Appendix A Subgraph Gas Data Analysis

Under the DIPS mechanism, indexers will strategically set their price in the presence of competing indexers. Specifically, the lower the indexer's posted price, the more likely it is to get selected to index and receive payment. However, prices that are too low may be insufficient to cover the cost of actually doing the indexing. As a result, indexers will carefully balance profit motive with competitive pressures. There are many factors that impact an indexer's profitability such as compute resources, labor time, hardware redundancy, geographic characteristics and capacity, among others. This experiment is *not* intended to provide a final pricing strategy. Instead, its goal is to illuminate how subgraph gas (and other verifiable quantities) correlates with real world quantities that are themselves correlated with indexer costs such as clock time, disk usage, network usage, memory utilization and capacity. Indexers could then use the results of the experiment to inform hardware provisioning and pricing decisions as well as providing sync-speed warranties.

A.1 Purpose

The purpose of the gas analysis is twofold:

1. Determine if subgraph graph is an appropriate pricing unit, and if not, determine a more appropriate pricing unit that correlates with cost.
2. Provide guidance to indexers on how to price their indexing services.

However, since we will be using two cloud providers for these experiments, these experiments will also provide a price/quality comparison of two cloud providers, Equinix and Google Cloud (GCP).

A.2 Experiment Details

The experiment will consist of two separate but related components. The first part will focus on gathering data when subgraphs are synced on dedicated hardware. This allows us to determine how sync speed (and therefore cost) relates to subgraph gas *when all of the hardware resources are available.* The second component will focus on how sync-speed declines as multiple subgraphs are synced on the same hardware. This may reveal — for example — when indexers need to provision new hardware in order to meet sync-speed warranties and illuminate where there might be 'jumps' in marginal costs.

A.2.1 Experiment I — Dedicated Hardware

Let S be the set of subgraphs and V be the top x percentile of subgraphs in terms of query volume. All samples are uniform. The set of hardware we will use is given in table A.2.1. The experiment setup is as follows:

- Randomly sample n_1 of V. Call this set U_1
- For hardware configuration i , randomly sample m_1 of the top $V \backslash V_1$ subgraphs. Call each of these sets ν_i.

Provider	Hardware
Equinix	n3.xlarge.x86
Equinix	c3.medium.x86
Equinix	m3.large.x86
GCP	n2d-standard-64
GCP	c3-standard-44
GCP	n1-highmem-32

Table 1: Hardware to Run Experiments

- Uniformly randomly sample m_2 of the set of $S \backslash V$ subgraphs. Call this set U_2
- For hardware configuration i, randomly sample m_2 of the top $S \backslash (V \cup U_2)$ subgraphs. Call each of these sets μ_i.
- For hardware configuration i, index subgraphs $U_1 \cup \nu_i \cup U_2 \cup \mu_i$ and collect data.

Intuitively, U_1 and U_2 are high volume and low volume subgraphs that every hardware indexes. We stratify based on subgraph volume in case there are structural differences between subgraphs with high-volume and low-volume. Forcing each hardware configuration to index the same subgraphs ensures that there is an "apples to apples" comparison among hardware types.

The sets ν_i and μ_i allow for a larger exploration of the space of subgraphs, provide robustness checks and also add statistical power. If our statistical estimates for each hardware type using U_1 and U_2 are similar to those using ν_i and μ_i, we can be confident in the results. However, estimates that are practically and statistically significantly different necessitate further analysis and experimentation and suggest sampling error or insufficient sample size.

Finally, we will log data on a block-by-block basis to increase the size of the dataset, have finer granularity and allow for partial syncing.

Finally, the data we will collect for this experiment is given in table A.2.1. Our statistical model will then use the rows labeled "Gas Quantity" to predict sync-time for each of the different hardware configurations while controlling for number of blocks. We will also provide statistical models for utilization, ingress and disk space.

A.3 Experiment 2 — Shared Hardware

The purpose of Experiment 2 is to examine performance degradation on shared hardware. As a simple thought experiment, suppose that CPU utilization was the limiting factor. If indexing one subgraph only used 25% of the CPU, then indexing another subgraph —provided it used less than 75% CPU — would have a marginal cost of 0. Therefore, to help indexers fully characterize their cost models, this experiment will examine the ways in which *marginal* cost varies as subgraphs are added to shared hardware.

