QUBO scaling

A deeper look into:

1) Time to solve of single sub-QUBOs
2) Time to reach 95% of the ground state energy
3) Success of solving a sub-QUBO

with respect to the sub-QUBO size

David Spataro
Hamburg, 19.01.2023

Motivation

Bigger quantum devices = better/faster results ?

For our project we need to investigate how the **global optimisation algorithm** performs with respect to the sub-QUBO size. Possible positive scenarios could be:

1) The QUBO is solved faster
2) The QUBO is solved better (we find a lower energy)

if the sub-QUBO size is increased.

Motivation

Bigger quantum devices = better/faster results ?

Looking at the energy levels is easy, but looking at the time to solve the QUBO is more difficult.

Possible problems are:

1) We measure the energy always after one complete impact list iteration, consisting of more than 100k sub-QUBOs in the case of xi=7. Having one more or one less iteration will increase or decrease the measured time by several hours. So we end up with large error bars when averaging over more than one BX.
2) The time needed to solve a sub-QUBO depends directly on xi, since with the impact list we collect possible conflicts and connections from outside the sub-QUBO and add it to the linear term of the qubo. For higher xi this needs longer, because there are simply more connections and conflicts to collect.

Motivation

Bigger quantum devices = better/faster results ?

3) The actual time for solving the sub-QUBO is only a part of the measured time. The qubo has to be created and transformed into suitable form for the simulation / real device, the quantum circuit has to be transpiled, simulation has to be computed, etc.
4) Linear terms of the first sub-QUBOs. They are heavily driven by the terms outside the respective sub-QUBO. And since we start with the highest impact, and a triplet has in general way more conflicts than connections, the first sub-QUBOs have the result [0, 0, 0,..,0].

   This is also the reason why the “poor man’s approach” or the “bit-flip” optimisation has the same result as the Eigensolver/VQE approach. Another problem is ,that it’s super easy to find the ground state of the first sub-QUBOs in the impact list because of the huge linear terms, and then it becomes more difficult.

Motivation

Approach to answer the questions / solve the problems

1) All used sub-QUBO sizes (5, 7, 10, 12, 16…) result in the same final energy

   → impact list problem because of linear sub-QUBO terms!

2) Comparing solving time of the QUBO:
   a) Define an energy level which should be reached (95% of the truth ground state energy)
   b) Compare average time to reach the solution, to solve a sub-QUBO, and the solving efficiency for sub-QUBO's
3) We would like to see, that the
   a) average solving efficiency stays constant >99% for the sub-QUBOs
   b) average time to solve a sub-QUBO x number of sub-QUBOs (= impact list iteration) decreases with the sub-QUBO size
   c) number of iterations needed to get to the 95% level decreases with the sub-QUBO size

