CSC 2012 – Virtualization Exercises

Solutions

Exercise 1 – CernVM out of the Box

1.4 The software is already in the CernVM-FS cache.

1.5 The CernVM-FS cache has increased the image size. The CMS event display requires around 760 files.

1.6 By recursively accessing all available software releases, the cache will be quickly bloated by tens of gigabytes of data and millions of files. At the same time, it causes in excessive network load.

Exercise 2 – CernVM Zoo

2.2 The machines have 1 GB RAM.

Bonus Through Amazon EC2 contextualization, we start a work_queue_worker process. This process is part of the *Work Queue* master-worker framework (`http://www.cse.nd.edu/~ccl/software/workqueue`).

The worker processes connect to a *catalog server* and wait for work given by a Work Queue master. In the following exercises, each Makeflow job will create a transient Work Queue master on the invoking machine.

Exercise 3 – Monte Carlo π

3.4 Let n be the total number of trials and h be the number of hits inside the circle. We have

$$\frac{h}{n} \approx \frac{\pi r^2}{4r^2}$$

and $\pi \approx 4h/n$.

3.5 It usually requires quite a lot of trials. The exact number depends of course on the odds. Something between 30 billion and 100 billion trials should work. As a rule of thumb, the error decreases proportionally to $\sqrt{n}$.

Bonus: The shootings into the square can be seen as independent Bernoulli experiments. The probability of a success is

$$p := \mathbf{P}(\text{shot into the circle}) = \frac{\pi}{4}.$$

The overall number of hits S is binomially distributed, that is $S \sim B(n,p)$. Here, the binomial distribution can be approximated by the normal distribution $\mathcal{N}(\mu, \sigma^2)$. We know that the mean μ of $B(n,p)$ is np and the variance σ^2 is $np(1-p)$. For the normal distribution, we further know that 95 % of the values are within $[\mu - 1.96\sigma, \mu + 1.96\sigma]$. Hence the error of the observed number of hits is (with a probability of 95 %) within $\pm 1.96\sqrt{np(1-p)}$ and the error for the observed value of π is within

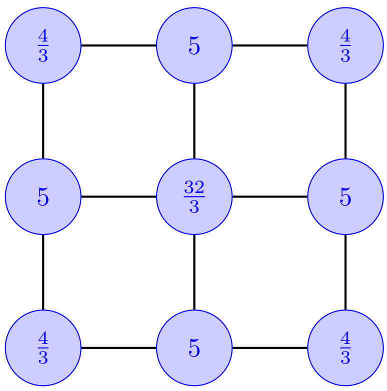
$$\pm 4 \cdot 1.96 \frac{\sqrt{\pi/4(1-\pi/4)}}{\sqrt{n}}.$$

For a desired error of less than 10^{-5}, we get $n > 10^{11}$.

Exercise 4 – Who's a Hub?

4.1 **Star** Only the center has non-zero betweenness. For the outer vertices, all $s-t$ paths are shortest paths and the center is on all of them. Hence for $n-1$ outer vertices, the betweenness of the center is $(n-1)(n-2)/2$. For the `star.graph` graph with 32 vertices, the center has betweenness 465.

Grid With increasing distance from the center, the betweenness decreases.



Complete Graph All shortest paths are direct connections between vertices. As no vertex is an inner vertex of a shortest path, the betweenness is 0 for all vertices.

4.2 Taking into account that the graph has around 3200 vertices, a sample Makeflow file for 4 virtual machines is

```
AIRPORTS: dsum DELTA0 DELTA1 DELTA2 DELTA3
        LOCAL cat DELTA0 DELTA1 DELTA2 DELTA3 | ./dsum > AIRPORTS

DELTA0: dependency routes.graph
        ./dependency -g routes.graph -s 0 -e 799 > DELTA0

DELTA1: dependency routes.graph
        ./dependency -g routes.graph -s 800 -e 1599 > DELTA1

DELTA2: dependency routes.graph
        ./dependency -g routes.graph -s 1600 -e 2399 > DELTA2

DELTA3: dependency routes.graph
        ./dependency -g routes.graph -s 2400 -e 3218 > DELTA3
```

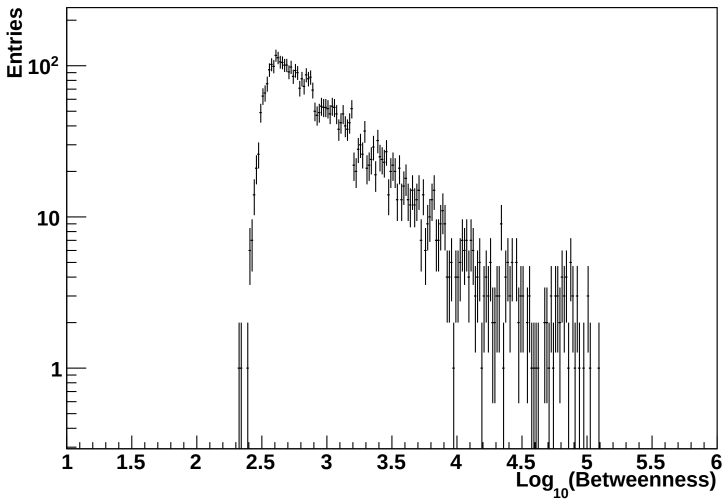
The top five airports and their approximate betweenness:

Airport	Approximate Betweenness
Ted Stevens Anchorage Intl	396 000
Chicago O'Hare Intl	312 000
Frankfurt Main	310 000
Charles De Gaulle Paris	304 000
Dubai Intl	269 000

4.3 The parallelized version has slightly sub-linear speedup. On a single virtual machine, the unparallelized version (`betweenness`) is faster than `dependency` and `dsum`. That is not necessarily the case. Sometimes an algorithm and its implementation have to be substantially improved in order to make it parallelizable. In such a case, the single-threaded version benefits as well.

4.4 For an unknown number of machines, the workload is not split into a few large pieces but into many small pieces of constant size k. Such Makeflow files have to be created by automation scripts. Many machines and many / large intermediate results, however, might introduce new bottlenecks when it comes to merging and moving the intermediate results around.

Bonus The distribution looks like this:



A few vertices have very large betweenness. In a log-log plot, the distribution comes close to a straight line. This is a typical characteristic of a “power law distribution”. When small world networks evolve, the likelihood for a vertex to become more connected increases proportionally to the level it is already connected. In other words, important vertices tend to become even more important.

For the reliability this implies that the network is vulnerable to targeted attacks on such important hubs. The network is very resilient against the loss of random vertices though.