Consensus and Collision Detectors

1 Overview

Two Goals:

1. Low-level question: How important is collision detection? What are the needed properties of collision detectors? Can devices detect unreliable communication/collisions? If so, how well?
2. Middleware/infrastructure question: What local services can we build? In particular, can we solve agreement (consensus), and if so, how well?

Effectively, we want to understand the interrelated issues of collision detectors and consensus. Or, phrased another way, use the problem of consensus to better understand the requirements of collision detectors.

This lecture is based on:

- "Consensus and Collision Detectors in Wireless Ad Hoc Networks" from PODC 2005
- An early draft of Calvin Newport's Master's thesis.

As a result, this is recent research and there are lots of open problems and ongoing research.

2 Basic Model

Physical system:

- Set I of possible devices.
- Set $P \subseteq I$ of actual devices. Assume $|I| \gg |P|$.
- $n = |P|$.
- $A(i)$ is the algorithm for device $i \in P$. Two subcases:
  - *Anonymous*: $\forall i, j \in I, A(i) = A(j)$. That is, there are no unique ids.
  - *Unique ids*: $\forall i, j \in I, A(i) \neq A(j)$. That is, each algorithm is unique.
- n is unknown. (This is equivalent to saying that algorithm A is correct for all subsets P of I.)
- I is known.

Communication: synchronous, all-to-all broadcast

- round-based communication
- In each round r, for each $i \in P$: $\texttt{bcast}(m)_i$ occurs.
- Let M_r be the set of messages sent in round r.
- Let $N_r[i]$ be the set of messages received by $i \in P$ in round r.

- In each round r, for each $i \in P$: $\texttt{rcv}(N_r[i])_i$ occurs.
- Unreliable: $\forall r, i : N_r[i] \subseteq M_r$.

Eventual Collision Freedom (ECF):

- Intuition: if only one node broadcasts, then there should be no interference.
- $\exists r_{cf}$ such that $\forall r \geq r_{cf}$, if only *one* node broadcasts in round r, then all correct nodes in P receive the message.
- Note: the term "collision" encompasses all receiver-side message loss/omissions.

3 Consensus

- V, value domain
- $\forall i \in P$, $v_i \in V$ is the initial value of i.
- Three properties:
  1. *Agreement*: No two correct processes output different decision values.
  2. *Validity*: If a process decides v, then v is the initial value of some process $i \in P$, i.e., $v = v_i$.
  3. *Termination*: All correct processes output a decision.
- Note: consensus impossible in asynchronous systems with even 1 failure.
- Note: consensus feasible in $f + 1$ rounds in synchronous system with f failures.

Theorem 3.1 Consensus is impossible in the basic model.

Proof. Divide the nodes into two groups: $A = \{v_1, v_2, \ldots, v_{n/2}\}$ and $B = \{v_{n/2+1}, \ldots, v_n\}$. Let all the nodes in V_A start with initial value 0, and all the nodes in V_B start with initial value 1. Let every message sent from a node in A be received by every node in A, and lost by every node in B. Similarly, let every message from a node in B be delivered to every node in B, but lost by every node in A.

Validity implies that every node in A must decide 0, and every node in B must decide 1, resulting in a contradiction. ■

Proof notes:

- The key problem is unreliable broadcast. It allows the network to be partitioned undetectably.
- ECF does not help; r_{CF} may be large.
- Unknown n is critical. Otherwise, it would be possible to distinguish between A and B.

4 Collision Detectors

Intuition: If v_1 and v_3 both send a message at the same time, v_2 may not receive it. A collision detector enables v_2 to detect that this bad event has occurred.

CD Notes:

- Receiver centric. (Sender cannot detect problems.)
- Imperfect. (Hard to implement perfect CD.)
  - False positives $\longrightarrow$ accuracy
  - False negatives $\longrightarrow$ completeness
- Binary: true/false output.
- Notation: $\texttt{CD}_r[j] \in \{\texttt{true}, \texttt{false}\}$.

	Complete	**maj-Complete**	**half-Complete**	**0-Complete**
Accurate	$\mathcal{AC}$	maj-$\mathcal{AC}$	half-$\mathcal{AC}$	0-$\mathcal{AC}$
Eventually Accurate	$\Diamond\mathcal{AC}$	maj-$\Diamond\mathcal{AC}$	half-$\Diamond\mathcal{AC}$	0-$\Diamond\mathcal{AC}$

Figure 1: A summary of collision detector classes.