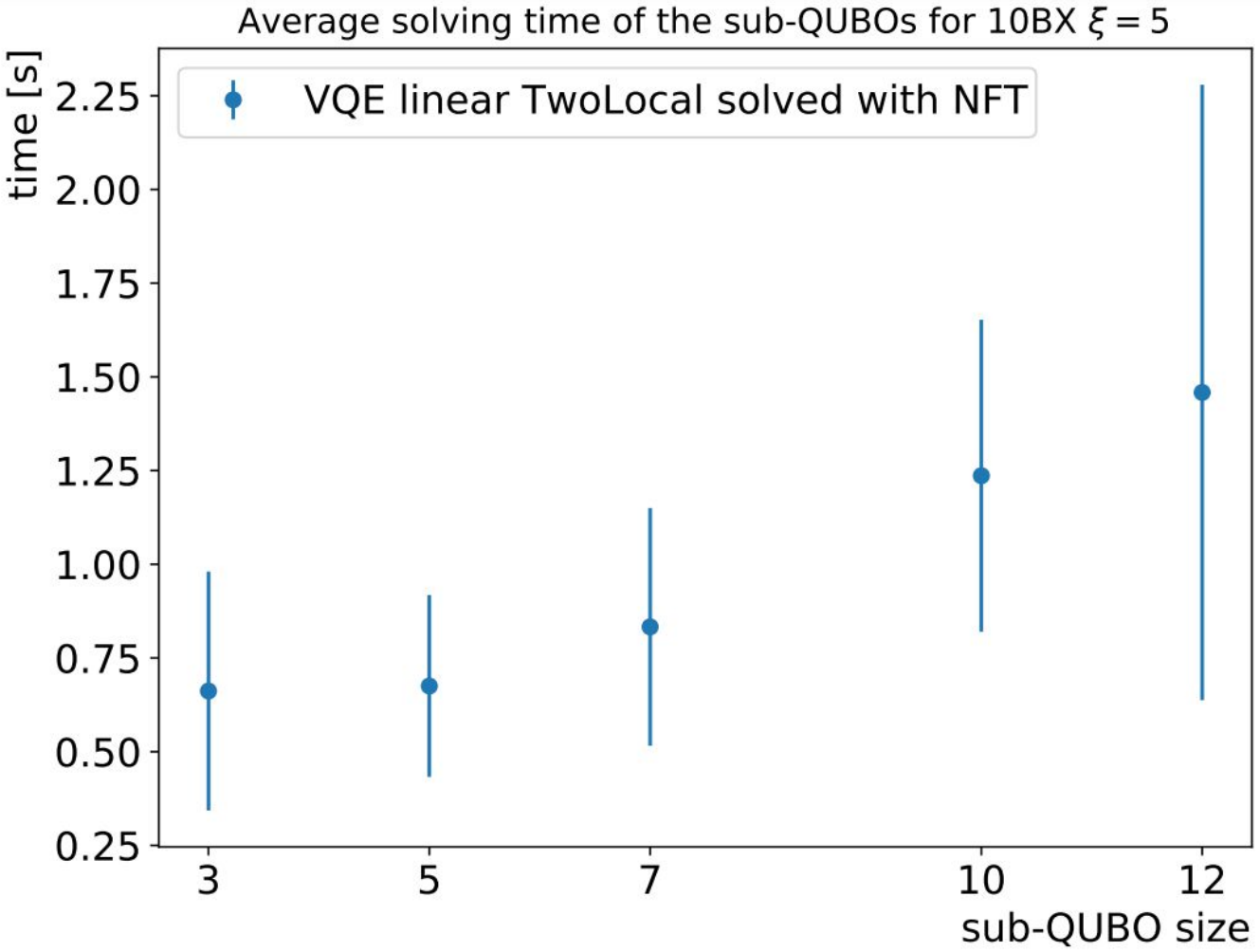
Summary: impact list algorithm time consumption

Actions involved when speaking of “one iteration”

1) Ordering list of triplets by impact
2) Iterating over first X entries where X is the chosen sub-QUBO size and pick up all information about conflicts and connections from triplets outside the sub-QUBO
3) sub-QUBO is solved via VQE, computational time driven by:
    a) circuit transpiling
    b) creating tensors / operator representation
    c) device sampling
    d) NFT algorithm
4) Energy evaluation of the system

