Quantum Algorithm as a PDE Solver for Computational Fluid Dynamics (CFD)

- **Name Of the Participant**: Sajib Halder
- **Wiser ID:** gst-bDwu8WDOuIWEGh3
- **Team Name:** Quevian's Paper Boat (Individual Participation)

Overview of the Project: Imagine we are trying to simulate how air flows over an airplane wing or how water moves through a pipe. Scientists use a technique called Computational Fluid Dynamics (CFD) to achieve this, which involves solving complex mathematical equations known as **Partial Differential Equations (PDEs)**. One such equation is called the **Burgers' Equation**. It is a **simplified version** of more complex fluid equations, but still useful for testing.

In this challenge, we build a quantum-enhanced PDE solver based on two recently proposed frameworks:

- **QTN (Quantum Tensor-Network):** It is a method where a complex and high-dimensional system is broken down into simpler components using **matrix product states (MPS).** It compresses the fluid velocity (data about how fluid moves) into something (i.e., shrinking big fluid data into smaller quantum bits) that a quantum computer can handle.
- **HSE (Hydrodynamic Schrödinger Equation):** In HSE, the dynamics of fluid flow, **described using a quantum wave-function**, similar to how particles are described in quantum mechanics. It reformulates incompressible fluid equations into a Schrödinger-like form, enabling simulation on quantum processors for enhanced performance.

We are free to utilize **either the QTN or HSE algorithm (hybrid QTN-HSE approaches also)** to solve the Partial Differential Equation in CFD of the Burgers' Equation for 1D Shock Tube:

PDE: $\frac{\partial u}{\partial t} + \frac{u\partial u}{\partial x} = \frac{v\partial^2 u}{\partial x^2}$ Domain: $x \in [0, 1]$ IC: Riemann step $u[x, 0] = 1$ for $x \leq 0.5, 0$ otherwise BC (Dirichlet): $u(0,t) = u_L, u(L,t) = u_R$ for all $t > 0$	• $u(x,t)$: Velocity as a function of position x and time t • $\frac{\partial u}{\partial t}$: Change in velocity over time • $u\frac{\partial u}{\partial x}$: Convective term (nonlinear transport) • $\nu\frac{\partial^2 u}{\partial x^2}$: Viscous diffusion term (spread due to viscosity ν)

Understanding the Burger Equation and Some Key Terminologies: The **Burgers' equation** originates from the **Navier-Stokes(NS)** equations, which govern fluid motion. By simplifying certain aspects **(1D flow, low pressure, no external forces),** we arrive at the Burgers' equation. Let's understand some key terminology:

- **1D Shock Tube:** It is a simplified experimental model used in fluid dynamics to study shock waves. Here, the problem considers only lengthwise movement (1-dimensional)of the fluid and waves, ignoring height or width, it is called a **"1D Shock Tube"** problem.
- **Convective Term (nonlinear transport): "Convection"** is the process where a quantity (like velocity, heat, or gas) moves and carries itself. Example: A river whose current becomes faster or slower based on how strong the flow is at a point - this is convection.
- **Viscous Diffusion Term: "Viscous"** refers to fluid friction (internal resistance), and **"Diffusion"** means spreading out. This term models how the velocity smooths out over space due to viscosity v, which causes the fluid to resist rapid changes. Example: when we stir sugar in water, it gradually spreads out evenly - that spreading is diffusion, and the fluid's viscosity affects how fast it happens.
- **IC - Initial Condition:** The state of the system at the beginning of time (t=0) is called the initial condition. It tells us the starting velocity across the domain.
- **BC - Boundary Condition:** It defines the behavior at the edges of the spatial domain (e.g., the ends of the tube). Let's if the velocity at the left end is fixed at 1, and the right end is 0 for all time, that's the boundary condition.
- **Trotterisation:** It is a method used to **break down** complex **Hamiltonians (operators that govern quantum systems)** into **simpler parts** that can be **executed as sequences of quantum gates.** It is based on **Trotter-Suzuki decomposition,** which approximates the exponential of a sum of non-commuting operators as a product of exponentials of individual terms. It's **crucial for simulating time evolution** in quantum systems.

$$e^{-i(H_A+H_B)t} \approx \left(e^{-iH_At/n}e^{-iH_Bt/n}\right)^n$$

Mapping the PDE to a Quantum World before Algorithm Design: We take the Burgers' equation and **translate it** into a form that qubits can understand. This is called **discretization** - we chop the tube into small sections (called **grid points**). Each point becomes a qubit or part of a qubit's job. For example, if we have 4 grid points (such as 00, 01, 10, 11), we need 2 qubits,

and for 16 grid points, we need 4 qubits (because $2^4 = 16$). We also choose time steps (like frames in a movie), i.e., telling the quantum machine to show how the fluid changes in this tube - frame by frame!"

