Consistency

chenzongzhi@360.cn

What is consistency

- Get multiple servers to agree on state

Consistency

- Weak
- Eventual
- Strong

Weak consistency

- After a write, reads may or may not see it
- Best effort only
- Used for Cache

Eventual consistency

- After a write, reads will eventually see it
- App: Mail, Search engine indexing, Bada

Strong consistency

- After a write, read will see it
- App Engine: datastore
- File System

Consensus Algorithm

- 2PC
- Paxos
- Zab
- Raft

Paxos

Paxos Roles

- Proposers
- acceptors
- learners

Two Phases

Phase 1a: Prepare

- A Proposer creates a proposal identified with a number N. This number must be greater than any previous proposal number used by this Proposer. Then, it sends a Prepare message containing this proposal to a Quorum of Acceptors. The Proposer decides who is in the Quorum.

Phase 1b: Promise

- If the proposal's number N is higher than any previous proposal number received from any Proposer by the Acceptor, then the Acceptor must return a promise to ignore all future proposals having a number less than N. If the Acceptor accepted a proposal at some point in the past, it must include the previous proposal number and previous value in its response to the Proposer.

Phase 2a: Accept Request

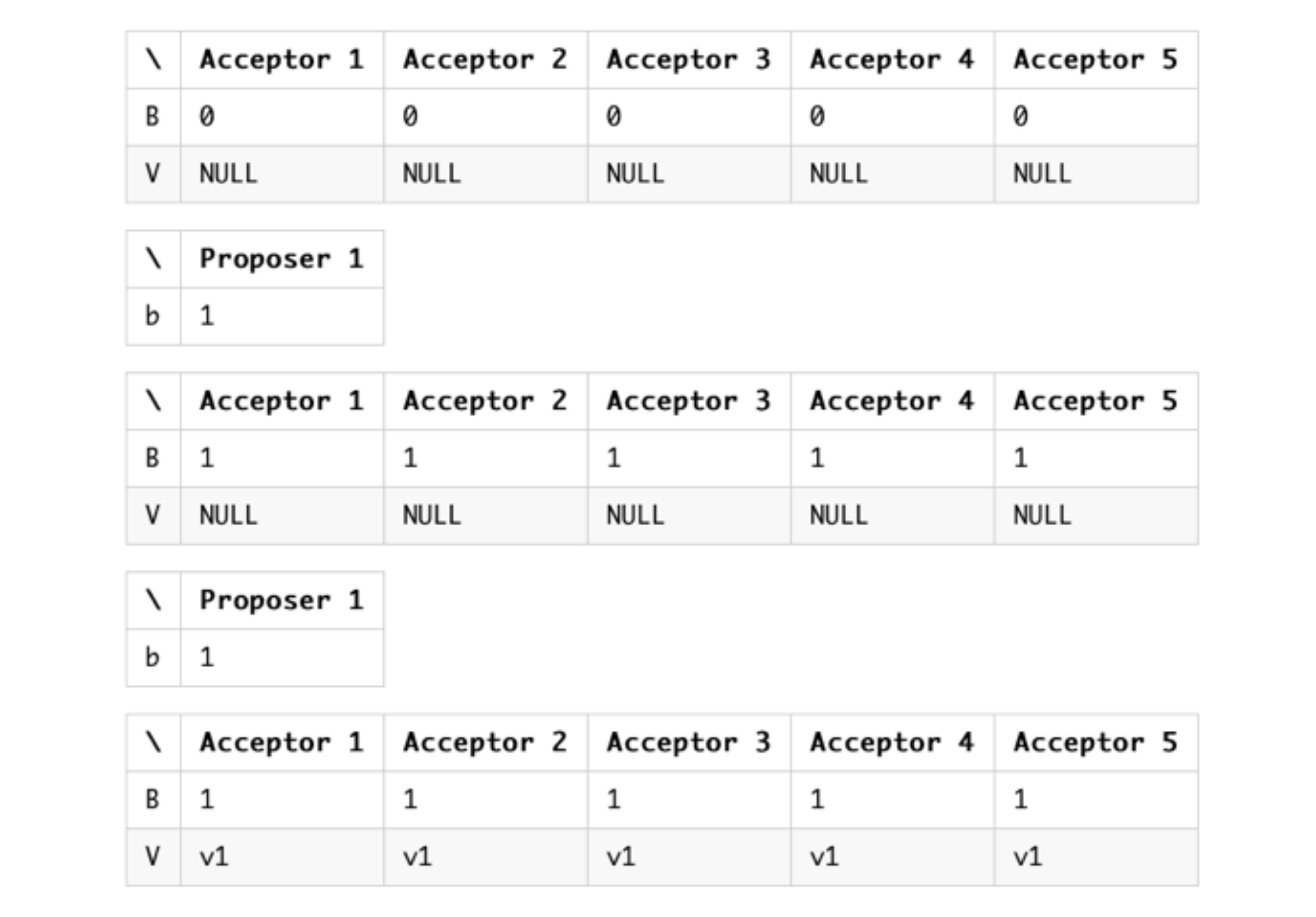
- If a Proposer receives enough promises from a Quorum of Acceptors, it needs to set a value to its proposal. If any Acceptors had previously accepted any proposal, then they'll have sent their values to the Proposer, who now must set the value of its proposal to the value associated with the highest proposal number reported by the Acceptors. If none of the Acceptors had accepted a proposal up to this point, then the Proposer may choose any value for its proposal.

