Algorithm 1 Consensus Protocol

receive τ **on coordinator** C **from client:**
1: $t_\tau \leftarrow (\text{now}, 0, C)$
2: **send** $PreAccept(\tau, t_\tau)$ to $\forall p \in \mathbb{E}^\tau$
receive $PreAccept(\tau, t_\tau)$ **on** p**:**
3: **if** $t_\tau > \max(T_\gamma \mid \gamma \sim \tau)$ **then**
4: $t_\tau \leftarrow t_\tau$
5: **else**
6: $t_\tau \leftarrow \max(T_\gamma \mid \gamma \sim \tau)$
7: $t_\tau.(seq, id) \leftarrow (t_\tau.seq + 1, p)$
8: **end if**
9: $T_\tau \leftarrow t_\tau$
10: $PreAccepted_\tau \leftarrow$ **true**
11: **reply** $PreAcceptOK(t_\tau, deps : \{\gamma \mid \gamma \sim \tau \wedge t_{0\gamma} < t_{0\tau}\})$
receive $PreAcceptOK(t, deps)$ **from** Ω^τ**:**
12: $deps \leftarrow \bigcup\{p.deps \mid p \in \Omega^\tau\}$
13: **if** $\exists \mathcal{F}^\tau \subseteq \Omega^\tau\ (\forall p \in \mathcal{F}^\tau : p.t = t_\delta)$ **then**
14: **send** $Commit(\tau, t_\delta, t_\delta, deps)$ to $\forall p \in \rho \in \mathbb{P}^\tau$
15: **go to** Execution Protocol
16: **else**
17: $t \leftarrow \max(p.t \mid p \in \Omega^\tau)$
18: **send** $Accept(\tau, t_\delta, t, deps)$ to $\forall p \in \rho \in \mathbb{P}^\tau$
19: **end if**
receive $Accept(\tau, t_{0\tau}, t_\tau, deps_\tau)$**:**
20: $T_\tau \leftarrow \max(t_\tau, T_\tau)$
21: $Accepted_\tau \leftarrow$ **true**
22: **reply** $AcceptOK(deps : \{\gamma \mid \gamma \sim \tau \wedge t_{0\gamma} < t_\tau\})$
receive $AcceptOK(deps)$ **on** C **from** Ω^τ**:**
23: $deps \leftarrow \bigcup\{p.deps \mid p \in \Omega^\tau\}$
24: **send** $Commit(\tau, t_\tau, t, deps)$ to $\forall p \in \rho \in \mathbb{P}^\tau$
25: **go to** Execution Protocol

Algorithm 2 Execution Protocol

Coordinator C**:**
26: **for** $\rho \in \mathbb{P}^\tau$ **do**
27: $deps_\rho \leftarrow \{\gamma \mid \gamma \in deps \wedge \rho \in \mathbb{P}^\gamma\}$
28: **send** $Read(\tau, t, deps_\rho)$ to some nearby $p \in \rho$
29: **end for**
receive $Commit(\tau, t_{0\tau}, t_\tau, deps_\tau)$**:**
30: $Committed_\tau \leftarrow$ **true**
receive $Read(\tau, t_\tau, deps_{\tau,\rho})$ **on** p**:**
31: **await** $Committed_\gamma\ \forall \gamma \in deps_{\tau,\rho}$
32: **await** $Applied_\gamma\ \forall \gamma \in deps_{\tau,\rho},\ t_\gamma < t_\tau$
33: $reads \leftarrow \text{read}(\tau)$
34: **reply** $ReadOK(reads)$
receive $ReadOK(reads)$ **from each shard:**
35: $result \leftarrow \text{execute}(\tau, reads)$
36: **send** $Apply(\tau, t, deps_{\tau,\rho}, result)$ to $\forall p \in \rho \in \mathbb{P}^\tau$
37: **send** $result$ to client
receive $Apply(\tau, t_\tau, deps_{\tau,\rho}, result_\tau)$**:**
38: **await** $Committed_\gamma\ \forall \gamma \in deps_{\tau,\rho}$
39: **await** $Applied_\gamma\ \forall \gamma \in deps_{\tau,\rho},\ t_\gamma < t_\tau$
40: apply$(writes, t_\tau)$
41: $Applied_\tau \leftarrow$ **true**

