Asynchronous Minimum Hops

Anna Zaks

October 23, 2007

Distributed Computing, NYU
**Min Hops Problem**

- Obtain the paths with the smallest number of links from a given node \( s \) to each other node of the network.

- Devise an asynchronous algorithm.

- The algorithm itself must determine when it terminates - we cannot just promise that it will eventually terminate.

![Network and Solution Graphs]
Initialization

- Let $s$ be the root node
- Let $\text{dist}(x)$ denote the length of the shortest path from $x$ to $s$
- $\text{dist}(s) = 0$
- For all nodes $x \neq s$: $\text{dist}(x) = \infty$
Action of the Root Node $s$

upon receipt of a message "compute min hops":

for all $i$ in neighbors($s$) {
    send a message to node $i$; //pass parameter dist($s$)
}

for all $i$ in neighbors($s$) {
    await for the reply from node $i$;
}

broadcast termination;
Action of Every Network Node $x$

upon receipt of a message from node $y$:

$$\text{if ( dist}(x) > \text{dist}(y)+1 ) \{$$

// reset the distance of $x$ and it’s successors

$$\text{dist}(x) = \text{dist}(y)+1;$$

set the first link on the shortest path to $y$;

$$\text{for all } i \text{ in neighbors}(x) \{$$

send a message to $i$; //pass parameter dist($x$)

$$\}$$

$$\text{for all } i \text{ in neighbors}(x) \{$$

await for the reply from node $i$;

$$\}$$

$$\}$$

send the reply to node $y$;
Correctness Argument (together with Dennis Shasha) 1/3

- Call a node "unhappy" with processing a message if it did not get a reply to at least one message that it had sent out. A node sends a reply if and only if it is "happy".

- If \( x \) is unhappy, there is a path of unhappy nodes from the root \( s \) to \( x \). Just follow the path along which the current message has been sent: \( s = y_0, y_1, \ldots, y_n, x \). All the nodes on that path are unhappy. Suppose \( y_i \) is unhappy, then so is \( y_{i-1} \) since it waits for a reply from \( y_i \).

- The root node \( s = y_0 \) is also unhappy so there won't be a premature termination.
Lemma: If a node $i$ is happy in reply to a message with distance $D$, then $\text{dist}(i) \leq D + 1$. Only the following two cases are possible:

- Its distance already satisfies $\text{dist}(i) \leq D + 1$ as a result of some previous event. Either because it had reset its distance to the lower value in response to some previous message; or, in case $i = s$, it had been initialized with the lower distance value.
- It has just reset its distance to $\text{dist}(i) = D + 1$. 


Correctness Argument (together with Dennis Shasha) 3/3

- Suppose a node $x$ is happy in response to a message and $\text{dist}(x) = D$. Then, for all nodes $j$ such that the min hops path from $x$ to $j$ is of distance $N$: $\text{dist}(j) \leq D + N$ and $j$ also is happy. Let's proof by induction on $N$:
  - Base Case: If $N = 0$, then $j = x$ and the claim trivially holds.
  - Inductive Hypothesis: Assume the claim holds for $N = K$: for all nodes $i$ s.t. min hops distance from $x$ to $i$ is $K$: $\text{dist}(i) \leq D + K$ and $i$ is happy.
  - Inductive Step: Let's show that the claim holds for each node $j$ whose min hops distance from $x$ is $K + 1$. The min hops path from $x$ to $j$ must go through node $i$, which is a neighbor of $j$ and is exactly $K$ hops away from $x$. By the inductive assumption, node $i$ is happy. Thus, $i$ has received a reply from $j$ (in response to a message with distance $\text{dist}(i)$). So $j$ must also be happy and, by the lemma, $\text{dist}(j) \leq \text{dist}(i) + 1$. Applying the inductive hypothesis, we obtain $\text{dist}(j) \leq D + K + 1$.

- If the root node $s$ is happy, then all nodes have the minimum hop value.

- The algorithm terminates because every change message reduces the number of hops to some node and that number is bounded by 1 from below.