This notebook demonstrates how derangement (permutation) circuits of the Error Suppression by Derangements (ESD) [arXiv:2011.05942] and Virtual Distillation (VD) [arXiv:2011.07064] techniques can be constructed and used to suppress errors in noisy quantum devices. The approach takes n copies of a noisy quantum circuit and reduces errors exponentially as Q^n, where Q<1.

Definitions

Make sure to run the code below -- but don't worry about the details, these are just definitions of the functions our demo will use

In[∘]:=

```
(* Import QuESTlink *)
Import["https://qtechtheory.org/QuESTlink.m"];
CreateDownloadedQuESTEnv[];

(* controlled SWAP operation between two full registers *)
cSWAPregs[reg1_, reg2_, qubitsperReg_, controlQB_] := Table[
  C_controlQB[SWAP_reg1*qubitsperReg+k,reg2*qubitsperReg+k], {k, 0, qubitsperReg - 1}]

(* Derangement circuit for 2 copies *)
D2[qubitsperReg_, observable_] := With[{controlQB = 2 * qubitsperReg},
  Join[{H_controlQB}, cSWAPregs[reg1 = 0, reg2 = 1, qubitsperReg, controlQB],
   Circuit[Evaluate@observable] /.
    {Id_n_ :> Id_n+qubitsperReg, O_n_ :> C_controlQB[O_n+qubitsperReg]}
   , {H_controlQB}]
 ]


txt[in_] := Text[Style[in, FontFamily → "Times New Roman", FontSize → 12]]

(* Draw derangement circuit with highligting the different registers *)
DrawDerangementCircuit[derangementCirc_, numQs_, numCopies_] :=
 DrawCircuit[derangementCirc,
  Epilog → Join[{
         FaceForm[None], EdgeForm[Directive[Red, Dashed]]},
    Flatten@Table[
          {Text[
        Style[StringForm["Reg. ``", k], FontSize → 12], {0, k * numQs + .4}, {-1, -1}],
       EdgeForm[Directive[{Red, Blue}[[Mod[k, 2] + 1]], Dashed, Opacity[0.4]]],
           Rectangle[{0, k * numQs + .2}, {Length[derangementCirc],
          (k + 1) numQs - .2}]]}, {k, 0, numCopies - 1}]
        ]
 ]

(* Create n copies of a circuit -- this
 can be the input of the derangement circuit *)
CopiesOfCircuit[circuit_, numCopies_, qubitsperReg_] :=
 Flatten@Table[circuit /. {gate_n_ :> gate_n+k*qubitsperReg,
     gate_n1_,n2_ :> gate_n1+k*qubitsperReg,n2+k*qubitsperReg}, {k, 0, numCopies - 1}]
```

1) Derangement (SWAP) circuit for two input registers

We set how many qubits there are in a register

In[ ]:=
```
numQs = 5;
```

We set the observable we want to measure as products of Pauli operators (note that the ESD/VD technique suppresses errors in measuring observables)

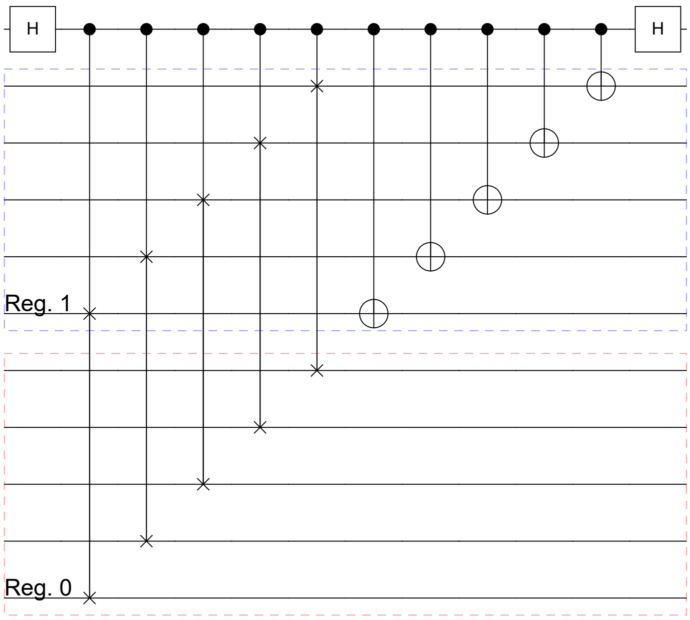
In[ ]:=
```
observable = Product[X_k, {k, 0, numQs - 1}]
```

Out[ ]= $X_0 X_1 X_2 X_3 X_4$

The derangement circuit permutes the two input registers Reg. 0 and Reg. 1 and then measures the expectation value of the observable

