Swap Reduction (%) by layout

This bar chart depicts the average percentage of swap gate reduction between BasicSwap and LookaheadSwap and SabreSwap and LookaheadSwap grouped by layout run on circuit size of 15. There are 13 layout and 15 algorithms to test for. The swap basic – lookahead all shows positive percentange, meaning that all of the implemented Lookahead swap are smaller than the BasicSwap. However all of sabreswap outperform the lookahead swap shown in negative percentage.

As a baseline, full\_20\_1 shows a fully connected graph in which every physical qubits are connected to each other, therefore, it does not require any additional swap gate because it does not have connectivity constraint. In average, the lookaheadswap can reduced the swap gate compared to basicswap around 50%, with a lower performance for full\_10\_2 and all grid layout. The bars shown in different colour shows that the layout does not run all algorithm and its number is shown in parentheses.

In contrast, the performance of sabre swap is better for all layout, with a significant reduction for full\_10\_2 and t\_horizontal\_5\_4 layout. The reduction varied between 100% to 250%, with a smaller number if the layout are line\_20\_1 ad full\_7\_3.

Depth Reduction (%) by Layout

Circuit depth reduction shows positive percentage for comparison basic-lookahead and sabre-loookahead. For basic-lookahead section, line, grid layout, t\_horizontal and t\_vertical layout shows similar performance with reduction around 65%. For full layout, it may show as fluctuative, but the number of algorithms run on full\_5\_4 is only 3, so it does not run all the benchmark algorithm, this value is almost similar with ring layout, where it get reduction around 50% and become lower at 30% if the group is increased.

Circuit depth for sabre-lookahead also shows positive value, although far lower than the basic-lookahead. The gate reduction for sabre-lookhead is nearly similar within range of 20-30% for almost all layout, except full\_5\_4 that only perform 10% for 3 algorithms.

This bar chart categorize the layout based on its group to show the swap reduction percentage. Group 1 is used as the benchmark where the layout is monolithic, without distributed system. The percentage for swap basic – lookahead are similar for every group with the tier of ring, full, then grid. For swap sabre-lookahead the result varied where at group 2 the performance of full\_10\_2 Is the worse, followed by ring\_10\_2 and grid\_9\_2. In group 3, the data is not complete because full can only run 9 algorithms, but in this case ring\_7\_3 perform better by than grid\_8\_3, for the last group 4, the worst reduction is by t\_horizontal as the highest percentage, ring\_5\_4 and grid\_6\_4 almost the same with a difference by 10% at around -165% for grid\_6\_4, and full\_5\_4 has a better chance with added percentage round 110% from sabreswap.

This bar chart illustrates when the layouts are categorized by group, as a benchmark group 1 of full (interconnected physical qubits) and line as a long chain of physical qubits.

For all, depth basic – lookahead are higher than depth sabre-lookahead, with group 2 where grid\_9\_2 perform at 65% depth reduction followed by full\_10\_2 and ring\_10\_2, and for group 3 ring\_7\_4 perform better at 50.67% than full\_7\_3 at 43.51%. Group 4 compared 5 layouts among full, grid, ring, t\_horizontal, and t\_vertical. All layouts have depth reduction at more than 65%. Although ring only runs 5 algorithms, it still yield far worse percentage at 27.41% and full which only run 3 algorithms at around 11.38%..

This bar chart depict the number of algorithms run as a benchmark with circuit size of 5, 10, and 15 grouped by layout. The layout contains approximately 20 qubits. The layout format is layout name as in Section 3 Methodology, number of qubits, followed by number of group. Most of the layout can run all the benchmark algorithms, with the exception of full and ring that show decreased performance.

For benchmark with circuit size of 5 qubits, ranging from ghz with the least number of gates of 7, to portfoliovqe with the 310 gates, all of the layout can run the algorithms denotes by 15 counts as the number of algorithms run on each layout. The exception is on ring\_5\_4 where qnn cannot be run on the layout. Next is for benchmark with circuit size of 10, ring with a group of 3 and a group of 4 cannot run the benchmark, with only 10 counts for the former and 7 for the latter. When the circuit size is increase to 15, two layouts show significant degradation with full\_7\_3 and full\_5\_4 hits interaction mapping layout timeout, while ring\_7\_3 and ring\_5\_4 manages to create the mapping layout, but get error timeout in swapping over the gate.

From this chart, it can be inferred that ring show the worst performance followed by full, and others are works as usual.

This chart is the flipped of previous chart where now the number of layout is counted and it is group by the algorithm. The algorithm is followed by the number of gates for each benchmark and it is sorted from the smaller gate to the largest vqe. It can be seen that the layout performance worsen when the benchmark is increased up to 15 circuit size.

The first one is all benchmark on 5 qubits can run all the algorithms, except qnn on ring\_5\_4. For the next benchmark with 10 qubits, several algorithms namely qft, qftentangled, several random algorithms failed at ring\_5\_4 layout as well. The rest from qnn to portfoliovqe fail at both ring\_5\_4 and ring\_7\_3. When the circuit size is increased by 5 to 15 qubits, the only algorithms that can fully run on all of the layout are dj, graphstate, and qaoa. While the others faced timeout in either Interaction Mapping layout or the lookahead swap routing. The algorithms su2random, qnn, portfolioqaoa, and random with gates more with gates more than 650 failed at full\_7\_3, full\_5\_4, ring\_7\_3, and ring\_5\_4.

In this following section, comparing several different layout for several representative algorithms

GHZ as the algorithm with smallest number of gates, shows the highest lookahead additional swap at layout ring, when it is divided by 2,3, and 4 group. SabreSwap also perform worse than BasicSwap.

The second Deutsch-Josza (DJ) algorithm shows that swap lookahead as a similar number of swap as sabre and it gets the highest number of swap gate when using BasicSwap

The next algorithm of Graph state with 150 gates illustrates similar performance between BasicSwap and LookaheadSwap for all layout at the range of additional 80 to 120 gates, as opposed to SabreSwap that in the approximate of only around 20 additional swap gates.

This bar chart is almost similar for algorithm Wstate (253) and VQE (253), where the highest falls layout ring at more than 120 gates. Most of the number for BasicSwap is 0 for the the first 5 layouts. The additional swap gates are better for full, grid, and t\_horizontal layout at less than 50 swap gates, and worse for ring and t\_vertical layout with more than 80 additional swap.

This algorithm of portfolioqaoa (1260) the largest 2505 operational gate is one of the representative of similar results for other algorithms, namely, Portfolio VQE (2505), QNN (914), Su2random (675), twolocalrandom (615), realamprandom (615), qftentangled (608), and qft (591). For LookaheadSwap, the additional swap gates is around 1000 gates, however, it fails for 4 layout that is full\_7\_3, full\_5\_4 because of interaction mapping layout timeout, and ring\_7\_3 and ring\_5\_4 because of swap routing timeout. SabreSwap is slightly lower than the lookahead at below 1000 additional gates, and basic swap is the worst with more than 8000 qubits for line\_20\_1 configuration, and fluctuative for other layouts.