UNIVERSITY OF TRENTO

Evaluating the Convergence Speed of Anti-Entropy Protocols

Name Surname, Pinco Pallino and Leslie Lamport

November 2, 2021

This a suggested report template. You are free to edit it and change its organization in the way you consider more appropriate to present your work.

ABSTRACT. Application-level broadcast/multicast is an important building block to create modern distributed applications. Epidemic protocols are proposed in the literature to support broadcast in distributed systems. The main differences among such protocols can be evaluated in terms of efficiency, robustness and speed when scaling. In this assignment we have focused on Anti-entropy protocols and implemented three fundamental well-known schemes, i.e., *Push*, *Pull* and *Push-pull*. Our implementations is based on *python-MESA* and let us conduct a comparative study in simulation which confirms that, as the theory suggests, push-pull protocols grant faster convergence speed.

1 INTRODUCTION

Provide a gentle introduction to the implemented/studied protocols. Basic example follows:

- Problem statement: e.g., "fast and reliable diffusion of a content in a distributed system."
- General/brief discussion of known approaches in the literature: e.g., short discussion about PROs and CONs of flooding, tree-based diffusion and gossip.
- Narrow down to your chosen protocols: introduce a bit more push/pull protocols and briefly discuss advantages and disadvantages.
- Declare goal and content of the assignment: e.g. *We wanted to study the properties of push/pull protocols and verify that, compared to flooding approaches, they enable a considerable reduction of the number of messages necessary to complete the diffusion process of a file in distributed systems. Moreover, we verified that they grant a higher degree of tolerance to failure if compared with tree-based approaches. Finally, we have studied in simulation the convergence properties of 3 different anti-entropy based protocols, namely, Push, Pull and Push-pull, with our simulation results confirming the expectation that push-pull protocol ensures shorter convergence times in all our simulated scenarios of file diffusion processes within different distributed systems with varying number of nodes N.*

2 Theory Background

Minimal more detailed background reporting the essential notions to understand the rest of the report. For example, this section can be a good place for explaining why with flooding the efficiency in terms of number of messages is known to be $O(n^2)$; why with tree-based protocols this efficiency improves up to $O(n)$ but why we have reliability issues.
Describe anti-entropy main principle, with pseudocode;
Describe distributed system model: crash and network failures models/assumptions.

3 Simulator/Implementation Architecture

Describe how you have developed an implementation of your target protocols. Describe if this implementation is tailored for being tested under simulation (Discrete Event / Agent-based modeling etc.) or if it is a minimal real-world implementation that has been tested/studied creating an appropriate test-bench / emulation framework.
Essentially, describe your code and your design choices. Provide your instructor the necessary information to understand your codebase and evaluate your design choices. Not only, remember that... system model and assumptions matter! If your simulation framework is responsible for the simulation of channel losses etc., then this section is the place where you should document the modeling assumptions relevant for the interpretation of your results (these latter should be reported and commented in Section 5).

Figure 3.1: A piece of code shown here and described in Section 3

```
class PushAgent(Agent):
    def __init__(self, unique_id):
        super().__init__(unique_id, model)


    def push(value):
        '''
        to be implemented
        '''
        pass


    def step(self):
        '''
        to be implemented
        '''
```

4 Experiment Setup

If your effort consists in implementing a particularly complicated protocol / distributed systems for the sake of proving the ability of implementing such one and for the academic purpose of showing that you master the theoretical and practical skills necessary for completing this implementation... then SKIP THIS SECTION and rather write a DEMO section, where you describe how to run your code to visualize and thus appreciate all the implemented mechanisms. Include screenshots if appropriate.
Otherwise, this section should be the classic section where, once that Background and Architecture (Sections 2 and 3) are already clear, you document the fixed and varying parameters

describing the experiment you did to measure the performance of your protocol/system. Document/Define also the performance metrics measured during experiments. Using a MESA jargon, metrics could be defined by the *model-level* or *agent-level reporters.* Example of a metric definition:

Definition (Convergence Speed)**.** *The pure number indicating, for a MESA experiment, the step index after which all processes have received the diffused file.*

5 Results

Report here your experimental results, make use of figures and tables if appropriate.

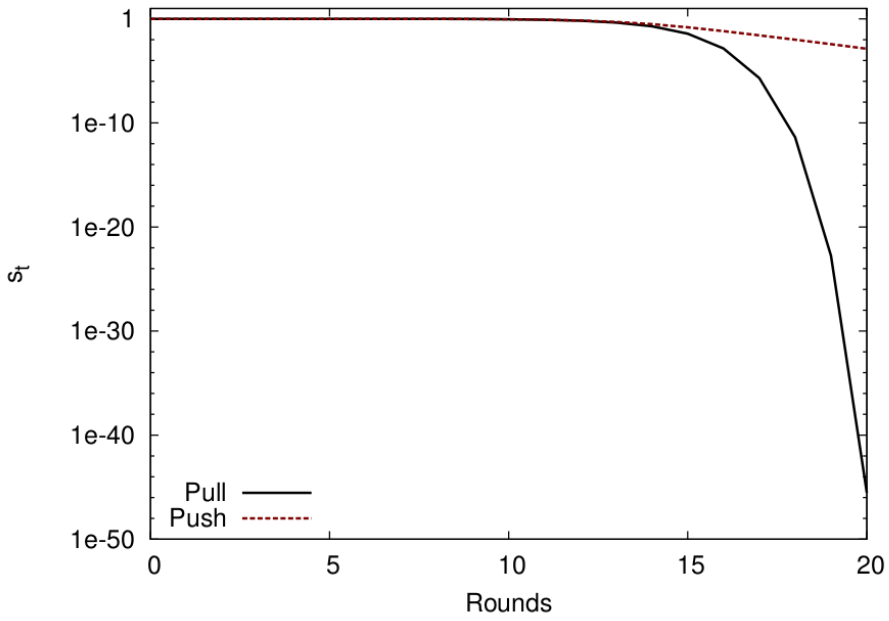

Figure 5.1: Termination Time for network size $N = 10000$

Network Size	**Push**	**Pull**	**Push-Pull**
10	X	Y	Z
50	...	...	...
100	...	...	...
500	...	...	...
1000	...	...	...
10000	...	...	...

Table 5.1: Termination Time for tested protocols and for growing number of nodes in the network.

6 Conclusion

Tell me how clever you have been! :)