In[ ]:=
```
derangementCirc = D2[numQs, observable];
(* Draw derangement circuit and highligt the different registers *)
DrawDerangementCircuit[derangementCirc, numQs, 2]
```

Out[ ]=



Total number of qubits we need to simulate (2 registers and 1 ancilla qubit)

In[ ]:=
```
totNumQ = 2 * numQs + 1
```

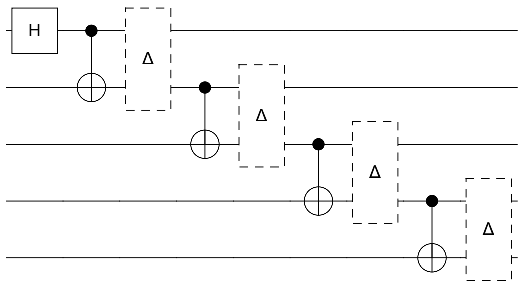
Out[ ]= 11

2) Mitigate errors in noisy circuits

Noisy circuit that generates GHZ states

Construct a noisy quantum circuit that generates an *n*-qubit GHZ state $(|0,0,0,0,0\rangle+|1,1,1,1,1\rangle)/\sqrt{2}$ with noise rate ϵ

In[ ]:=
```
ϵ = 0.01;
noisyGHZcirc =
  Join[{H_{numQs-1}}, Flatten@Table[{C_n[X_{n-1}], Depol_{n,n-1}[ϵ]}, {n, numQs - 1, 1, -1}]];
DrawCircuit[noisyGHZcirc]
```

Out[ ]=


Simulate the noisy circuit using QuESTlink

In[ ]:=
```
{ρ, ϕ} = CreateDensityQuregs[numQs, 2];
ApplyCircuit[noisyGHZcirc, InitZeroState@ρ];
```

Compute the error of the expectation value $\langle X_0\ X_1\ X_2\ X_3\ X_4 \rangle$

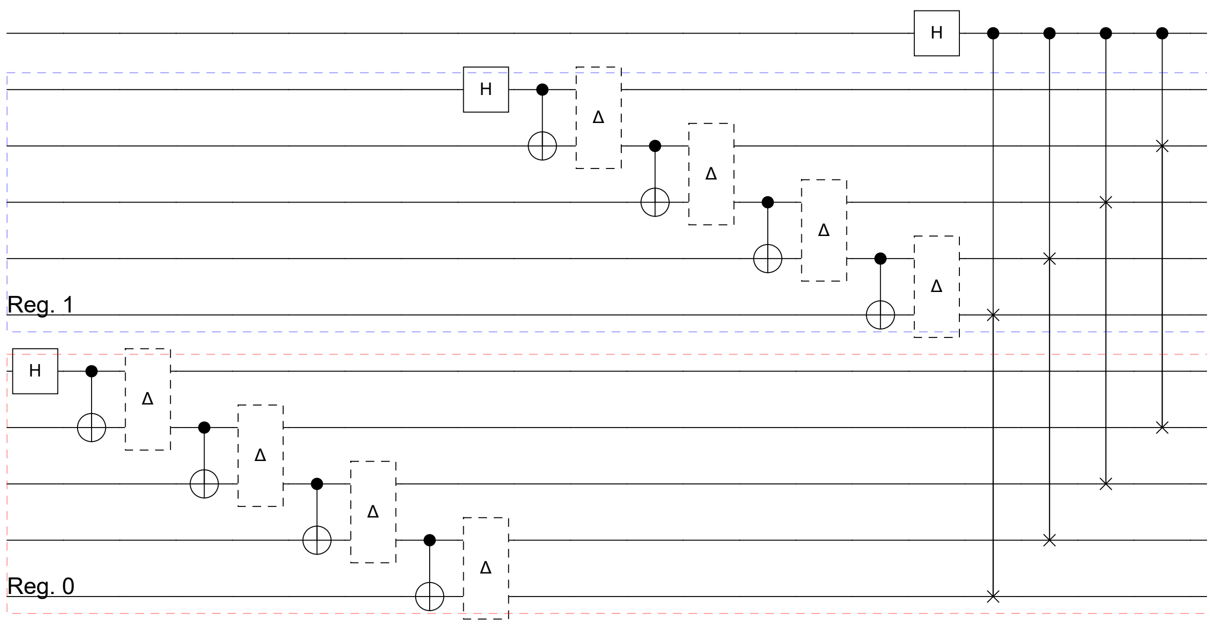
In[ ]:=
```
idealExpectation = 1.; (* observable's known ideal expectation value is 1 *)
unmitigatedError = Abs[CalcExpecPauliSum[ρ, observable, ϕ] - idealExpectation]
```

