OS Lab 1

Goutham Devarayasamudram and Siddharth Udayappan Chidambaram

March 7, 2013

Introduction

Our world is a one-dimensional 'Grid'. Each position on this grid can be occupied by a Pig or a stone column or it can be empty. The leftmost-position on the grid is taken as the origin and all distances are measured relative to this position. So when we say a Pig or a Stone Column is at a distance of 6, it means it is 6 positions to the right of the origin. The bird starts from outside the grid and to the left of it. So whichever Pig is in the leftmost position on the grid will first observe the Bird fly. We assume that the Pig knows the exact landing position and the time taken by the Bird. Once the Pig sees the Bird go, it will notify all its peers about the final bird position and the time it takes to reach them.

Setup

Each Pig is represented as an object and is assigned a random position on the grid when it is created. Stone columns are also given random positions on the grid, with the extra stipulation that no 2 stone columns can be next to each other. When the game starts, we create a thread and call the mainloop of each Pig instance. This mainloop sets up a socket for listening on a certain port. Any peer that wants to send information to this Pig will have to connect to this port. Whenever this socket recieves any connection, it immediately opens a new thread to handle that information and gets back to listening for more connections. Pigs also send their locations over the p2p network using a 'sendAddress' packet. This is done so they can note who their physical neighbors (i.e. those Pigs within a distance of 2 on each side) are. Each Pig is notified when the game starts so that it can update its start time. Note that there is no global clock and the start time for each Pig is the current time of the system where the Pig instance is residing.

As soon as the bird starts flying, The Pig with id 0 creates a bird approaching packet and sends it to all its peers and those peers will in turn send the packet to their peers and so on. This packet will contain the message id (1 in this case), the bird's landing position and the time taken to reach that position. It also contains a hopcount, which is initially equal to the number of Pigs in the system and is decremented each time the packet is recieved by a Pig. If the hopcount becomes 0, the bird approaching packet will be discarded. Finally, the Bird Approaching packet contains the path it has traversed so far. Note that the Pig with id 0, which initiates the bird approaching process, can be anywhere on the grid.

Notify Bird Approaching

On receiving the Bird approaching packet, many things happen. Each Pig will send this packet to its peers if the hopcount is non-zero and then send back an acknowledgement along the path (got from the message) to the Pig that initiated the packet. It will then proceed to check if the bird landing position in the message is within its vicinity. It does the below things in order :

- It will first check if there is a stone column to its immediate left and if the bird is colliding with the stone column. If that is the case, then if the current time on the Pig's system is less than its start time plus the bird landing time (got from the message), it will try to move to its right (if thats possible) and then notify its neighbors to also change their position (using a 'takeshelter' message). Otherwise, if it has run out of time, it will note the fact that it has been hit (this will be used later when we need the score) and then if there is a Pig to its right, it will send a packet (called 'killneighbor' in our program) telling the other Pig to also note the fact that it (the other pig) has been hit (This is because the stone column hits the first Pig and then this Pig will collide with the Pig next to it).

- If that is not the case (that either there is no stone column next to it or the bird is not landing to its left), then it will proceed to check if the bird is landing on its current position. It will check if the bird has not already landed (by looking at the time as before). If it has time, it will try to update its position and notify its neighbors that they might be affected. If it doesn't have time, then it will note the fact that its been hit and ask its neighbors to also note that they've been hit if their position hasn't changed.

The main Game thread will sleep until the acknowledgement to all the bird approaching packets have been recieved.

Status Query and Score calculation

Once all the bird approaching packets have been recieved by the Pig that started the process, the Pig will then proceed to query for the status of all the other Pigs. Again, this querying is done through the P2P network. The Status packet consists of the message id (2), the id of the Pig whose status is asked for, the hopcount and the path taken by the packet so far. Whenever a Pig receives this status packet, it will immediately respond with whether it was affected or not by the bird by sending a was-hit packet along the path. These was-hit packets are used to keep the score. The packet consists of message id (3), the Pig that originally sent out the packet, the current pig id, a boolean indicating whether it was affected or not and the pig ids along the path. The Pig that initiated the status request waits until it has received the was-hit packets from all the Pigs and once it has done so, it proceeds to calculate the score by looking at how many was-hit packets had their result value as True (True here indicates that a Pig was affected by the Bird, either directly or from one of its neighbors or the stone columns).

Messages

The Pigs use various messages to communicate with each other. Given below are the message Ids and their descriptions.

Table 1: Message IDs

Message ID	Name	Description
1	Bird Approaching	Sent to all the Pigs (through the P2P network) to tell them when and where the Bird will land
2	Status Request	Sent by the Pig that observed the bird to some specific Pigs querying its status after an attack.
3	Was Hit	Sent to whichever Pig initiated the status query saying whether the Pig was affected by the Bird or not
4	Take Shelter	Sent to a Pig's neighbors telling them they might get impacted by the Bird (directly or indirectly)
5	Kill Neighbor	Sent by a Pig to its neighbors, asking them to kill themselves if their position hasn't changed.
6	Send Address	Used by the Pigs to notify each other who their physical neighbors are.
7	Status All	Same as Status Request, except this is sent to all the Pigs (Again through the network).
8	Bird Reply	This packet is sent as an acknowledgement after receiving the Bird approaching packet.

Improvements

We can think of many possible improvements to our design. Our grid is currently 1-dimensional. We would like to change this to 2-dimensions and see what new insights come out of it. Another thing we want to see is to make the bird's landing position and speed (i.e time taken) random. That is, the Pigs will know that the bird will fall somewhere in the vicinity of the original position that was sent to all Pigs but they won't know the exact position. For eg: The Pig that first observes the Bird can fit a curve to the trajectory of the Bird to guess where it might land.

Assumptions

We've collected below all the assumptions we have made in our program:

- The Pig that first observes the Bird knows its exact landing position and the time it takes to reach there
- No 2 Stone columns are next to each other
- Whenever a Pig is hit by a Bird, the pig can only affect its physical neighbor to its right. So if there is a Pig to the left of the Pig that got hit, then that Pig will be unhurt. Same goes for Stone Columns. If a Bird hits a stone column, the column can only fall to the right.
- We define a Pig's physical neighbors to be those Pigs within a distance of 2 from the current Pig.
- The initial hopcount is equal to the number of Pigs in the system.