Results

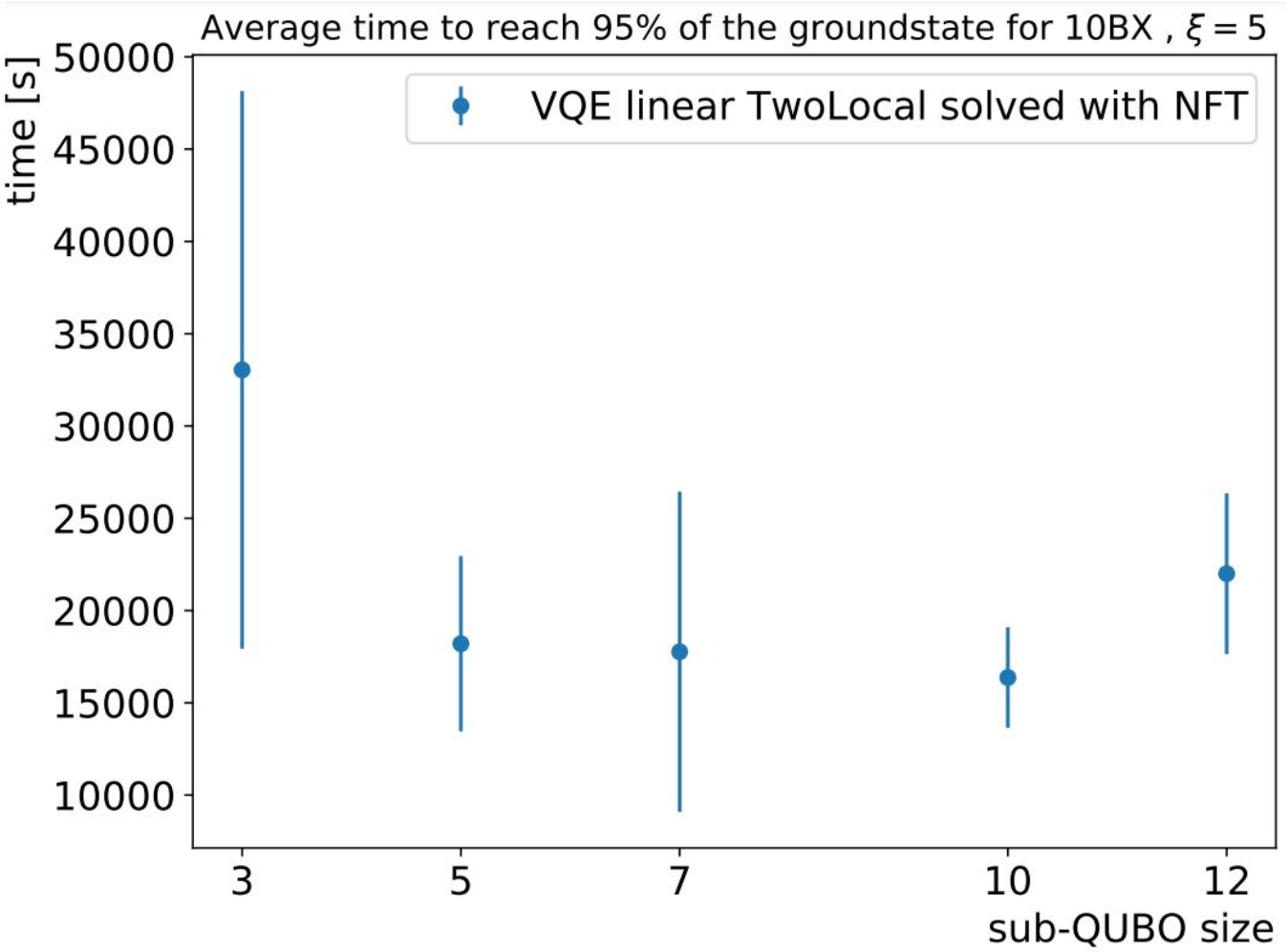
xi=5, 10BX



Average time of solving the sub-QUBOs has large error bars, this is most likely caused by the “easy” sub-QUBOs at the beginning vs the “difficult” ones towards the end.

