Qubit allocation for quantum circuit compilers

Nov. 10, 2017
JIQ 2017

Marcos Yukio Siraichi
Vinícius Fernandes dos Santos

Sylvain Collange
Fernando Magno Quintão Pereira
Compilers for quantum computing

• The first generation of “usable” quantum computers is here
  - e.g. IBM Quantum Experience
  - Enables experimental computer science

• Existing and near-future architectures:
  - 10s to 50 qubits
  - No error correction
  - Low-level constraints on circuits:
    set of gates, qubit connectivity

• Need compilers of circuits down to low-level gates
  - Many differences from classical compilers
Focus: the qubit allocation phase

- Map logical qubits to physical qubits
  - Need to meet hardware constraints: connectivity between physical qubits
  - Transform circuit to fit on given quantum computer
- Minimize runtime and gate count to minimize noise

Software: circuit on logical qubits

Hardware: physical qubits
Agenda

- The qubit allocation problem
- An exact algorithm
- A greedy heuristic
- Comparison of allocation accuracy
- Future directions
Level of abstraction: quantum circuits

Input: reversible quantum circuits described at gate level

- Between initialization and measurement: unitary gates only
- After decomposition into single-qubit and CNOT gates
- Expressed in QASM language

```qasm
qreg l[2];
creg c[2];
x l[0];
h l[0];
cx l[0] l[1];
t l[1];
measure l[0] -> c[0];
measure l[1] -> c[1];
```
Limited-connectivity quantum computer

Target: superconducting qubit based quantum computers

- Constraints on which qubits are allowed to interact
  - e.g. IBM QX2, 5 qubits
  - e.g. IBM QX5, 16 qubits
The qubit assignment problem

Can we label logical qubits with physical qubits so that all gates obey machine connectivity constraints?

- “Easy part” of qubit allocation
- Already NP-Complete (subgraph isomorphism)

In practice, most circuits will need transformations to “fit” the connectivity graph
Circuit transformation primitives

- **CNOT reversal**

- **Bridge**

- **Swap**

Effect on dependency graph (assuming no other dependency)

Change mapping!
The qubit allocation problem

- Qubit allocation with swaps only
  - Minimize number of swaps inserted
  - NP-Complete (Token Swapping problem)

- General qubit allocation problem
  - Use CNOT reversal, bridge and swap
  - Minimize cost of circuit transformations

- Subproblem of depth $d$
  - Circuit depth := number of CNOT gates
  - Starts and ends with logical-to-physical qubit mappings

- Slice of depth $d$
- Slice of depth $d+1$

initial $L \rightarrow Q$ mapping

final $L \rightarrow Q$ mapping
Exact algorithm: dynamic programming

- Assume we know partial solutions of depth $d$ with final mapping $M$ and their cost, for all $M$
- Compute solutions of depth $d+1$ with final mapping $M$, for all $M$
  - Select the (solution of depth $d$) + (permutation) that minimizes cost

<table>
<thead>
<tr>
<th>Solutions of depth d: cost</th>
<th>Cost of permutation from Mi to Mj</th>
<th>Solutions of depth d+1</th>
<th>...</th>
<th>Solutions for full circuit</th>
</tr>
</thead>
<tbody>
<tr>
<td>M0 22</td>
<td></td>
<td>∞</td>
<td></td>
<td>29</td>
</tr>
<tr>
<td>M1 26</td>
<td>3</td>
<td>25</td>
<td>6</td>
<td>∞</td>
</tr>
<tr>
<td>M2 20</td>
<td>10</td>
<td>26</td>
<td>3</td>
<td>28</td>
</tr>
</tbody>
</table>

Unfeasible solutions have cost $\infty$

Final solution: minimum cost
Exact algorithm: dynamic programming

Complexity $O((n!)^2 \times m)$ for $n$ qubits and circuit depth $m$

Suitable for $\leq 8$ qubits

Gives an optimal reference to compare heuristics
The Weighted Partial Matching heuristic

1. Find a good initial mapping
   - Favor most-often used dependencies

2. Extend the mapping, transforming circuit as needed
   - Perform swap when it can be amortized
   - Use CNOT reversal on backward edges
   - If we have a 2-step path through another qubit, use a Bridge
   - If all else fails, insert as many swaps as needed
Results: cost on IBM QX2, actual circuits

- Heuristic outperforms state of the art in this 5-qubit configuration
- Exact algorithm shows heuristics have potential for improvement
  - Both in initial mapping choice and migration strategy
Next steps for qubit allocation

- Improved heuristics
  - Seek run-time vs. accuracy tradeoffs
  - Specialize for regular quantum computer structures
  - Take advantage of quantum circuit properties: spacial, temporal locality

- Coordinate circuit optimization with qubit allocation
  - e.g. optimize away redundant Hadamard gates when placing reverse CNOT next to H gates

- Recycle *ancillae* qubits
  - Qubit with completely known equal state are interchangeable (e.g. $|0\rangle$)
  - Static analysis to find qubit equalities?

- Optimize for device characteristics
  - Different qubits and couplings have different noise levels
Compiler optimization for quantum circuits

- Mapping high-level gates to hardware-supported gates
  - Single-qubit gates: accuracy/cost tradeoffs
  - Toffoli gates: exploit freedom on relative phase
- Time/space tradeoffs
  - Adapt number of ancillae qubits to resource availability
- Formalization
  - Which semantics for quantum programs and quantum computers?
  - Which intermediate representation for quantum circuits?
  - Correctness proofs of compiler transformations