Note to the paper of Onzecurrency Crowd:

At the time of writing [2] I wasn't aware of the existance of PBFT [1] consensus algorithm and this is a small note of comaprison.

Extract from PBFT algorithm [1]:
1. A client sends a request to invoke a service operation to the primary
2. The primary multicasts the request to the backups
3. Replicas execute the request and send a reply to the client
4. The client waits for f + 1 replies from different replicas with the same result; this is the result of the operation.

POCO algorithm of onzecurrrency:
1. A client sends a request to invoke a service operation to the coördinator
2. The coördinator sends the request to the (maximum 100) chosen ones
3. The coörinator and the chosen ones verify and execute the request and inform their part of the network, including the client
4. The client waits a reply from a replica/peer which might or might not arrive because of your chosen one's verification.

PBFT and POCO are very similar:

POCO uses the concept of chosen ones to inform the whole network while PBFT needs all of its replicas to reply to the client and that client will receive replies of all the replicas, so there are scalability issues. POCO doesn't have this single point of possible failure and should therefore scale better.

PBFT and POCO both are susceprible to the 51% Sybil attack.

Reference:
[1] Practical Byzantine Fault Tolerance: https://pmg.csail.mit.edu/papers/osdi99.pdf
[2] Onzecurrency crowd: https://github.com/nvrrdt/onzecurrency/blob/main/papers/whitepaper%20onzecurrency%20crowd.pdf