Gathering for mobile agents with a strong team in weakly Byzantine environments

Jion Hiorse\textsuperscript{1}  Masashi Tsuchida\textsuperscript{1}  Junya Nakamura\textsuperscript{2}
Fukuhito Ooshita\textsuperscript{1}  Michiko Inoue\textsuperscript{1}

\textsuperscript{1}Nara Institute of Science and Technology
\textsuperscript{2}Toyohashi University of Technology
Agenda

- Background
- Contribution
- Model & Goal
- Proposed Algorithm
  - Basic Idea
  - Details
- Conclusion
Mobile agents

- Software programs moving autonomously from node to node in a distributed system
  - An agent can keep its state and program during move
- A paradigm to design distributed systems
Gathering

- **Goal:** All agents meet at a single node
- **Efficient information sharing scheme among all agents**
- **Many researches on various models:**
  - Presence/Absence of whiteboards (memories on nodes)
  - Anonymous/Distinct agents (no/unique IDs)
  - Synchronous/Asynchronous agents, etc.
Byzantine environments

- **Byzantine agents** exist in the systems
  - They behave arbitrarily
  - They imitate software bugs, cracked agents, etc.

**Example**

**Arbitrary behavior**

**False information**

> All agents gathered on this node

**Algorithms for Byzantine environments tolerate any fault of agents**
Gathering in synchronous Byzantine environments

Time complexity of the existing algorithms isn’t small

<table>
<thead>
<tr>
<th>Input</th>
<th>Byzantine fault</th>
<th>Condition of #Byzantine agents</th>
<th>Time complexity</th>
</tr>
</thead>
<tbody>
<tr>
<td>$n$</td>
<td>Weak</td>
<td>$f + 1 \leq k$ (Optimal)</td>
<td>$O\left(n^4 \cdot</td>
</tr>
<tr>
<td>$f$</td>
<td>Weak</td>
<td>$2f + 2 \leq k$ (Optimal)</td>
<td>Poly. of $n$ &amp; $</td>
</tr>
<tr>
<td>$n, f$</td>
<td>Strong</td>
<td>$2f + 1 \leq k$ (Optimal)</td>
<td>Exp. of $n$ &amp; $</td>
</tr>
<tr>
<td>$f$</td>
<td>Strong</td>
<td>$2f + 2 \leq k$ (Optimal)</td>
<td>Exp. of $n$ &amp; $</td>
</tr>
<tr>
<td>$\log \log n$</td>
<td>Strong</td>
<td>$5f^2 + 7f + 2 \leq k$</td>
<td>Poly. of $n$ &amp; $</td>
</tr>
</tbody>
</table>

$n$: #nodes, $k$: #agents in the network, $f$: #Byzantine agents, $|\Lambda_{good}|$: The length of the largest ID among good agents, $X(n)$: Time required to visit all $n$ nodes

Reduce time complexity in weakly Byzantine environments by relaxing the condition of #Byzantine agents

#Byzantine agents is small in real networks

<table>
<thead>
<tr>
<th>Input</th>
<th>Startup delay</th>
<th>Condition of #Byzantine agents</th>
<th>Simultaneous termination</th>
<th>Time complexity</th>
</tr>
</thead>
<tbody>
<tr>
<td>[1]</td>
<td>$n$</td>
<td>Presence</td>
<td>$f + 1 \leq k$</td>
<td>$O \left( n^4 \cdot \Lambda_{good} \cdot X(n) \right)$</td>
</tr>
<tr>
<td>Algo. 1</td>
<td>$N$</td>
<td>Absence</td>
<td>$4f^2 + 9f + 4 \leq k$</td>
<td>$O \left( (f + \Lambda_{good}) \cdot X(N) \right)$</td>
</tr>
<tr>
<td>Algo. 2</td>
<td>$N$</td>
<td>Absence</td>
<td>$4f^2 + 9f + 4 \leq k$</td>
<td>Possible</td>
</tr>
</tbody>
</table>

