Assignment 2 – Parallel Computing (CS F422)

Rajath S 2012B5A7589P

Ajinkya Kokandakar 2012B3A7463P

Varad Gautam 2012A8TS237P

System Design

Program Components

The program consists of three components:
Informant
Reporters
Editors

Informant – NEWS SOURCE

The news source is modelled as an informant which generates random events in different areas and sends the news to a random reporter belonging to the area.

Reporter

Each reporter plays one of the two roles: Lead Reporter or Normal Reporter

Meeting/Round:

In each round, all the reporters (including Lead Reporter) receive news from the informant and store them in a sorted array. After receiving the news, the normal reporters send a ping to the Lead indicating that a news item has been received.
The lead reporter in addition to receiving news, receives the pings and keeps a count of the total news items received in the round. If the collective newscount crosses a certain threshold or if no news is received for a certain amount of time (MAX_REPORTER_SESSION_DURATION) then the lead reporter broadcasts a message to all colleagues to indicate a meeting has been called.
Meeting refers to all reporters calling MPI gather function. (This is the only blocking function in the entire reporter code)
After the meeting, the Lead reporter sorts the gathered messages by event id and sends the cache of news to their corresponding editor. The Lead reporter is then changed in round robin fashion for **load balancing.**

Editor

Each editor also plays one of the two roles in each round: Lead or Normal

Meeting/Round:

The design of editors is almost same as that of reporters. Whereas reporters receive single news “tips” from the informant, editors receive a cache of news from one of their reporters (thus informant is replaced by reporter and newsitem/tip is replaced by a **cache of news**). After receiving a cache of news, the normal editors send a ping to the Lead editor.
(NOTE: Cache is a bunch of news sent/received together)
The Lead editor in addition to receiving caches, receives pings from other editors and keeps a count of the number of caches received collectively. Similar to the lead reporter if the number of caches exceeds a certain threshold or if no cache is received for a certain amount of time (MAX_EDITOR_SESSION_DURATION) then the lead editor broadcasts a message to all colleagues to indicate a meeting has been called.
Meeting, as before, refers to all editors calling MPI Gather (**Both gathers, in reporter and editor are the only blocking calls in the entire program**). The Lead editor then merges (i.e. sorts) the news from all caches and prints it on the terminal.
After a meeting the Lead editor role is cycled in round robin fashion.

Some Implementation details:

1. All communication operations except for the two gathers are **non-blocking.**
2. Specific communicators are used for:
   - Reporter group communications (Reports under same editor working in same area)
   - Reporter-Editor communications (Self-explanatory)
   - Informant-Tipper communications (All reporters and one informant)
3. This ensures that (MPI_COMM_WORLD is an exception in all cases):
   - Reporters working under different editors and/or different areas cannot communicate with each other.
   - Editors cannot communicate with other editors’ reporters
   - Tipper can communicate with all communicators
4. Communicators have been designed in such a way that partition of reporters can be controlled across two parameters – (i) area (ii) editor. The partition must be explicitly provided as an input. This ensures a high level of flexibility as well as scalability
   Specific communicators for different partitions minimizes communication costs. (The alternative is doing repeated send/receive which incur significant overhead)
5. Online sorting (Insertion sort) is employed to sort the news queue.