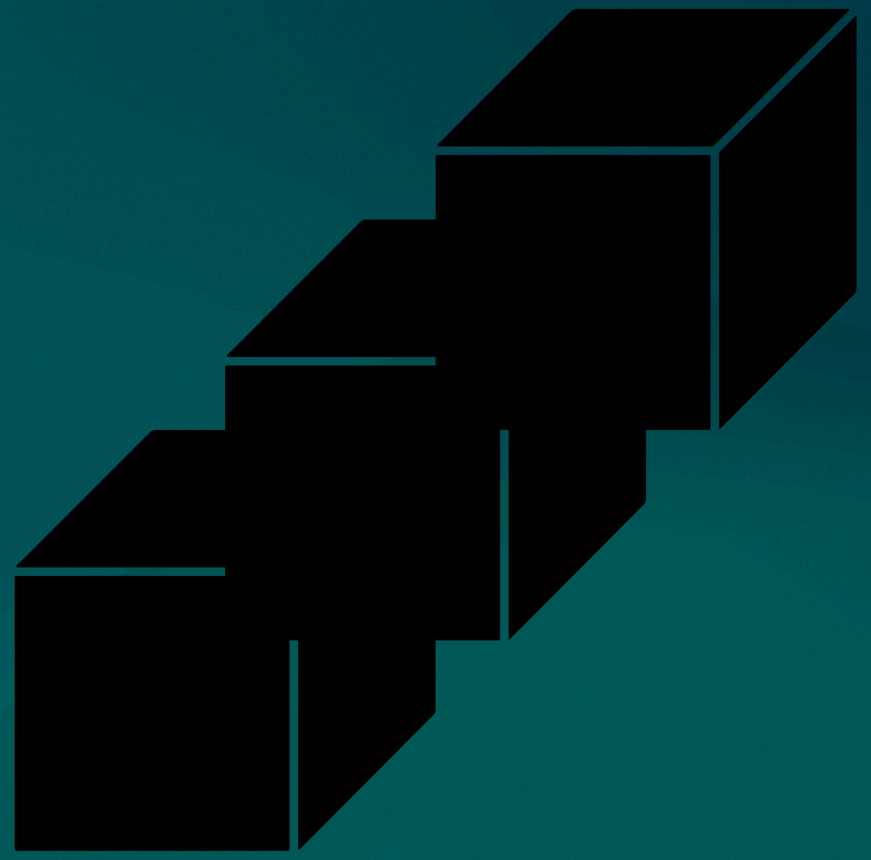
Blockchain network orchestration to improve performance

Оркестрация блокчейн сети для повышения производительности



Выполнил: Мазуренко Тимофей Матвеевич БПАД222

Научный руководитель: Янович Юрий Александрович

Relevance

01.

Rapidly evolving blockchain technology

02.

Performance of the network is crucial

03.

Improving the network performance even slightly leads to tangible difference

Hyperledger Fabric

Enterprise-grade, permissioned distributed ledger platform

Companies that use Fabric:

- Anthem - US health insurer
- Dubai authorities
- Home Depot
- ...

Network configuration parameters

Settings and values that define how a blockchain network operates and performs

```
# Maximum count of blocks stored in memory
maxBlockCountToStore: 10
# Max time between consecutive message pushes(unit: millisecond)
maxPropagationBurstLatency: 10ms
# Max number of messages stored until a push is triggered to remote peers
maxPropagationBurstSize: 10
```

Example of few parameters from Fabric source code

Benchmarking

- Act of running a set of standardised tests to assess various performance metrics
- Benchmarking tool: Hyperleger Caliper

Name	Succ	Fail	Send Rate (TPS)	Max Latency (s)	Min Latency (s)	Avg Latency (s)	Throughput (TPS)
readAsset	16777	0	566.9	0.05	0.00	0.00	566.9

Sample benchmark results

Problem with current state of benchmarking solutions

Benchmark execution steps:

1. Configuring network parameters
2. Rebuilding binaries
3. Deploying the network
4. Running the benchmark
5. Collecting benchmarking data
6. Shutting down the network
7. Repeat

Goal

Automate the process of performing multiple benchmarks on the given set of parameters with predefined values to test

Objectives

- Develop architecture
- Build a set of helper functions
- Build a final wrapper script
- Test the solution
- Visualise and analyse retrieved data