Algorithm 3 Recovery Protocol

Coordinator C**:**
1: $b \leftarrow$ fresh ballot
2: **send** $Recover(b, \tau, t_\delta)$ to $\forall p \in \rho \in \mathbb{P}^\tau$
receive $Recover(b, \tau, t_*)$ **on** p**:**
3: **if** $b \leq MaxBallot_\tau$ **then**
4: **reply** $NACK(b_\tau)$
5: **else**
6: $MaxBallot_\tau \leftarrow b$
7: $Accepts \leftarrow \{\gamma \mid \gamma \sim \tau \wedge \tau \notin deps_\gamma \wedge Accepted_\gamma\}$
8: $Commits \leftarrow \{\gamma \mid \gamma \sim \tau \wedge \tau \notin deps_\gamma \wedge Committed_\gamma\}$
9: $Wait \leftarrow \{\gamma \in Accepts \mid t_{0\gamma} < t_{0\tau} \wedge t_\gamma > t_{0\tau}\}$
10: $Superseding \leftarrow \{\gamma \in Accepts \mid t_{0\gamma} > t_{0\tau}\}$
11: $\cup\{\gamma \in Commits \mid t_\gamma > t_{0\tau}\}$
12: **if** $PreAccepted_\tau$ **then**
13: **run** Consensus Protocol 1[3] to 1[10] ▷ *PreAccept*
14: **end if**
15: **if** $\neg Accepted_\tau \wedge \neg Committed_\tau \wedge \neg Applied_\tau$ **then**
16: $deps_\tau \leftarrow \{\gamma \mid \gamma \sim \tau \wedge t_{0\gamma} < t_{0\tau}\}$
17: **end if**
18: **reply** $RecoverOK(*_\tau, Superseding, Wait)$
19: **end if**
receive $NACK$ **on** C**:**
20: **yield** to competing coordinator
receive $RecoverOK(*, Superseding, Wait)$ **from** $p \in \mathcal{R}^\tau$
21: **if** $\exists p \in \mathcal{R}^\tau\ (p.Applied_\tau)$ **then**
22: **send** $response(result)$ to client
23: **send** $Apply(\tau, t_\delta, p.t, p.deps, result)$ to $\forall p' \in \rho \in \mathbb{P}$
24: **else if** $\exists p \in \mathcal{R}^\tau\ (p.Committed_\tau)$ **then**
25: **send** $Commit(\tau, t_\delta, p.t, p.deps) \forall p' \in \rho \in \mathbb{P}^\tau$
26: **go to** Execution Protocol
27: **else if** $\exists p \in \mathcal{R}^\tau\ (Accepted_\tau)$ **then**
28: **select** p with highest $accepted$ ballot
29: $t \leftarrow p.t$; $deps \leftarrow p.deps$
30: **go to** Consensus Protocol 1[18] ▷ Accept
31: **else**
32: $t \leftarrow t_\delta$; $deps \leftarrow \bigcup\{p.deps_\tau \mid p \in \mathcal{R}^\tau\}$
33: **if** $\exists i\ \left|\{p \in \mathbb{E} \mid p.t > p.t_i\}\right| > |\mathbb{E}| - |\mathcal{F}_i|$ **then**
34: $t \leftarrow \max(t \mid p \in \mathcal{R}^\tau)$
35: **else if** $\exists p \in \mathcal{R}^\tau\ (p.Superseding \neq \emptyset)$ **then**
36: $t \leftarrow \max(p.t \mid p \in \mathcal{R}^\tau)$
37: **else if** $\bigcup\{p.Wait \mid p \in \mathcal{R}^\tau\} \neq \emptyset$ **then**
38: **await** $Committed_\tau\ (\forall \gamma \in \bigcup\{p.Wait \mid p \in \mathcal{R}^\tau\})$
39: **restart** Recovery Protocol
40: **end if**
41: **go to** Consensus Protocol 1[18] ▷ Accept
42: **end if**

Algorithm 4 Consensus Protocol (with reconfiguration and ballots)