Results

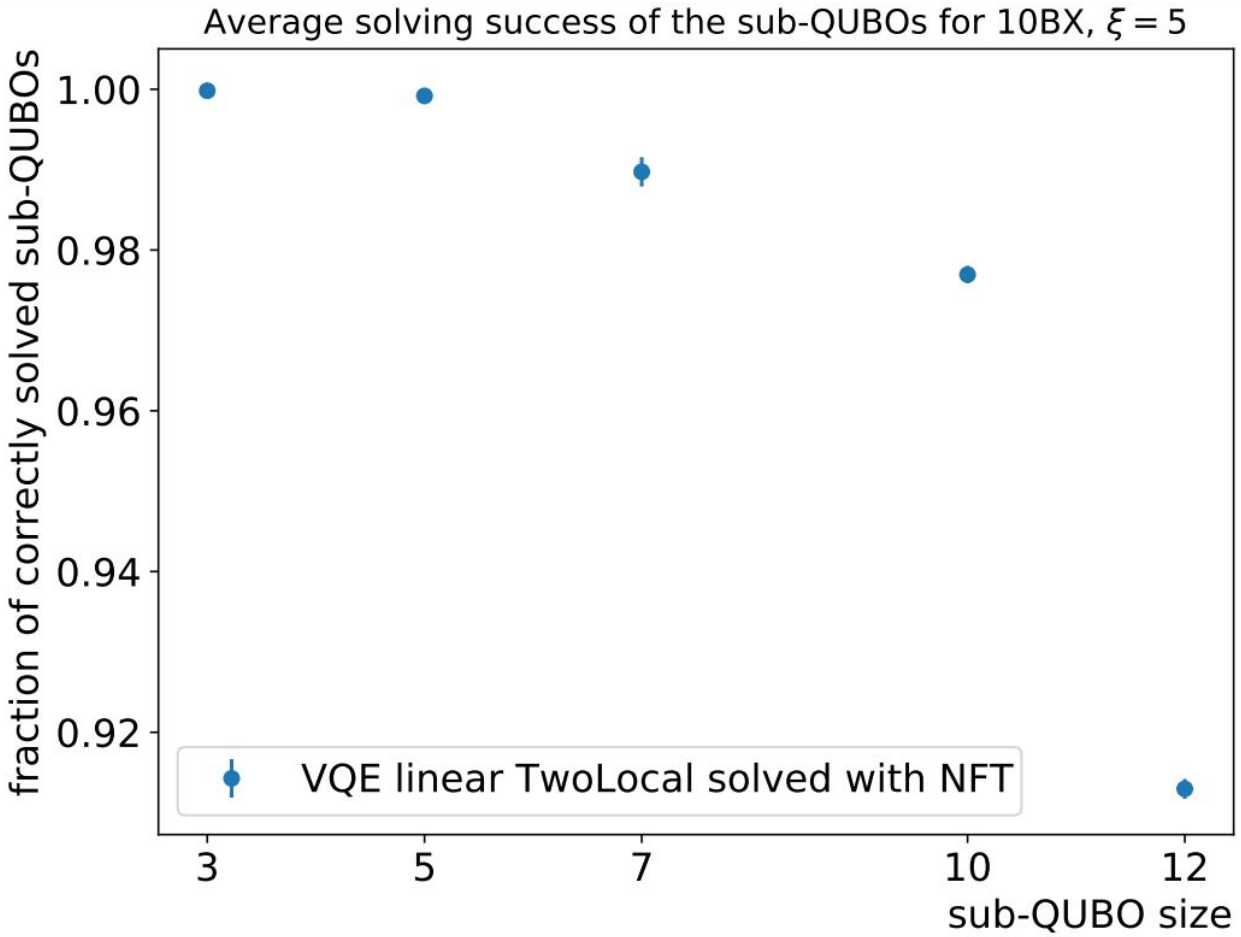
xi=5, 10BX



Average time in seconds for reaching 95% ground state energy. As before everything is compatible with everything…

Results

xi=5, 10BX



Average solving success for a sub-QUBO vs. sub-QUBO size.

Summary Results

The time until the solution reaches 95% of the ground state *might* depend on the sub-QUBO size, but due to large error bars everything is compatible with everything.

The sub-QUBO solving efficiency drops with the size of the sub-QUBO. For a size of 16, the 95% level was not reached, so this data could not be shown.

Next step: Optimise for sub-QUBO size of 12, too and use the new parameters to make a new calculation