolution. Models are developed to illustrate the behaviour of naturally occuring phenomena, develop these algorithms and test out the application of the corresponding theory. The general process of any EA is depicted in figure 1. In a nutshell, an initial population of candidate solutions is generated and each solution is labelled according to their performance on a set fitness function. Further down the line, these candidates are improved upon by conducting a set of operations on them for a predetermined number of generations. After some stopping criterias are achieved, the process strops and the best candidate is hypothesized as the global solution. The advantages offered by this technqiue include but not limited to resilience to noise, in-built support for parallelism and distributed learning, multi-pronged attack and handling complex problems.

Our paper provides an insight into the ensemble of both the previously mentioned techniques and preparing algorithms for a future where quantum computers replace traditional computers as commonplace machines.

II. Methodology

One of the most famous quantum effect invovles "superposition", which allows a quantum entity to exist simultaneously. As a result memory conservation capabilities of any quantum system increases exponentially. Coupling this idea with multiple evolutionary aspects, we present a novel computational frontier.

A. Existing Convention

Consider the following problem; Assume the position of a process *P* in a finite state system of *n* states at time *t* to be x_i. Create a general procedure to reach position x_j at time *t+1*.

The conventional process would involve declaration of a register of size *n* and listing out all possible combinations

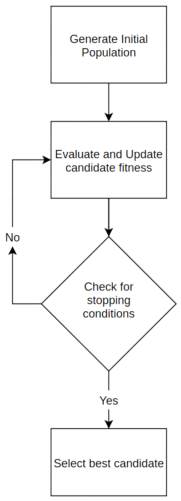

Fig. 1. General Flow of Evolutionary Algorithms

for reaching state x_j along with the costs. This would be followed by creation of a logical circutry of gates mapping the input state to the output state using boolean primitives. The entire process could still be automated using certain advanced algorithms. However, the memory utilization will not be optimal as all possible state transitions must be considered before building the circuit. Figure 2 displays the state table, state diagram and accompanying circuit diagram for a D flip flop. Notice that all possible states for the input have been mentioned in the state table. This constitues a huge waste of memory as majority of the inputs are never encountered and some states could be invalid. QC aims to reduce this inefficiency between making all the states dependent on each other.

B. Applied Quantum Mechanics

Classicial state transitions cannot fathom superposition and hence will always be fixated on one fixed state. For the D flip flop example, the number of inputs are two and hence the number of output states is 2^2 for each starting state. The

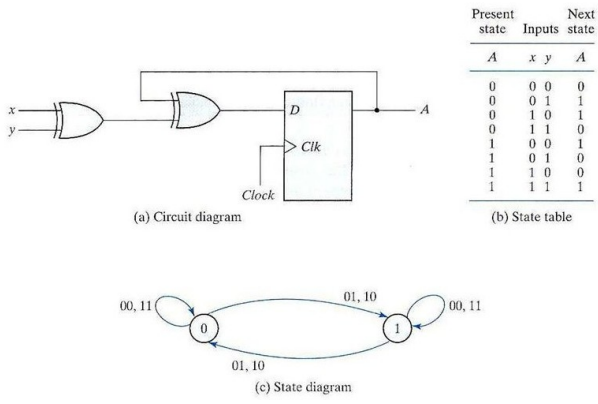

Fig. 2. States for D flip flop

4 possible states can be represented as an one-hot encoded vector with each bit representing the state of the output. For example, the vector shown below represents state 00 for an input state of 0.

$$\begin{bmatrix}1 & 0 & 0 & 0\end{bmatrix}$$

Quantum Mechanics is a generalized extension of classical method. The first generalization comes in the form of the temporal values of the elements of the matrix. In QC qubits are utilized instead of bits and as a result the state vectors are variables. Instead of 0 and 1, state vector can be constituted of complex numbers which meet the condition of unitarity. For any matrix M to be an unitarity matrix, the only necessary condition is shown in Equation 1 where $M^\dagger$ is the Hermitean adjoint of M and I is the identity matrix.

$$M^\dagger M = M M^\dagger = I \tag{1}$$

To facilitate the design of circuitry of state diagrams, QC introduces a set of gates to utilize the power of quantum mechanics. The noteworthy ones are mentioned below:

- QNOT: This gate is the quantum extension of the controlled NOT gate in CC. A controlled NOT gate performs the inversion operation on the right-most bit of a vector iff the left-most bit is 1. A 1-bit QNOT gate is depicted as follows:
$$\begin{bmatrix}0 & 1\\ 1 & 0\end{bmatrix}$$
- SRN: The application of Square Root of NOT puts a qubit into a state of equal superposition through controlled randomization based on the qubit's past state. A 1-bit SRN is shown below:
$$\begin{bmatrix}\frac{1}{\sqrt{2}} & \frac{-1}{\sqrt{2}}\\ \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}}\end{bmatrix}$$
- Swap: As the name suggests, this gate is used to swap the states of two qubits and is represented as:
$$\begin{bmatrix}1 & 0 & 0 & 0\\ 0 & 0 & 1 & 0\\ 0 & 1 & 0 & 0\\ 0 & 0 & 0 & 1\end{bmatrix}$$
- Parameterized Rotation: This is a family of 1-qubit rotations parameterized by angle θ and is given by:
$$\begin{bmatrix}cos(\theta) & sin(\theta)\\ -sin(\theta) & cos(\theta)\end{bmatrix}$$

C. Applied Evolutionary Operators

EA have been used extensively for complex optimization problems due to quality results even with a lack of mathematical representation, convexity and continuity. They apply a stochastic method and hence expedite the search process. However, EA are not without their flaws which include a probabilistic convergence to the global optima.

In this paper, we follow a genetic modified Grey Wolf Optimization (GWO) algorithm to obtain and update candidate

Algorithm 1 GWO pseudocode

Input: Population size n, random vectors $\vec{r_1}$, $\vec{r_2} \in [0,1]$, initial prey location $\vec{D}$, number of iterations I, fitness function f, coefficient vectors $\vec{A}$, $\vec{C}$ and $\vec{a}$
Set $t = 0$
for i $\in$ [1,n] **do**
    Generate wolf pack population $X_i(t)$ at instance t
    Evaluate each individual against the fitness function
end for
Assign α, β, δ titles to the top three solutions
Evaluate $\vec{D} = |\vec{C} \cdot \vec{X_p}(t) - \vec{X}(t)|$
for i in I **do**
    for Each individual in n **do**
        $\vec{X_1} = \vec{X_\alpha} - \vec{A_1} \cdot \vec{D_\alpha}$
        $\vec{X_2} = \vec{X_\beta} - \vec{A_2} \cdot \vec{D_\beta}$
        $\vec{X_3} = \vec{X_\delta} - \vec{A_3} \cdot \vec{D_\delta}$
        Evaluate $\vec{X}(t+1) = \frac{\vec{X_1}+\vec{X_2}+\vec{X_3}}{3}$
    end for
    Update the coefficient vectors $\vec{A}$ and $\vec{C}$
    $\vec{A} = 2\vec{a} \cdot \vec{r_1} - \vec{a}$
    $\vec{C} = 2\vec{r_2}$
    Linearly decrease $\vec{a}$ from 2 to 0
    Perform Crossover, Selection and Mutation
    Increment t
end for
$\vec{X_\alpha}$ corresponds to the global optimum.

solutions. GWO follows a hierarchical structure of candidate solution which imitates the behaviour of Grey Wolves as observed in nature. The structure of α, β, δ and ω labelled wolves. The α wolves are responsible for finding the global optimum whilst being guided by the β wolves. The δ and ω provide secondary assistance to the leaders and hence maitain the dominant structure. After each search iteration, genetic operators like selection, crossover and mutation are applied to every candidate in the structure. The entire process is discribed in Algorithm 1

The genetic operators for a sample candidate in the form of a genomic are explained here following which the process is depicted in Figure :

- Selection: Given a population of potential solutions, selection involves grading these solutions against a fitness function. The selected solutions are cast into the crossover stage.
- Crossover: From the selected set of solutions, randomly two solutions are selected and the next generation chromosome is created which shares the best attributes of it's parents.
- Mutation: The selected solutions undergo random changes in their composition to promote diversity and increase their changes of converging to the global optimum.

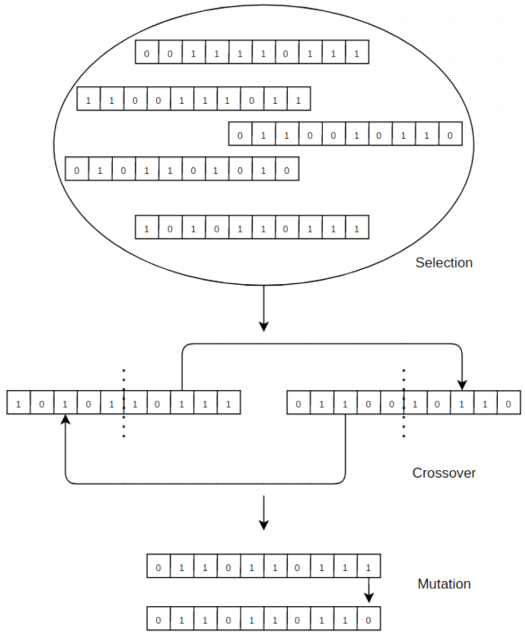

Fig. 3. Genetic Operators