Out[ ]= 0.0419888

Simulate derangement circuits: virtual distillation of 2 noisy GHZ states

Set up derangement circuit with two input GHZ circuits (for measuring the expectation value $\langle X_0\ X_1\ X_2\ X_3\ X_4 \rangle$)

In[ ]:=
```
derangementCirc = D2[numQs, observable];
fullCirc = Join[CopiesOfCircuit[noisyGHZcirc, 2, numQs], derangementCirc];
DrawDerangementCircuit[fullCirc, numQs, 2]
```

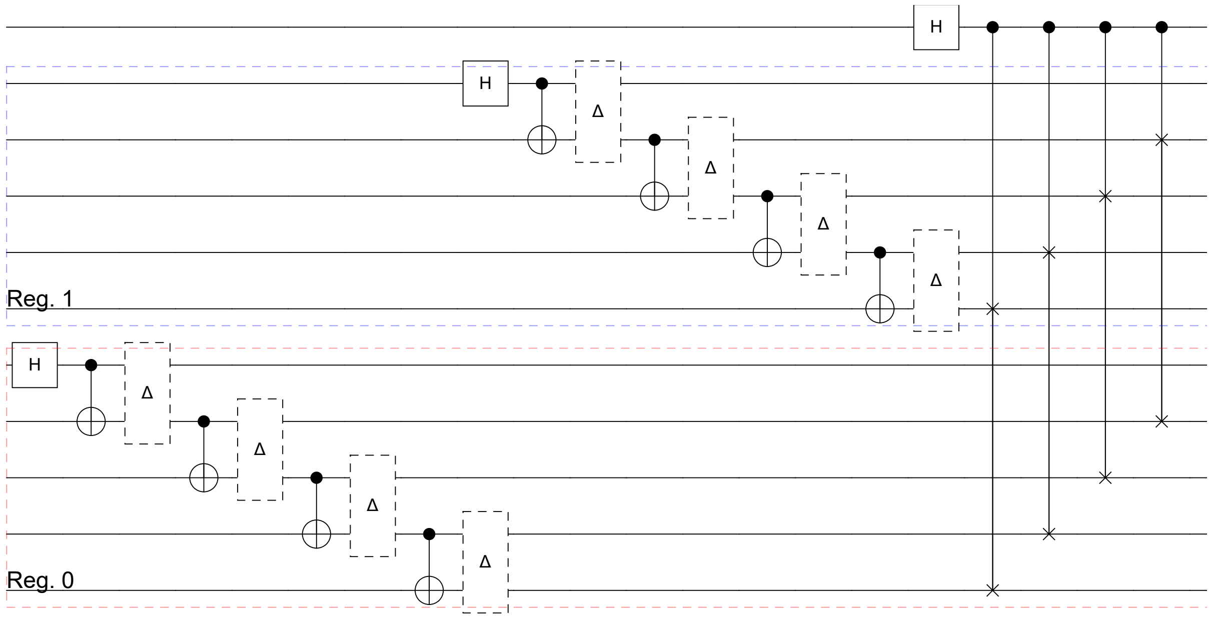
Out[ ]=


Simulate the circuit in QuESTlink and compute the probability on the ancilla qubit as $prob_0 = Tr[\rho^2\ X_0\ X_1\ X_2\ X_3\ X_4]$

In[]:=
```
ρ = CreateDensityQureg[totNumQ];
ApplyCircuit[fullCirc, InitZeroState@ρ];
prob0 = 2 CalcProbOfOutcome[ρ, 2 * numQs, 0] - 1;
```

Set up the circuit that measures the expectation value of the identity operator (required by the ESD/VD technique for correct normalisation)

In[]:=
```
derangementCircId = D2[numQs, Id0];
fullCircId = Join[CopiesOfCircuit[noisyGHZcirc, 2, numQs], derangementCircId];
DrawDerangementCircuit[fullCircId, numQs, 2]
```

Out[]=


Simulate the circuit in QuESTlink and compute the probability on the ancilla qubit as $prob'_0 = Tr[\rho^2]$

In[]:=
```
ApplyCircuit[fullCircId, InitZeroState@ρ];
prob0Prime = 2 CalcProbOfOutcome[ρ, 2 * numQs, 0] - 1;
```

The mitigated expectation value is computed via the ratio $prob_0 / prob'_0$

In[]:=
```
mitigatedExp = prob0 / prob0Prime
```

Out[]= 0.999832

Observe that the mitigated error is significantly smaller than the unmitigated one

In[]:=
```
errorWithDerangement = Abs[mitigatedExp - idealExpectation]
unmitigatedError
```

Out[]= 0.000168041

Out[]= 0.0419888

Try to vary the noise rate ϵ above and see how it changes the mitigated and unmitigated errors (see plot in the advanced notebook)