receive τ **on coordinator** C **from client:**
1: $t_\tau \leftarrow (Epoch, \text{now}, 0, C)$
2: **send** $PreAccept(\tau, t_\tau)$ to $\forall p \in \mathbb{E}^\tau$
receive $PreAccept(\tau, t_{0\tau})$ **on** p**:**
3: **if** $MaxBallot_\tau > 0$ **then**
4: **reply** $NACK$
5: **else if** $PreAccepted_\tau \vee Accepted_\tau \vee Committed_\tau \vee Applied_\tau$ **then**
6: **return**
7: **else**
8: **if** $t_\delta.epoch = Epoch \wedge t_{0\tau} > \max(T_\gamma \mid \gamma \sim \tau) \wedge ReadyElectorate_e$ **then**
9: $t_\tau \leftarrow t_{0\tau}$
10: **else**
11: $t_\tau \leftarrow \max(T_\gamma \mid \gamma \sim \tau)$
12: $t_\tau.(seq, id) \leftarrow (t_\tau.seq + 1, p)$
13: **end if**
14: $T_\tau \leftarrow t_\tau$
15: $PreAccepted_\tau \leftarrow$ **true**
16: **reply** $PreAcceptOK(t_\tau, deps : \{\gamma \mid \gamma \sim \tau \wedge t_{0\gamma} < t_{0\tau}\})$
17: **end if**
receive $PreAcceptOK(t, deps)$ **from** $\Omega^\tau_{t_0}$ **where** $\Omega^\tau_{t_0} \not\supseteq \Omega^\tau_t$ **:**
18: **send** $PreAccept(\tau, t_0)$ to $\forall p \in \mathbb{E}^\tau_t \setminus \mathbb{E}^\tau_{t_0}$
receive $PreAcceptOK(t, deps)$ **from** $\Omega^\tau \supseteq \Omega^\tau_{t_0} \cup \Omega^\tau_t$**:**
19: $deps \leftarrow \bigcup\{p.deps \mid p \in \Omega^\tau\}$
20: **if** $\exists \mathcal{F}^\tau \subseteq \Omega^\tau\ (\forall p \in \mathcal{F}^\tau : p.t = t_\delta)$ **then**
21: **send** $Commit(\tau, t_\delta, t_\delta, deps)$ to $\forall p \in \rho \in \mathbb{P}^\tau_{t_\delta} \cup \mathbb{P}^\tau_t$
22: **go to** Execution Protocol
23: **else**
24: $t \leftarrow \max(p.t \mid p \in \Omega^\tau)$
25: **send** $Accept(0, \tau, t_\delta, t, deps)$ to $\forall p \in \rho \in \mathbb{P}^\tau_{t_\delta} \cup \mathbb{P}^\tau_t$
26: **end if**
receive $Accept(b, \tau, t_{0\tau}, t_\tau, deps_\tau)$**:**
27: **if** $b < MaxBallot_\tau$ **then**
28: **reply** $NACK$
29: **else if** $Committed_\tau \vee Applied_\tau$ **then**
30: **return**
31: **else**
32: $MaxBallot_\tau \leftarrow b$
33: $AcceptedBallot_\tau \leftarrow b$
34: $T_\tau \leftarrow \max(t_\tau, T_\tau)$
35: $Accepted_\tau \leftarrow$ **true**
36: **reply** $AcceptOK(deps : \{\gamma \mid \gamma \sim \tau \wedge t_{0\gamma} < t_\tau\})$
37: **end if**
receive $NACK$ **on** C**:**
38: **yield** to competing coordinator
receive $AcceptOK(deps)$ **on** C **from** $\Omega^\tau_{t_0} \cup \Omega^\tau_t$**:**
39: $deps \leftarrow \bigcup\{p.deps \mid p \in \Omega^\tau\}$
40: **send** $Commit(\tau, t_0, t, deps)$ to $\forall p \in \rho \in \mathbb{P}^\tau_{t_0} \cup \mathbb{P}^\tau_t$
41: **go to** Execution Protocol

Algorithm 5 Execution Protocol (with reconfiguration and ballots)

Coordinator C**:**
42: **for** $\rho \in \mathbb{P}^\tau$ **do**
43: $deps_\rho \leftarrow \{\gamma \mid \gamma \in deps \wedge \rho \in \mathbb{P}^\gamma\}$
44: **send** $Read(\tau, t, deps_\rho)$ to some nearby correct $p \in \rho$
45: **end for**
receive $Commit(\tau, t_{0\tau}, t_\tau, deps_\tau)$**:**
46: $Committed_\tau \leftarrow$ **true**
receive $Read(\tau, t_\tau, deps_{\tau,\rho})$ **on** p**:**
47: **await** $Committed_\gamma\ \forall \gamma \in deps_{\tau,\rho}$
48: **await** $Applied_\gamma\ \forall \gamma \in deps_{\tau,\rho},\ t_\gamma < t_\tau$
49: $reads \leftarrow \text{read}(\tau)$
50: **reply** $ReadOK(reads)$
receive $ReadOK(reads)$ **from each shard:**
51: $result \leftarrow \text{execute}(\tau, reads)$
52: **send** $Apply(\tau, t, deps_{\tau,\rho}, result)$ to $\forall p \in \rho \in \mathbb{P}^\tau_t$
53: **send** $result$ to client
receive $Apply(\tau, t_\tau, deps_{\tau,\rho}, result_\tau)$**:**
54: **if** $\neg Applied_\tau$ **then**
55: **await** $Committed_\gamma\ \forall \gamma \in deps_{\tau,\rho}$
56: **await** $Applied_\gamma\ \forall \gamma \in deps_{\tau,\rho},\ t_\gamma < t_\tau$
57: apply$(writes, t_\tau)$
58: $Applied_\tau \leftarrow$ **true**
59: **end if**