$n$: #nodes, $N$: upper bound of #nodes, $k$: #agents in the network, $f$: #Byzantine agents, $\Lambda_{good}$: The length of the largest ID among good agents, $\Lambda_{all}$: The length of the largest ID among agents, $|\Lambda_{good}| \leq |\Lambda_{all}|$, $X(n)$: Time required to visit all $n$ nodes

$f < n \& |\Lambda_{all}| = O(|\Lambda_{good}|)$

$\Rightarrow$ faster than [1]
Agenda

- Background
- Contribution
- Model & Goal
- Proposed Algorithm
  - Basic Idea
  - Details
- Conclusion
A distributed system
- A node has neither ID nor whiteboard
- Edges incident to a node are locally ordered with a fixed port numbering

Agents
- Awake at the same time (no start-up delay)
- Have unique IDs
- Know the upper bound $N$ of #nodes
- Have unlimited amount of memory
- Can share information with other agents on the same node
Synchronous agents
- An agent can move to its adjacent node in 1 unit time (round)

A weakly Byzantine environment
- $f$ weakly Byzantine agents
  - Behave arbitrarily without following an algorithm
  - Can’t change their IDs
- At least $4f^2 + 8f + 4$ good agents
Goal: Gathering in a weakly Byzantine environment
  - All good agents gather on a single node
    - Don’t care Byzantine agents’ locations

Two types of termination
  - Non-simultaneous termination: All good agents terminate
  - Simultaneous termination: All good agents terminate at the same round
Agenda

- Background
- Contribution
- Model & Goal
- Proposed Algorithm
  - Basic Idea
  - Details
- Conclusion
Exploration: Visiting all the nodes in a network
- We use the existing algorithm [4]
- The time complexity is denoted by $X(N)$
  - $N$: Upper bound of #nodes

Tree and Ring: $X(N) = O(N)$
Arbitrary graph: $X(N) = O(N^5 \log N)$
Gather on the node with the agent with the smallest ID

1. Agents collect all agent IDs
2. Agents search for the agent with the smallest ID
   - The agent with the smallest ID waits for their arrival

Basic strategy: Smallest agent Collect agent IDs Wait Search Search Search Search Gather

: Smallest agent : Agent
Gathering: Difficulty & Strategy

**Difficulty**
- If a Byzantine agent has the smallest ID, agents fail to gather.
  - Agents don’t meet the Byzantine agent when collecting IDs
  - The smallest ID that agents know may be different

**Strategy**
- Create a reliable group consisting of enough agents & use only the information from reliable groups.
  1. Collect all the reliable group IDs
  2. Gather on the node with the group of the smallest ID

\[
\text{At least } 2f + 1 \text{ agents convey the same Info.} \\
\Rightarrow \text{This group has at least one good agent} \\
\Rightarrow \text{This information can be trusted}
\]
Overview

1. CollectID stage: Collect IDs including all good agents
2. MakeGroup stage: Create a reliable group
3. Gather stage: Achieve gathering
CollectID stage

- **Goal**: Collect IDs including all good agents
- **Behavior**: Execute the rendezvous algorithm [5]
  - Repeat **Exploring the network** or **Waiting for \(X(N)\) rounds** based on ID
    \(\Rightarrow\) A good agent can meet all the other good agents
- **Time complexity**: \(O \left( |\Lambda_{good}| \cdot X(N) \right)\)
  \(N\): upper bound of #nodes
  \(\Lambda_{good}\): The largest ID of good agents

Goal: Create a reliable group consisting of enough agents

Idea: Move to the node with the smallest ID agent

- **The smallest ID agent** waits
- **Other agents** search for the agent by the exploration algorithm

If the smallest ID agent is Byzantine, agents may not create a group.
MakeGroup stage: Improved idea 1/2

Improved idea: Make the smallest $f + 1$ agents wait

Agents with the smallest $f + 1$ IDs
⇒ At least one good agent waits

Agents searching for waiting good agents succeed to search for them

Example ($f = 1$)