Building the Quantum Circuit (Gate Decomposition): Now, we construct the machine - the quantum circuit. We use quantum gates to control qubits; think of gates like they rotate, flipping, and entangling qubits. Some common gates are - **Hadamard (H)** – Splits the coin between heads and tails (a superposition), **CNOT** - Links two coins so they act like twins (called entanglement), **RX / RZ / U3** – Rotate the coin in space (like a spinning top). We carefully arrange these gates to simulate the Burgers' flow. Each time step is meant to be one layer of gates. Multiple time steps are intended to be many layers (i.e., a deep circuit).

Algorithmic Design with Various Quantum Frameworks as follows:

1. Considering the QTN framework and its Workflow:

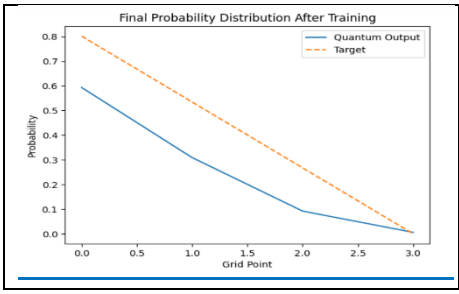
- First, we **discretize the domain** using 4 grid points (2 qubits), then 16 grid points (4 qubits). Each point on the grid corresponds to a **qubit amplitude, forming a quantum state.** Now, **QTN** implies that, applying a tensorized set of unitary gates to evolve the quantum state represented by u(x).
- **Encoding**, i.e., mapping classical vectors u0, u1, etc., into quantum amplitudes.
- **Time Evolution**, i.e., applying a quantum circuit (approximating Burgers' operator via Trotterization).
- **Measurement**, i.e., reading back values ui(t) from qubit measurement.

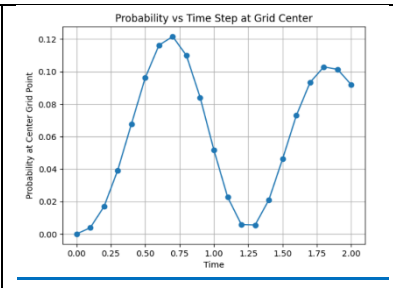
2. Considering the HSE framework and its Workflow:

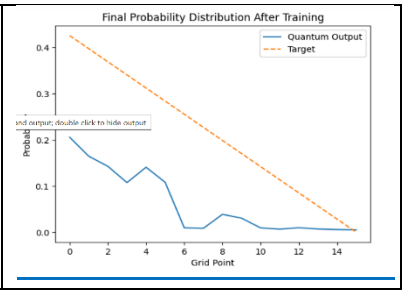
- The HSE method, in general, converts the nonlinear Burgers' Equation into a quantum-like Schrödinger Equation using a complex wavefunction ψ. But our focus is mainly on building a **practical quantum circuit implementation using a simplified Hamiltonian.**
- First, we construct a **toy Hamiltonian using Pauli gates:** $H = v \cdot (Z \otimes Z) + X$. That is, we model viscous effects with simple Pauli terms. Toy Hamiltonian combining Pauli $Z \otimes Z$ interaction (entanglement) and Pauli X (rotation) to mimic dissipative fluid behavior.
- Then we **initialize the superposition state by Hadamard superposition** and apply Trotterized time evolution using a simplified Hamiltonian. And then measured the full probability distribution of qubit states.

Performing a quantitative comparison of the quantum solver against a classical solver:

