Currently EVM processes transaction sequentially. While for the main net it is not a big deal since the blocks come slowly anyway, it is a problem for us at SKALE.

One would think that if transaction X modifies the state of a contract A and transaction Y modifies the state of contract B, then X and Y could be processed in parallel.

The problem is both A and B may call contract C. This couples the state of A and B and prevents parallel processing.

Here is how this can be solved:

When A wants to call C, instead of calling it directly, it creates a virtual transaction Z at the end of the block. This transaction calls C.

When all the original transactions are processed, there is a set of virtual transactions at the end of the block S1. These transactions are then ordered by hash, and processed as a virtual block B1. This can lead to another vritual block B2, which is in turn processed by EVM.

Ultimately the processing converges, creating an empty virtual block.

Note, that at every processing stage figuring out parallelism is easy. Since there are now subcontract calls, two transactions that call two different contracts and have two different senders can be processed in parallel.