Scott Mionis Franz Franchett
Jason Larkin
https://github.com/spiral-software/spiral-software
Our Approach: SPIRAL For more about the classical SPIRAL compilation system, visit http://spiral.net

SPIRAL Quantum Compiler


Defining a Circuit
A top-level circuit definition:


Embedding a Transform


## Optimization

$r t_{\text {opt }}($ arch $)=\underset{r t, \text { arch }}{\arg \min \operatorname{Cost}(\text { Rewrite }(\operatorname{Breakdown}(r t, \text { circ, } \text { arch })))}$
Breakdown(tr, circ): applies breakdown rule sequence $r t$ to transform circ
Rewrite(c) : applies rewrite rules to simplify expression $c$
Costt $t$ ) : returns the cost of gate expression $t$
arch: The qubit topology of the architecture, as an adjacency matrix

Unparse the circuit as a OASM program
Actally a factorization of subgrops of the permutation group

SPIRAL System Overview


## The Embed Operation

Architecture is Pruned in recursive calls
$\qquad$
$\qquad$
QEmbed ( $[0,3,1]$, arch, qFT) * qEmbed $[9,2]$, arch, qCNOT)



## Rewrite Rules

- Perform direct or conditional substitutions to collapse gates and smplify circuit descriptio

Rewriting Rules
\# Flatten Tensors
ex) Tensor (Tensor(H2, I2), Tensor (X2, Y2 ) ) = Tensor(H2, I2, X2, Y2
\# Combine Tensors
ex) Tensor (H2, I2) *ensor (I2, H2) $=>$ Tensor (H2, H2)
\# Combine Reorder

The "Best" embedding has adjacent Reorder steps that reduce

## First Results

We tested SPRRAL against IBM's Quiskit optimizer with Cost(t) = \#(CNOT)(t) - Executed final QASM code on IBM's Tenerife and Bogota devices


[^0]
[^0]:    Next steps are to add breakdown-rwle heuristics, additional rewriting rules