Phase 2b: Accept

- If an Acceptor receives an Accept Request message for a proposal N, it must accept it if and only if it has not already promised to only consider proposals having an identifier greater than N. In this case, it should register the corresponding value v and send an Accepted message to the Proposer and every Learner. Else, it can ignore the Accept Request.

Examples

Base Example



\	Acceptor 1	Acceptor 2	Acceptor 3	Acceptor 4	Acceptor 5
B	0	0	0	0	0
V	NULL	NULL	NULL	NULL	NULL

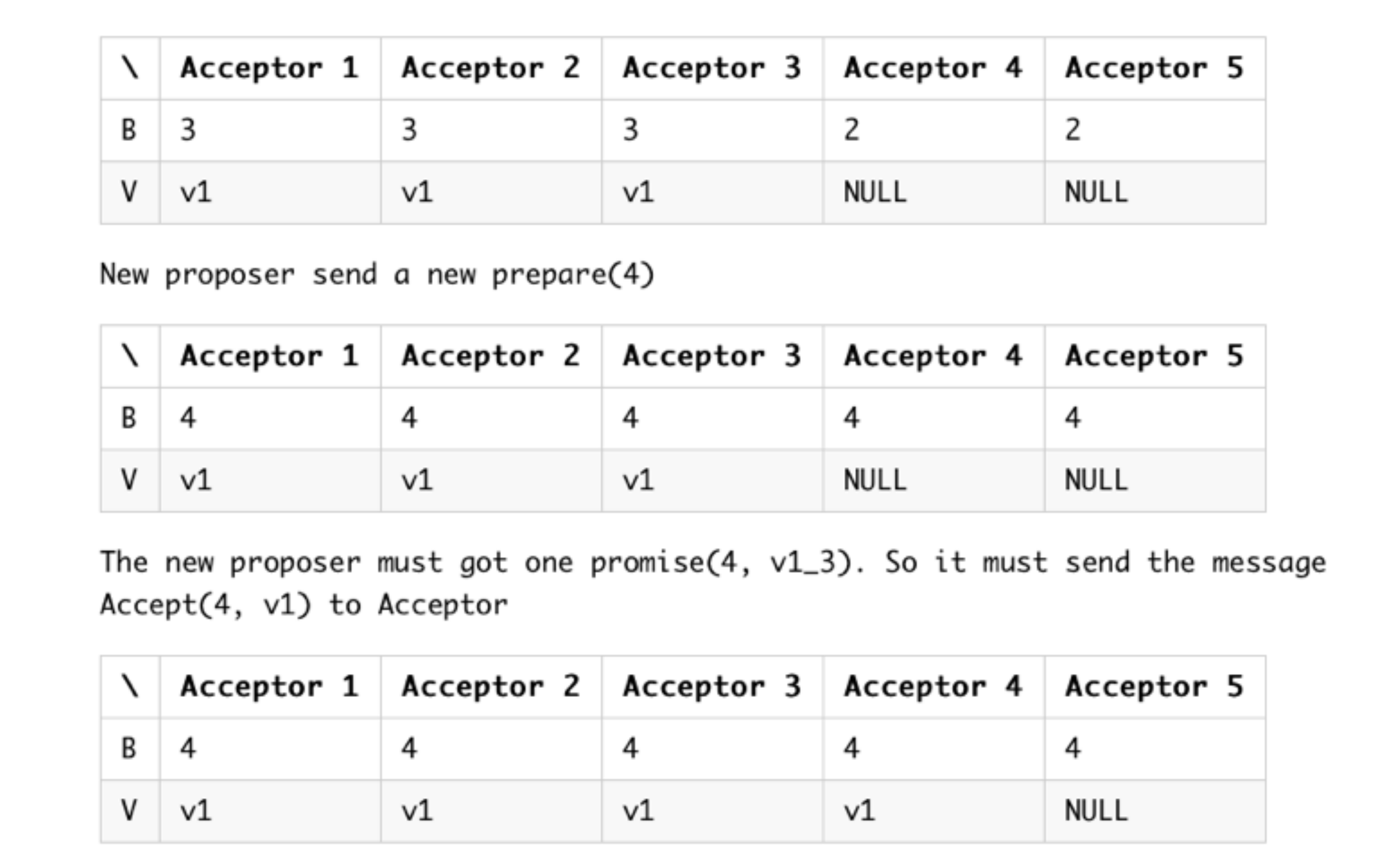
\	Proposer 1
b	1

\	Acceptor 1	Acceptor 2	Acceptor 3	Acceptor 4	Acceptor 5
B	1	1	1	1	1
V	NULL	NULL	NULL	NULL	NULL