Algorithm 6 Reconfiguration Protocol

receive new configuration $Config_{e+1}$ **on process** p **:**
1: **await** $ReadyEpoch = e$
2: **if** $p \in \mathbb{E}_{e-1}$ **then**
3: **await** $ReadyElectorate_e$
4: **send** $JoinElectorate(\{\tau \mid t_\tau = t_{0\tau}\}, e+1)$ to $\forall p \in (\mathbb{E}_{e+1} \setminus \mathbb{E}_e)$
5: **end if**
6: **if** $p \in \mathbb{E}_{e+1} \wedge p \notin \mathbb{E}_e$ **then**
7: $ReadyElectorate_{e+1} \leftarrow$ **false**
8: **else**
9: $ReadyElectorate_{e+1} \leftarrow$ **true**
10: **end if**
11: $Epoch \leftarrow e + 1$
12: $ReadyReconfigure_{e+1} \leftarrow$ **true**
13: **if** $p \in \rho_e \cup \rho_{e+1}$ **then**
14: **await** $ReadyShard$
15: **await** $Committed_\tau \cdot \forall \tau \in \{\tau \mid PreAccepted_\tau \wedge t_\tau.epoch \leq e\}$ ▷ Should only wait for those notified on join, so introduce a special flag, to prevent forward progress
16: **await** $Applied_\tau \cdot \forall \tau \in \{\tau \mid Committed_\tau \wedge t_\tau.epoch \leq e\}$
17: **send** $JoinShard(\{(\tau, *_\tau) \mid Applied_\tau \wedge t_\tau.epoch \leq e\}, e+1)$ to $\forall p \in (\rho_{e+1} \setminus \rho_e)$ ▷ TODO: also send commits that are dependencies!
18: **end if**
19: $ReadyEpoch \leftarrow e$
receive $JoinElectorate(Txns, e)$ **on replica** p **from** p' **:**
20: **await** $ReadyReconfigure_{e+1}$
21: **for** $\forall \tau \in Txns$ **do**
22: **if** $\neg PreAccepted_\tau$ **then**
23: **run** Consensus Protocol 1[8] to 1[15] ▷ *PreAccept*
24: **end if**
25: **end for**
26: $ReceivedJoinElectorate_{e+1} \leftarrow ReceivedJoinElectorate_{e+1} \cup \{p'\}$
27: $ReadyElectorate_{e+1} \leftarrow |ReceivedJoinElectorate_{e+1}| > \mathbb{E}_e - \mathcal{F}_e$
receive $JoinShard(Txns, e+1)$ **on replica** p **from** p' **:**
28: **await** $ReadyReconfigure_{e+1}$
29: **for** $\forall (\tau, *_\tau) \in Txns$ **do**
30: **if** $Applied_\tau$ **then**
31: **async receive** $Apply(\tau, t_\tau, deps_{\tau,\rho}, result_\tau)$
32: **else if** $Committed_\tau$ **then**
33: **async receive** $Commit(\tau, t_{0\tau}, t_\tau, deps_\tau)$
34: **end if**
35: **end for**
36: **await**
37: $ReceivedJoinShard_{e+1} \leftarrow ReceivedJoinShard_{e+1} \cup \{p'\}$
38: $ReadyShard \leftarrow \exists \mathbb{Q}_e \subseteq ReceivedJoinShard_{e+1}$