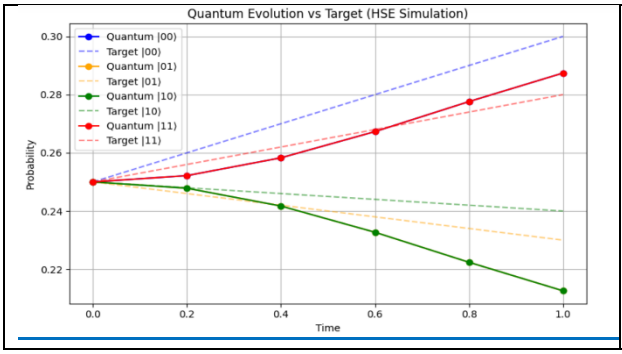
Case 1: Validation & Benchmark for QTN

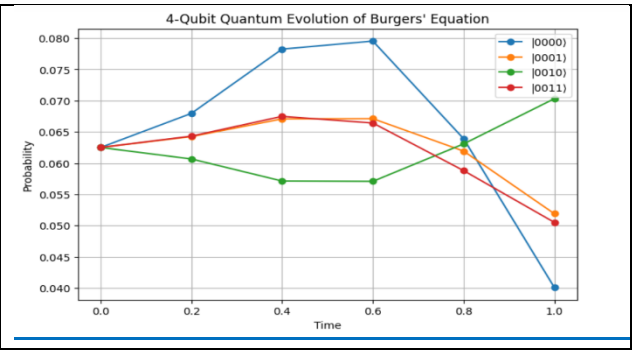






Case 2: Validation & Benchmark for HSE





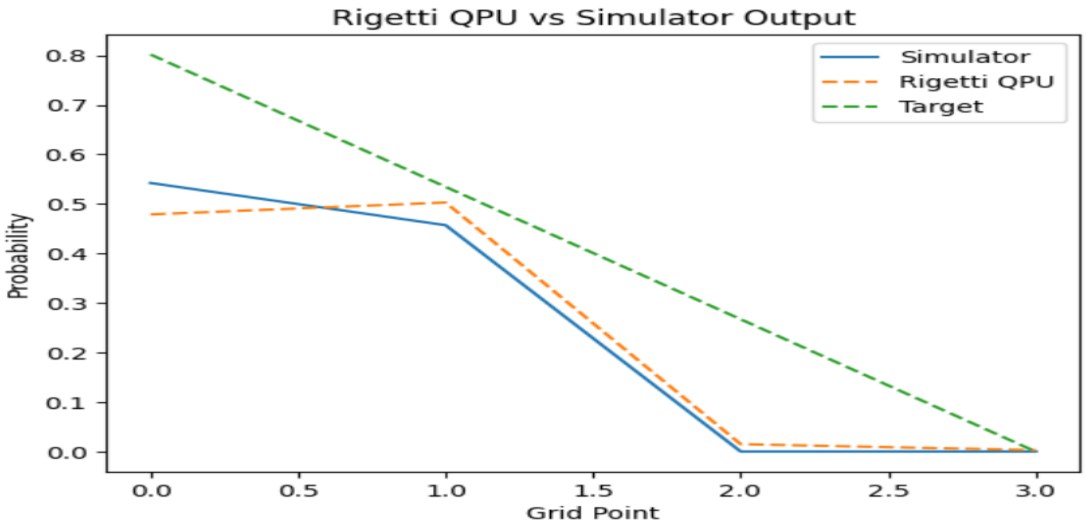
Resource Analysis – No. of qubits and Gate depth:

Gate Depth: Maximum number of layers that must be executed sequentially, assuming parallel gates on disjoint qubits.

Framework	***No. of Qubits***	***Two-Qubit-Gate Depth***
1. QTN	2	6
2. HSE	2	4

Noise Analysis: In our HSE approach, we depolarize noise after Hadamard gates (initialization) and after the Hamiltonian evolution to simulate realistic noise effects and observe how noise affects the output of the circuit. These noisy results can serve as a baseline for future comparison with mitigated outputs.

Benchmark on a physical QPU: Until now, all implementations of the QTN and HSE quantum frameworks have been executed and validated on our local simulation environment. To evaluate more appropriate behavior in a realistic quantum system, we conducted the same experiments **using the Rigetti backend via Amazon Braket.** The real quantum outputs were successfully generated and stored in my Amazon S3 bucket. Image below as follows:



Scalability Study - how resources scale with grid size:

As the **grid size increases** (let's, from 2 to 4 qubits), the quantum resource requirements - such as the number of gates, gate depth, and **measurement overhead - grow significantly.** This scaling directly affects both simulation time and noise on real hardware. In particular, the QTN approach requires more operations, as we have seen from the previous table.

Algorithm Comparison - trade-offs between QTN and HSE:

The **QTN (Quantum Tensor Network)** approach **offers a higher accuracy** by preserving entanglement structures **but demands deeper circuits and more qubits.** On the other hand, the **HSE (Hydrodynamic Schrödinger Equation)** approach **uses shallower circuits,** though it may compromise precision in capturing complex dynamics. The choice between these approaches involves balancing computational efficiency and the demand for early-stage experimentation.