COL106: Data Structures, I Semester 2018-19

Assignment 5

A publish-subscribe social platform

October 26, 2018

In this assignment we will build the basic data structures underlying publish-subscribe social platforms like Twitter, Instagram etc.

Overview

A publish-subscribe system typically has a set of users, U. Each user has two roles: (a) as a *publisher* the user is allowed to publish media and as a (b) *reader* the user is allowed to view media posted by certain other users.

To keep this assignment manageable we will make the following simplifying assumptions:

1. Time is discrete with no subdivisions, i.e., every action we describe below will take place at a time $t \in \{0, 1, \ldots\}$.

2. The only media publishers are allowed to post are short text of up to 30 characters.

To implement this assignment you will have to assign each user a *unique* user id. You will also have to assign each short text a unique text id. Each short text will also contain a field which will have a flag indicating whether it is a new text, a republished (reposted) text or a reply to an earlier text. Now let us see the details of the working of the system.

The publishing process. Each publisher (user) is allowed to publish *only one* short text of up to 30 characters at every time step. This is done through a PUBLISH action. Publication takes place in one of three ways

1. The user *writes a new short text*. In this case the text enters the data bases as a tuple: [[time; publishers id; NEW; Text string within quotes]],

   > e.g., [[86; u27; NEW; "Good morning!"]] is a new short text posted by user u27 at time 86. Once this is entered into the system (posted) it has to be assigned a text id. Let's say the id assigned to this text is "t1385".

2. The user *reposts an existing text*: Format: [[time; publishers id; REPOST(original text id)]]. Note that this is a text in itself and must be assigned its own unique text id.

   > Suppose user u14 reposts the text from the example above at time 122, it enters the text database as [[122; u14; REPOST(t1385)]]. Let's say this repost gets text id t2277.

3. The user *replies to an earlier text*: Format: [[time; publishers id; REPLY(original text id); Text string within quotes]]. This must also be assigned a unique text id.

   > Suppose user u31 replies to u14's report of u27's tweet at time 136, it is entered as [[136; u31; REPLY(t2277); "What's so good about the morning?"]].

The reading process Each user subscribes to a set of publishers (who are also users). At any time t the user may perform a SUBSCRIBE action to subscribe to a publisher or an UNSUBSCRIBE action to unsubscribe.

To read text posted on the platform the user may perform a READ action. Let us say the current time at which the user, v, performs a READ action is t_0 and the last time the user performed a read was at time $t_{-1} < t_0$ then a list of texts are returned. A text x is included in the list

1. If the user who posted x is u_x and v subscribed to u_x at time $t_s < t_0$ and did not unsubscribe from u_x between the t_s and t_0 and

2. if the timestamp t_x of x is such that $\max\{t_{-1}, t_s\} \leq t_x < t_0$.

3. Or if $t_{-1} \leq t_x < t_0$ and x is a reply to some text posted by v.

In other words the text should either be a reply to one of v's text from *any* user or it should be an unseen text from a publisher that v is *currently* subscribed to.

The texts should be output sorted by time. Before a user has performed any READ we can assume that t_{-1} has some default value like -1 which is smaller than any allowed timestamp.

The programming assignment

In this programming assignment you will be asked to implement the publish-subscribe system. The difference between the previous assignments and this one is this: you will not be given *any* instructions on how to implement the system. You are allowed to use whatever data structures you like, use inbuilt implementations from Java where required, and generally build the implementation independently. The conditions are

1. You have to implement the features asked for.
2. You have to take the actions input and give the output in the required format.
3. You have to be able to explain the workings of your system clearly at viva time, and mention the source of every piece of code, every function.

Assignment 5 [100 marks]
Deadline: 11:55PM, 11 November 2018

Actions:

- `PUBLISH,t,uid,NEW,text,tid:` User `uid` publish new `text` at time `t` having text id `tid`.
- `PUBLISH,t,uid,REPOST(ptid),tid:` User `uid` repost existing text represented by text id `ptid` at time `t`. The repost text is assigned a new text id `tid`.
- `PUBLISH,t,uid,REPLY(ptid),text,tid:` User `uid` reply to an existing text represented by text id `ptid` at time `t`. The reply `text` is assigned a new text id `tid`.

- `SUBSCRIBE,t,uid,pid:` User `uid` subscribe to a publisher `pid` at time `t`.
- `UNSUBSCRIBE,t,uid,pid:` User `uid` unsubscribe to a publisher `pid` at time `t`.
- `READ,t,uid:` User `uid` reads text at time `t`.

Points to note:

- Your implementation should catch exceptions for any action and throw error messages. For eg. Suppose a user wants to unsubscribe from a publisher who the user has not subscribed.
- A user can reply to any type of post (NEW/REPOST/REPLY).