CH/OTP Test Task

Jonn Mostovoy

`jonn.mostovoy@iohk.io`

August 2016

1 Task

Using 'distributed-process' library, implement a program by the following specification —

1.1 Specification

Several nodes continuously send messages to other nodes in such way, that every message reaches every node. *You are free to use any model of communication.* Each message contains a deterministic random number $n \in (0, 1]$. That happens for some time, afterwards, during the grace period, each node prints out the following tuple:

$$\left\langle |m|, \sum_{i=1}^{|m|} i \cdot m_i \right\rangle$$

where m is the list of all messages sent by all nodes, ordered by sending time, and m_i is the i-th message sent by some node. *You are most welcome to print out debug information to stderr.*

The larger your score with a given sequence of random numbers is, the better. Your code will be run under different network failure scenarios. For now we don't reveal "the failure maps", please use your best judgement of the most applicable communication model. You are free to tell us under which assumptions you chose it.
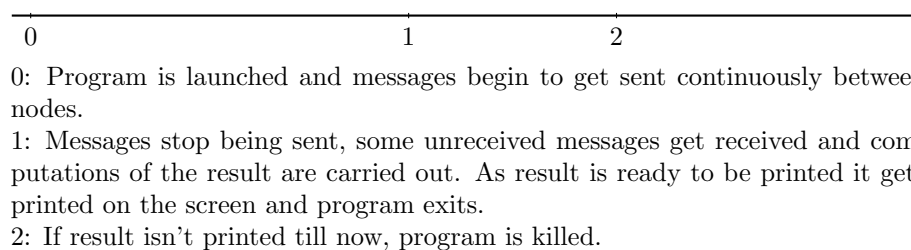
1.2 Clarification regarding ordering of messages in m

List of messages m has the following ordering —

$$i < j < k \wedge m_i = x \wedge m_j = y \wedge m_k = z \iff \tau(x) < \tau(y) < \tau(z)$$

where τ is a (hypothetical) function that takes a message as an argument and returns timestamp of when it got sent.

1.3 Clarification regarding submission execution periods

```
0               1        2
```

0: Program is launched and messages begin to get sent continuously between nodes.

1: Messages stop being sent, some unreceived messages get received and computations of the result are carried out. As result is ready to be printed it gets printed on the screen and program exits.

2: If result isn't printed till now, program is killed.

2 Submission requirements

2.1 Seeded RNGs

All the random choices made must be deterministic and seeded with a seed. It should be easy to re-run the program with a given seed.

2.2 Command-line arguments

Program should accept the two command line arguments: `--send-for k`, which denotes how many seconds does the system send messages, and `--wait-for l`, which denotes the length of the grace period in seconds. $s, k, l \in \mathbb{N}$. We also suggest providing `--with-seed s` argument, which defines seed for RNGs.

2.3 Cluster configuration

At the moment, we run submissions manually by patching node lists in submissions. That poses a requirement of submission having either source file or configuration file where we can put the list of nodes of our testing cluster. Please refer to this file in README.