Algorithm 7 Recovery Protocol

Coordinator C**:**
1: $b \leftarrow$ fresh ballot
2: **send** $Recover(b, \tau, t_\delta)$ to $\forall p \in \rho \in \mathbb{P}^\tau_{[t_\delta]}$
receive $Recover(b, \tau, t_*)$ **on** p**:**
3: **if** $b \leq MaxBallot_\tau$ **then**
4: **reply** $NACK(b_\tau)$
5: **else**
6: $MaxBallot_\tau \leftarrow b$
7: $Accepts \leftarrow \{\gamma \mid \gamma \sim \tau \wedge \tau \notin deps_\gamma \wedge Accepted_\gamma\}$
8: $Commits \leftarrow \{\gamma \mid \gamma \sim \tau \wedge \tau \notin deps_\gamma \wedge Committed_\gamma\}$
9: $Wait \leftarrow \{\gamma \in Accepts \mid t_{0\gamma} < t_{0\tau} \wedge t_\gamma > t_{0\tau}\}$
10: $Superseding \leftarrow \{\gamma \in Accepts \mid t_{0\gamma} > t_{0\tau}\}$
11: $\cup\{\gamma \in Commits \mid t_\gamma > t_{0\tau}\}$
12: **if** $PreAccepted_\tau$ **then**
13: **run** Consensus Protocol 1[8] to 1[15] ▷ *PreAccept*
14: **end if**
15: **if** $\neg Accepted_\tau \wedge \neg Committed_\tau \wedge \neg Applied_\tau$ **then**
16: $deps_\tau \leftarrow \{\gamma \mid \gamma \sim \tau \wedge t_{0\gamma} < t_{0\tau}\}$
17: **end if**
18: **reply** $RecoverOK(*_\tau, Superseding, Wait)$
19: **end if**
receive $NACK$ **on** C**:**
20: **yield** to competing coordinator
receive $RecoverOK(*, Superseding, Wait)$ **from** $p \in \mathcal{R}^\tau \supseteq \Omega^\tau_{t_0}$
21: **if** $\exists p \in \mathcal{R}^\tau\ (p.Applied_\tau)$ **then**
22: **send** $response(result)$ to client
23: **send** $Apply(\tau, t_\delta, p.t, p.deps, result)$ to $\forall p' \in \rho \in \mathbb{P}^\tau_t$
24: **else if** $\exists p \in \mathcal{R}^\tau\ (p.Committed_\tau)$ **then**
25: **send** $Commit(\tau, t_\delta, p.t, p.deps) \forall p' \in \rho \in \mathbb{P}^\tau_{[t_\delta]} \cup \mathbb{P}^\tau_t$
26: **go to** Execution Protocol
27: **else if** $\exists p \in \mathcal{R}^\tau\ (Accepted_\tau)$ **then**
28: **select** p with highest $AcceptedBallot_\tau$
29: $t \leftarrow p.t$; $deps \leftarrow p.deps$
30: **go to** Consensus Protocol 1[19] ▷ Accept
31: **else**
32: $t \leftarrow t_\delta$; $deps \leftarrow \bigcup\{p.deps_\tau \mid p \in \mathcal{R}^\tau\}$
33: **if** $\exists i\ \left|\{p \in \mathbb{E} \mid p.t > p.t_i\}\right| > |\mathbb{E}| - |\mathcal{F}_i|$ **then**
34: $t \leftarrow \max(p.t \mid p \in \mathcal{R}^\tau)$
35: **else if** $\exists p \in \mathcal{R}^\tau\ (p.Superseding \neq \emptyset)$ **then**
36: $t \leftarrow \max(p.t \mid p \in \mathcal{R}^\tau)$
37: **else if** $\bigcup\{p.Wait \mid p \in \mathcal{R}^\tau\} \neq \emptyset$ **then**
38: **await** $Committed_\tau\ (\forall \gamma \in \bigcup\{p.Wait \mid p \in \mathcal{R}^\tau\})$
39: **restart** Recovery Protocol
40: **end if**
41: **go to** Consensus Protocol 1[19] ▷ Accept
42: **end if**

The Accord Consensus Protocol

Fast General Purpose Transactions

Elliott Smith, Benedict Miller, Alex — *Zhang, Tony Capwell, David* — *Eggleston, Blake Andreas, Scott Yeschenko, Aleksey* — *Dr. Howard, Heidi Weisberg, Ariel*

True ACID Compliance	→ Global Strict Serialisability
Transactions	→ Multi-partition, Multi-table
Resilience & Linear scalability	→ Leaderless, Unique Recovery (No rollbacks)
Performance	→ Single Network Round-trip
Accessible	→ Commodity Clocks → Standalone library (not tied to C*)
Novel	→ Flexible Fast-Path Electorates → Timestamp Reorder Buffer