Let S' be the set of subgraphs whose maximum indexing time in Experiment 1 was in the interval $[T_L, T_H]$. Experiment 2 is as follows:

- Randomly select N sequences of subgraphs of length K from S'.

Field	**Description**	Data Category
subgraph-id	ID of subgraph indexed	Identifier
abort	Operation Count and Gas value (2 fields)	Gas Quantity
store.set	Operation Count and Gas value (2 fields)	Gas Quantity
store.remove	Operation Count and Gas value (2 fields)	Gas Quantity
store.get	Operation Count and Gas value (2 fields)	Gas Quantity
ethereum.call	Operation Count and Gas value (2 fields)	Gas Quantity
dataSource.create	Operation Count and Gas value (2 fields)	Gas Quantity
log.log	Operation Count and Gas value (2 fields)	Gas Quantity
bigInt.plus	Operation Count and Gas value (2 fields)	Gas Quantity
bigInt.minus	Operation Count and Gas value (2 fields)	Gas Quantity
bigInt.times	Operation Count and Gas value (2 fields)	Gas Quantity
bigInt.divided_by	Operation Count and Gas value (2 fields)	Gas Quantity
bigInt.mod	Operation Count and Gas value (2 fields)	Gas Quantity
bigInt.pow	Operation Count and Gas value (2 fields)	Gas Quantity
bigInt.bit_or	Operation Count and Gas value (2 fields)	Gas Quantity
bigInt.bit_and	Operation Count and Gas value (2 fields)	Gas Quantity
bigDecimal.plus	Operation Count and Gas value (2 fields)	Gas Quantity
bigDecimal.minus	Operation Count and Gas value (2 fields)	Gas Quantity
bigDecimal.times	Operation Count and Gas value (2 fields)	Gas Quantity
bigDecimal.divided_by	Operation Count and Gas value (2 fields)	Gas Quantity
bigDecimal.equals	Operation Count and Gas value (2 fields)	Gas Quantity
Block Start Time	Time block sync started	Outcome Metric
Block End Time	Time block sync Ended	Outcome Metric
Chain	The chain where the subgraph resides	Identifier
cpu	Computer Processor Type	Hardware Description
cores	Number of Cores of CPU	Hardware Description
boot	Boot Drive (Equinix)	Hardware Description
memory	Amount of Memory	Hardware Description
network	Network Package	Hardware Description
storage	Type of Storage Drive	Hardware Description
Cloud Provider	Either GCP or Equinix	Hardware Description
Block Number	Block Number	Identifier
CPU Utilization	Average CPU Utilization for Duration of Sync	Outcome Metric
Memory Utilization	Average Memory Utilization for Duration of Sync	Outcome Metric
Ingress	Network Ingress	Outcome Metric
Disk Usage	Size of Subgraph on Disk	Outcome Metric
Triggers	Count of Triggers Processed	Outcome Metric
Entity Ops	Number of Entity Ops	Outcome Metric
Bytes Written	Bytes written by entity ops	Outcome Metric
Bytes Read	Bytes written by entity ops	Outcome Metric
eth call	calls to ethereum network	verifiable quantity
firehose call	calls to firehose service (may be many)	outcome metric

Table 2: Data for Experiment 1

Field	Description
Sequence	The sequence of subgraph IDs
j	The depth of the sequence

- For each hardware and each sequence and for j in 1,2...K/m, simultaneously sync the first $j \times m$ subgraphs in the sequence and record the same data as Experiment 1.

Intuitively, Experiment 2 progressively adds larger loads to the hardware and collects metrics. The generated data will be the same as in Experiment 1 with two additional fields

We will again use the generated data and gas quantities to model how performance degrades with hardware load.

A.4 Commentary

Two meta-points regarding the experiment. First, there are many parameters in the experimental design. We will choose these by doing an exploratory data analysis so that our experiment uses approximately one month of compute time.

Second, this experiment is for subgraphs only and not for substream-backed subgraphs. It might (and likely) will eventually be valuable to repeat a similar experiment for substream-backed subgraphs, but that is out of scope for this initial experiment.