Components of developed solution

Function name	Language	Input parameters	Performed action
edit	Python3	Value, Parameter_key	Change parameters value
execute_benchamrk	Bash	-	Build binaries, deploy the network, execute benchmark
parse_results	Python3	Parameter_value, Output_path	Parse benchmark results from report.html
shut_down	Bash	-	Shut down the network
reset_core	Bash	-	Set all parameters to default state

Benchmark launch configuration

```
{
    "parameter_name_1": {
        "key": "one.two.three.parameter_name_1",
        "values": [10, 15, 20, 25, 30, 35, 40, 45, 50],
        "path_to_output": "path/to/output"
    },
    ....
}
```

Design of the wrapper script

```
1   read parameter.json
2   for parameter in parameter_names:
3       for value in parameter_values:
4           edit parameter value
5           execute benchmark
6           parse results
7       reset parameters to default state
```

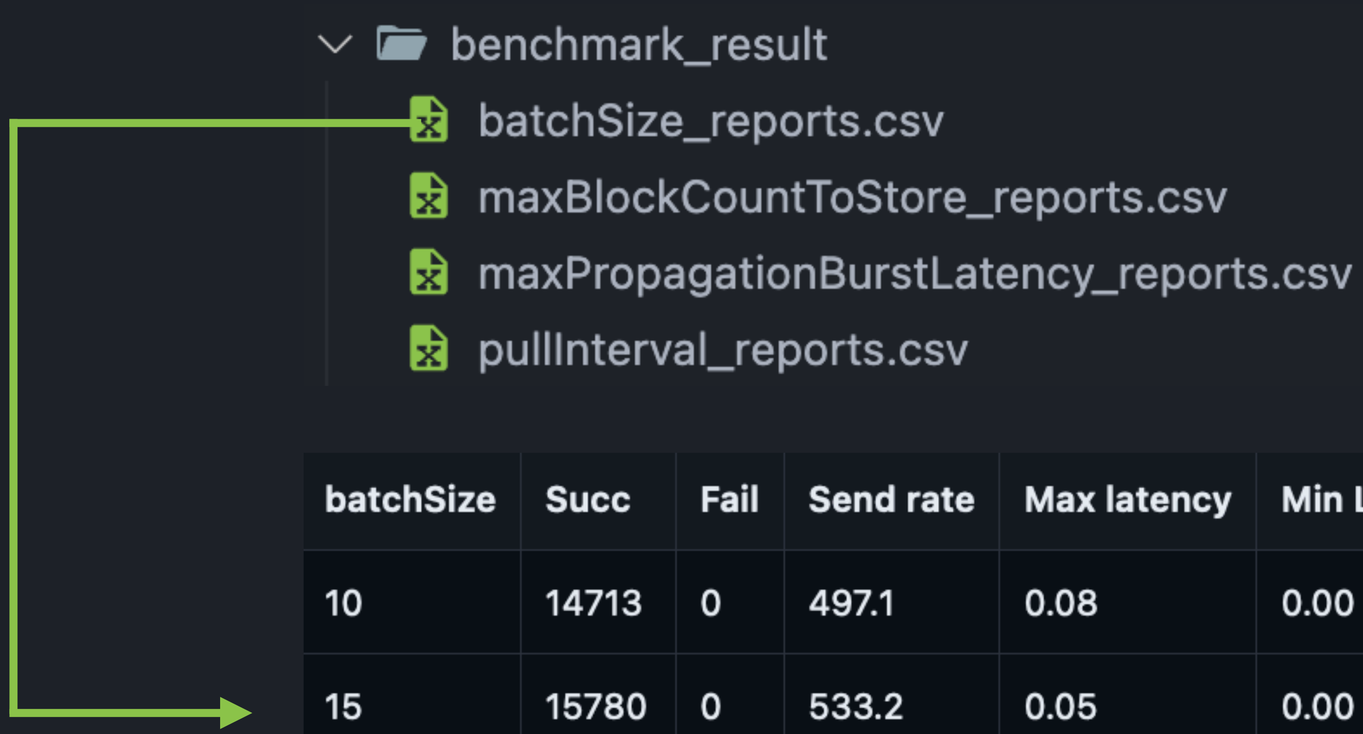
Testing the solution

Sample config

Parameter	batchSize	maxBlockCountToStore	maxPropagation-BurstLatency	pullInterval
Values	10, 15, 20, 25, 30, 35, 40, 45, 50	10, 15, 20, 25, 30, 35, 40, 45, 50	2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12	2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12