Completeness

Definition 4.1 A collision detector is *complete* if:

- $\forall r, \forall j \in P$, either:
  1. $|N_r[j]| = |M_r|$ or
  2. $\texttt{CD}_r[j] = \texttt{true}$
- That is, no false negatives.

Definition 4.2 A collision detector is *majority complete* if:

- $\forall r, \forall j \in P$, either:
  1. $|N_r[j]| > |M_r|/2$ or
  2. $\texttt{CD}_r[j] = \texttt{true}$

Definition 4.3 A collision detector is *half complete* if:

- $\forall r, \forall j \in P$, either:
  1. $|N_r[j]| \geq |M_r|/2$ or
  2. $\texttt{CD}_r[j] = \texttt{true}$

Definition 4.4 A collision detector is *zero complete* if:

- $\forall r, \forall j \in P$, either:
  1. if $M_r \neq \emptyset$, then $|N_r[j]| \geq 0$ or
  2. $\texttt{CD}_r[j] = \texttt{true}$

Accuracy

Definition 4.5 A collision detector is *accurate* if:

- $\forall r, \forall j \in P$: if $\texttt{CD}_r[j] = \texttt{true}$, then $N_r[j] \neq M_r$.
- That is, if there is a collision detected, then there was a message lost.

Definition 4.6 A collision detector is *eventually accurate* if:

- $\exists r_{acc}$ such that $\forall r \geq r_{acc}, \forall j \in P$: if $\texttt{CD}_r[j] = \texttt{true}$, then $N_r[j] \neq M_r$.

	Eventual Collision Freedom	**No Collision Freedom**
$\mathcal{AC}$	$\Theta(1)$	$\Theta(\log \lvert V \rvert)$
maj-$\mathcal{AC}$	$\Theta(1)$	$\Theta(\log \lvert V \rvert)$
half-$\mathcal{AC}$	$\Theta(\log \lvert V \rvert)$	$\Theta(\log \lvert V \rvert)$
0-$\mathcal{AC}$	$\Theta(\log \lvert V \rvert)$	$\Theta(\log \lvert V \rvert)$
$\Diamond\mathcal{AC}$	$\Theta(1)$	Impossible
maj-$\Diamond\mathcal{AC}$	$\Theta(1)$	Impossible
half-$\Diamond\mathcal{AC}$	$\Theta(\log \lvert V \rvert)$	Impossible
0-$\Diamond\mathcal{AC}$	$\Theta(\log \lvert V \rvert)$	Impossible
No Accuracy	Impossible	Impossible
No CD	Impossible	Impossible

Table 1: Solving consensus with different collision detector classes.

5 Contention Manager

- Encapsulate randomness / backoff protocols.
- Intuition: eventually selects *one* active node.
- Notations: $\texttt{CM}_r[j] \in \{\texttt{true}, \texttt{false}\}$

Definition 5.1 The contention manager guarantees the following property:

- $\exists r_{cm}, j \in P$ such that:
  - $\forall r \geq r_{cm}$, $\texttt{CM}_r[j] = \texttt{true}$
  - $\forall r \geq r_{cm}, \forall i \neq j$, $\texttt{CM}_r[i] = \texttt{false}$

Backoff protocol example:

1. $x \leftarrow$ flip coin
2. If x is heads, then output `true` and goto step 1.
3. If x is tails, then output `false` and continue to next round.
4. If round is silent, goto step 1, else output `false` and repeat step 4.

6 Consensus

Definition 6.1 Communication Stabilization Time: $\text{CST} = \max(r_{cf}, r_{cm}, r_{acc})$.

Algorithm 1: Consensus with a maj-$\Diamond\mathcal{AC}$ collision detector.

```
1  Process P_i:
2      est_i ∈ V, initially set to the initial value of process P_i
3      r,  round number, initially 0
4
5      Phase 1:
6          if CM_r[i] = true
7              then bcast(est_i)
8          N_r[i] ← rcv()
9          est_i ← min(N_r[i])
10         r ← r+1
11
12     Phase 2:
13         if CD_{r−1}[i] = true or |N_{r−1}| ≠ 1
14             then bcast(veto)
15         N_r[i] ← rcv()
16         if CD_r[i] = false and veto ∉ N_r[i]
17             then decide(est_i)
18         else
19             r ← r+1
20             goto Phase 1
21
```

Proof. Consider agreement:
Assume i decides v in round r. Then:

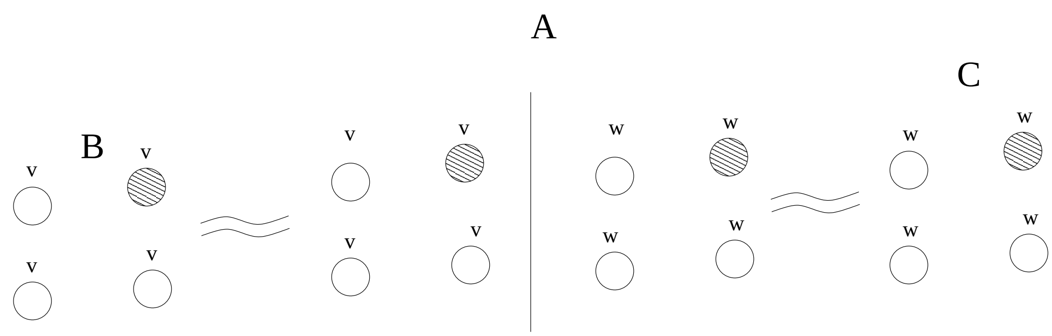
- $\texttt{CD}_r[i] = \texttt{false}$ and $N_r[i] = \emptyset$, by line 16.
- $\implies M_r = \emptyset$ by majority completeness.
- $\implies \forall j \in P, \texttt{CD}_{r-1}[j] = \texttt{false}$ and $|N_{r-1}[j]| = 1$ by line 13.
- $\implies \forall j \in P, |N_{r-1}[j]| > |M_{r-1}|/2$, by majority completeness.
- $\implies \forall j, k \in P, N_{r-1}[j] = N_{r-1}[k]$, since all majorities intersect.
- $\implies \forall j, k \in P, est_j = est_k$ after round $r-1$.

Validity is trivial. Termination: within 3 rounds after CST. ■

7 Lower Bound

Theorem 7.1 Let A be an anonymous consensus algorithm. Then $\exists$ a system S with half-$\Diamond\mathcal{AC}$ and a contention manager, and $\exists$ and execution of S where CST=0 and A does not terminate for $\Omega(\log|V|)$ rounds.

Proof. Consider the following three systems, A, B, and C.



All the nodes in System B start with value v, all the nodes in system C start with value w, and half the nodes in A start with v and half start with w. A is partitioned into two halves, and all the messages from the left half are lost by the right half and vice versa.

Note: the shaded nodes are advised by the contention manager to be awake for the entire execution. CST(B) = CST(C) = 0. By contrast, CST(A) is large.

Key point: If the same number of nodes broadcast in left and right half of A, then $|N_r[j]| = |M_r|/2$ for all $j \in A$, and hence no collision are detected. In this case, nodes in the left half of A cannot distinguish between A and B; nodes in the right half of A cannot distinguish between B and C. We need to identify such a v and w.

Counting: In each round $r \leq (\log|V| - 1)/2$, there are four possible behaviors for nodes in the left half of A:

1. Awake nodes broadcast.
2. Asleep nodes broadcast.
3. All nodes broadcast.
4. No nodes broadcast.

This follows (by induction) from the fact that all the sleeping nodes do the same thing: they start with the same initial value, receive the same contention management input, the same collision detection input, and receive the same messages. (All the same holds for the nodes in the right half of A.)

Therefore, there are how many execution behaviors:

$$4^{\frac{\log|V|-1}{2}} = \frac{|V|}{2}$$

And there are $|V|$ initial values. Therefore, by the pigeon-hole principle, there are two initial values, $v, w \in V$ with the same execution behavior for $(\log|V| - 1)/2$ rounds.

Conclusion:

- All the nodes in B decide v.

- All the nodes in C decide w.
- Assume, by contradiction, that all the nodes in B and C decide by round $(\log |V| - 1)/2$.
- Then the nodes in the left half of A cannot distinguish whether or not they are in B, and decide v.
- Then the nodes in the right half of A cannot distinguish whether or not they are in C, and decide w.
- Contradiction.

■

8 Conclusions

- Collision Detectors:
  1. Can we build them?
  2. 0-complete is most plausible. Is it good enough? What if you have unique identifiers?
  3. Are there other ways of modeling unreliable collision detectors? Probabilistic failure models?
- Consensus
  1. Nack-based protocols paradigm works well.
  2. Are the synchrony assumptions reasonable? What about in the multi-hop case?
  3. Is this a good building block? What else can we build?
- Other:
  1. Is contention management really a separable problem? What about in the multi-hop case?
  2. Can we come up with better upper bounds when unique identifiers are available?
  3. Can we come up with algorithms that are adaptive with respect to completeness? That is, if the collision detector is complete, then they are fast; if the collision detector is 0-complete, then they are slow.