Good agents don't know the exact $f$
⇒ Use an estimate value

Within $f + 1$
MakeGroup stage: Improved idea 2/2

Improved idea: Make the smallest $f + 1$ agents wait

- **Agents with the smallest $f + 1$ IDs** wait
- **Other agents** search for the smallest ID agent among collected IDs

- If they fail to find the smallest one, they search for the next one

Good agents are divided into at most $f + 1$ groups

Example ($f = 1$)

1. Agents with the smallest IDs wait.
2. Other agents search for the smallest ID agent.
3. If they fail, they search for the next one.
4. Good agents are divided into at most $f + 1$ groups.
At least one group must have at least $4f + 4$ agents (Reliable group)

- Good agents are divided into at most $f + 1$ groups
- Assumption of $f : 4f^2 + 9f + 4 = (4f + 4)(f + 1) \leq \text{#agents}$

When a reliable group is created, the group

- Decides the smallest ID in the group as the group ID
- Divides the group in half for the Gather stage

Information provided by a group of at least $2f + 2$ agents is reliable.
MakeGroup stage: Time complexity

- Time complexity: $O(f \cdot X(N))$
- #search iterations: at most $f + 1$

Worst case

An agent fails to search for Byzantine agents $f$ times
Goal: Achieve gathering

Idea:
1. Collect all reliable group IDs
2. Gather on the node with the reliable group with the smallest group ID
Gather stage: Step 1

Step 1: Collect all reliable group IDs

- Agents in a reliable group: Convey and collect group IDs
  - (Waiting gr.): Wait for $X(N)$ rounds
  - (Exploring gr.): Explore the network

- Other agents: Collect group IDs, while exploring the network
Gather stage: Step 2

Step 2: Achieve gathering

- Waiting group with the smallest group ID:
  - : Waits for $X(N)$ rounds

- Other agents:
  - : Search for the waiting group with the smallest group ID

The smallest group ID is 2
Gather stage: How to enter

Requirement

- All agents enter the Gather stage at the same round after they create a reliable group (RG)

Solution

- Enter the Gather stage every \( X(N) \) rounds of the MakeGroup stage
  
  ⇒ All agents repeatedly enter the Gather stage at the same round
  
  - All agents achieve the gathering if a RG is created
  
  - Agents don’t disturb the MakeGroup stage if no RG is created
CollectID stage: $O\left(\left|\Lambda_{good}\right| \cdot X(N)\right)$
- $\left|\Lambda_{good}\right|$: The length of the largest ID among good agent IDs

MakeGroup stage: $O(f \cdot X(N))$

Gather stage: $2X(N)$ rounds are inserted every $X(N)$ rounds
- Step 1: $X(N)$
- Step 2: $X(N)$

Total time complexity: $O\left(\left(f + \left|\Lambda_{good}\right|\right) \cdot X(N)\right)$

$f$: #Byzantine agents
$\left|\Lambda_{good}\right|$: The length of the largest ID among good agents
$X(n)$: Time required to visit all $n$ nodes
## Conclusion

### Contribution

Reduce time complexity in weakly Byzantine environments by relaxing #Byzantine agents.

### Future work

Design an algorithm that works even in the presence of startup delay.

### Table

<table>
<thead>
<tr>
<th>Input</th>
<th>Startup delay</th>
<th>Condition of #Byzantine agents</th>
<th>Simultaneous termination</th>
<th>Time complexity</th>
</tr>
</thead>
<tbody>
<tr>
<td>[1]</td>
<td>$n$</td>
<td>Presence</td>
<td>$f + 1 \leq k$</td>
<td>$O \left( n^4 \cdot</td>
</tr>
<tr>
<td>Algo. 1</td>
<td>$N$</td>
<td>Absence</td>
<td>$4f^2 + 9f + 4 \leq k$</td>
<td>$O \left( (f +</td>
</tr>
<tr>
<td>Algo. 2</td>
<td>$N$</td>
<td>Absence</td>
<td>$4f^2 + 9f + 4 \leq k$</td>
<td>$O \left( (f +</td>
</tr>
</tbody>
</table>