Timing Model for Ion-Trap Quantum Architectures

Alexander Hankin and Mark Hempstead
Tufts University, Medford, MA, USA
alexander.hankin@tufts.edu
Mentor: Kenneth Brown, Duke University
Quantum computers are here

- Quantum information theory has been steadily and stealthily progressing over the last few decades
- Quantum algorithms can have lower big O complexity than classical (e.g., prime factorization)
- Today’s quantum computers have 10s of “qubits” and 100s of gates
- Industry is predicting dramatic scaling
  - Google has predicted 1000s of qubits in the next 10 years
Types of quantum computers

Superconductors
- Google
- IBM

Photonics
- PsiQuantum

Quantum Dots
- Intel

Ion-Trap
- Honeywell
- IONQ
How can we evaluate this?

• Researchers currently have access to limited size quantum computers
• We must rely on models and simulation for studying larger-scale problems
• Models and tools are starting to emerge:
  – IBM: QisKit
  – Microsoft: Azure Quantum
  – Amazon: Braket
• A couple issues:
  – Mostly *functional* models (no timing, energy, etc.)
  – Less coverage for certain types of QCs (e.g., ion-trap)

Our contribution: we propose a timing model for ion-trap architectures
Ion-trap architecture (1/3)

- Building block is a “chain”
- Chain contains trapped ions which are the qubits
- Presently, the maximum number of ions in a chain is 32
Ion-trap architecture (2/3)

- To scale the number of qubits, chains are linked together to form multiple ion chains.

- Ion-trap QCs provide:
  - all-to-all connectivity between qubits within a chain, and
  - all qubit pairs are equally good.
• But, connections between multiple chains have a longer latency
• This is due to the optics required over free-space paths
• We call it a “weak link”
Design Goals

• From proposal:
  – Understand the effect of weak links on latency as number of qubits and number of gates scales up
  – Python-based
  – Extendability, usability, and flexibility
  – Easy to script and create batch jobs
Circuit description

- User specifies circuit depth and the number of 1- and 2- qubit gates
- Assume that each layer in a $d$-depth circuit is uniform
- Compute gate latency for 2-qubit gates based on whether a weak link is involved

<table>
<thead>
<tr>
<th>parameter</th>
<th>meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>$q$</td>
<td>number of 1-qubit gates</td>
</tr>
<tr>
<td>$p$</td>
<td>number of 2-qubit gates (random qubit pairs)</td>
</tr>
<tr>
<td>$d$</td>
<td>circuit depth</td>
</tr>
<tr>
<td>$\delta$</td>
<td>time for 1-qubit gate</td>
</tr>
<tr>
<td>$\gamma$</td>
<td>time for 2-qubit gate inside chain</td>
</tr>
<tr>
<td>$\alpha\gamma$</td>
<td>time for 2-qubit gate between chains</td>
</tr>
</tbody>
</table>
Base timing expression

\[ t \propto d(q\delta + p\Gamma) \]

\[ \Gamma \equiv P_{wi} \alpha \gamma + (1 - P_{wi})\gamma \]

\[ P_{wi} = \frac{1}{\binom{32}{2} + 1} \]

\( t \) is overall execution time
\( \Gamma \) is equivalent to the latency of an arbitrary 2-qubit gate
Model changes since proposal

• Multiple circuit layers
• Ability to sweep number of weak links
• Ability to sweep number of chains and ions per chain
• Graph representation
Graph Representation
A minimum working example

- Circuit summary:
  - # qubits: 5
  - 1-qubit gates: 1
  - 2-qubit gates: 4
- Let’s assume 4-qubit chains
- So number of chains = ceil(number of qubits/qubits per chain) = 2
- Need to make 1 cut
- A couple possible cuts shown right
Effect of weak link

- The sum of the edge weights between chains is the number of weak links ("cut size" in graph theory)
  - This must be less than the number of available weak links (i.e., 2 chains => 2 possible weak links available)
  - Could decide to use more chains to have more weak links

- Chains are currently assumed to execute serially
Areas for improvement

• Every node (qubit) should have at least one edge

• Need to keep track of order of operations

• Between-chain parallelism (weak link still serial)
Status of the code

• Using python graph API called NetworkX
• Built out simple examples
• Have started working on scaling up to arbitrarily large random circuits
• Can scale to arbitrarily large random circuits and sweep parameters like:
  – Number of gates
  – Number of ions per chain
Simulation overview (pseudocode)