Testing the solution

Structure of collected data



batchSize	Succ	Fail	Send rate	Max latency	Min Latency	Avg Latency	Throughput
10	14713	0	497.1	0.08	0.00	0.00	497.1
15	15780	0	533.2	0.05	0.00	0.00	533.1
20	14893	0	503.3	0.03	0.00	0.00	503.2
25	15126	0	511.1	0.05	0.00	0.00	511.0

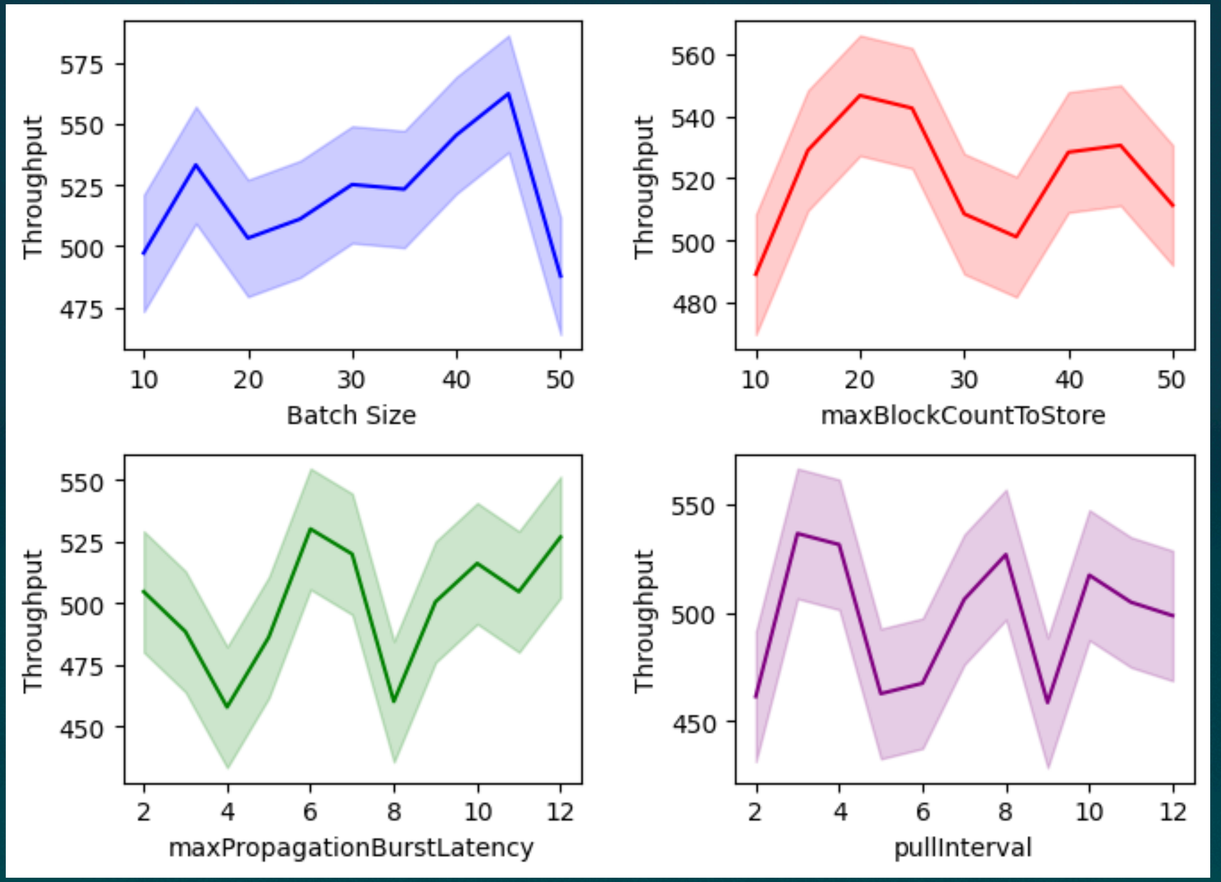
Testing the solution

Basic descriptive statistics

Parameter	Average	Median	Min	Max
batchSize	520.87	523.2	487.7	562.2
maxBlockCountToStore	520.8	528.4	489.0	546.7
maxPropagation-BurstLatency	499.62	504.7	457.8	530.2
maxPropagation-BurstLatency	497.39	504.8	458.5	536.5

Testing the solution

Dependence visualisation



Summary

- Built program allows for running a set of benchmarks on predefined input parameters and corresponding values
- The solution is proved to be working correctly

Opportunities for further research and development

01.

Adding options for choosing benchmark scenario

02.

Collecting more data and optimising the network

03.

Developing similar solutions for benchmarking other Blockchains