1. build graph of circuit based on input parameters  # (e.g., number of qubits, number of gates, chain-size, etc.)
2. compute minimum number of chains needed  # based on number of qubits and chain size
3. find all possible ways to place qubits into chains (cuts) given number of available weak links (may not be possible) and randomly select one placement
4. compute \text{sum}(\text{cut sizes})  # this is the number of weak links needed for this placement
5. while \text{sum}(\text{cut sizes}) > number of available weak links:  # keep trying until valid placement
   6. find all possible ways to place qubits into chains (cuts) given number of available weak links
      (may not be possible) and randomly select one placement
5. compute \text{sum}(\text{cut sizes})
7. compute timing (serial)
Opportunities for optimization

• May be beneficial not to use minimum number of chains
• Placing qubits into chains randomly given weak link constraint can be computationally expensive
• How qubits are placed can affect the number of weak links
### Configured Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ion per chain</td>
<td>8</td>
</tr>
<tr>
<td>Num qubits</td>
<td>20</td>
</tr>
<tr>
<td>Num 1-qubit gates</td>
<td>0</td>
</tr>
<tr>
<td>Num 2-qubit gates</td>
<td>30</td>
</tr>
<tr>
<td>Latency for 1-qubit gate</td>
<td>1us</td>
</tr>
<tr>
<td>Latency for 2-qubit gate</td>
<td>100us</td>
</tr>
<tr>
<td>Penalty for weak link</td>
<td>100us</td>
</tr>
</tbody>
</table>

### Computed Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Num chains</td>
<td>3</td>
</tr>
<tr>
<td>Num weak links available</td>
<td>3</td>
</tr>
<tr>
<td>Num weak links used</td>
<td>2</td>
</tr>
</tbody>
</table>

Total time [ms]: 3.2
Example study

• Fix circuit (same circuit as previous slide)
• Increase chain size (4, 6, and 8)
  – Generally decreases number of weak links
  – Having more weak links and entangling earlier could be useful though
• Average of 5 runs per chain size
Future Work

- Make 1-qubit gates a node attribute and make sure all qubits have at least 1 edge
- Formulate placement of qubits as optimization problem
- Formulate weak link choice as optimization problem
- Model real applications like BV and QAOA in addition to random circuits
- Study how performance scales as number of qubits/gates scales
- Add in error correction
Acknowledgements

Kenneth Brown, Duke University
Peter Love, Tufts University
Questions

alexander.hankin@tufts.edu
Ion-Trap in the Market

- IonQ first Quantum computing company set to become public (as of 2021)

- The market is betting on IonQ being able to scale up their architecture
Background: qubits (1/2)

- Quantum bit ("qubit")

\[ |\psi\rangle = a |0\rangle + b |1\rangle \quad \text{and} \quad |a|^2 + |b|^2 = 1 \]

- Qubit state, \( \psi \), is related to two basis states 0 and 1 by complex scalars, \( a \) and \( b \).

- This relationship is called a superposition and the state, or qubit, \( \psi \), is said to be in a superposition of the two basis states with probability amplitudes of \( a \) and \( b \). Born’s rule says that the squares of \( a \) and \( b \) sum to 1.
Background: qubits (2/2)

\[ |\psi\rangle = a |0\rangle + b |1\rangle \quad \text{and} \quad |a|^2 + |b|^2 = 1 \]

- When the qubit is measured, the superposition collapses and we observe either a 0 or 1 (a must now be equal to 1 and b equal to 0, or vice versa)
Background: quantum gates (1/3)

- In classical computing, NAND is universal.
- In quantum computing, need more than one gate:

  ![Qubit Gates](image)

  - Qubits and quantum gates can be represented as vectors and matrices
Background: quantum gates (2/3)

\[ |\psi\rangle = a |0\rangle + b |1\rangle \quad \text{and} \quad |a|^2 + |b|^2 = 1 \]

- 0 can be written as \([1 \ 0]^T\) and 1 can be written as \([0 \ 1]^T\)
- So, \(\psi\) can be rewritten as \([a \ b]^T\)
- The gates from the previous slide can be represented as:

\[
H = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}, \quad S = \begin{pmatrix} 1 & 0 \\ 0 & i \end{pmatrix}, \quad T = \begin{pmatrix} 1 & 0 \\ 0 & e^{i\frac{\pi}{2}} \end{pmatrix}
\]
$|\psi\rangle = a |0\rangle + b |1\rangle$ and $|a|^2 + |b|^2 = 1$

$H = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}, \quad S = \begin{pmatrix} 1 & 0 \\ 0 & i \end{pmatrix}, \quad T = \begin{pmatrix} 1 & 0 \\ 0 & e^{i \frac{\pi}{4}} \end{pmatrix}$

$H |\psi\rangle = \frac{a+b}{\sqrt{2}} |0\rangle + \frac{a-b}{\sqrt{2}} |1\rangle$