\	Proposer 1
b	1

\	Acceptor 1	Acceptor 2	Acceptor 3	Acceptor 4	Acceptor 5
B	1	1	1	1	1
V	v1	v1	v1	v1	v1

Example



\	**Acceptor 1**	**Acceptor 2**	**Acceptor 3**	**Acceptor 4**	**Acceptor 5**
B	3	3	3	2	2
V	v1	v1	v1	NULL	NULL

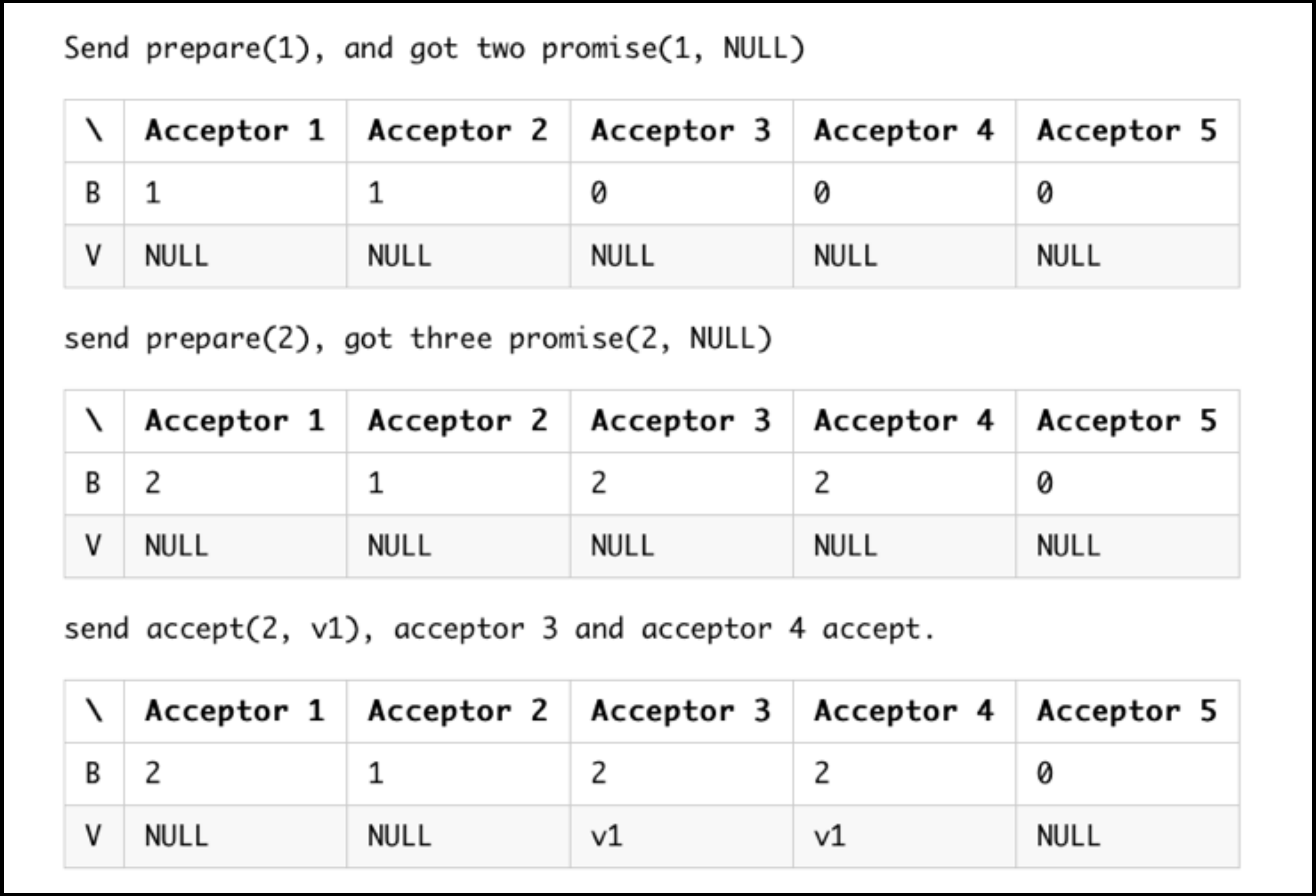
New proposer send a new prepare(4)

\	**Acceptor 1**	**Acceptor 2**	**Acceptor 3**	**Acceptor 4**	**Acceptor 5**
B	4	4	4	4	4
V	v1	v1	v1	NULL	NULL

The new proposer must got one promise(4, v1_3). So it must send the message Accept(4, v1) to Acceptor

\	**Acceptor 1**	**Acceptor 2**	**Acceptor 3**	**Acceptor 4**	**Acceptor 5**
B	4	4	4	4	4
V	v1	v1	v1	v1	NULL

Example:conflict



Send prepare(1), and got two promise(1, NULL)

\	Acceptor 1	Acceptor 2	Acceptor 3	Acceptor 4	Acceptor 5
B	1	1	0	0	0
V	NULL	NULL	NULL	NULL	NULL

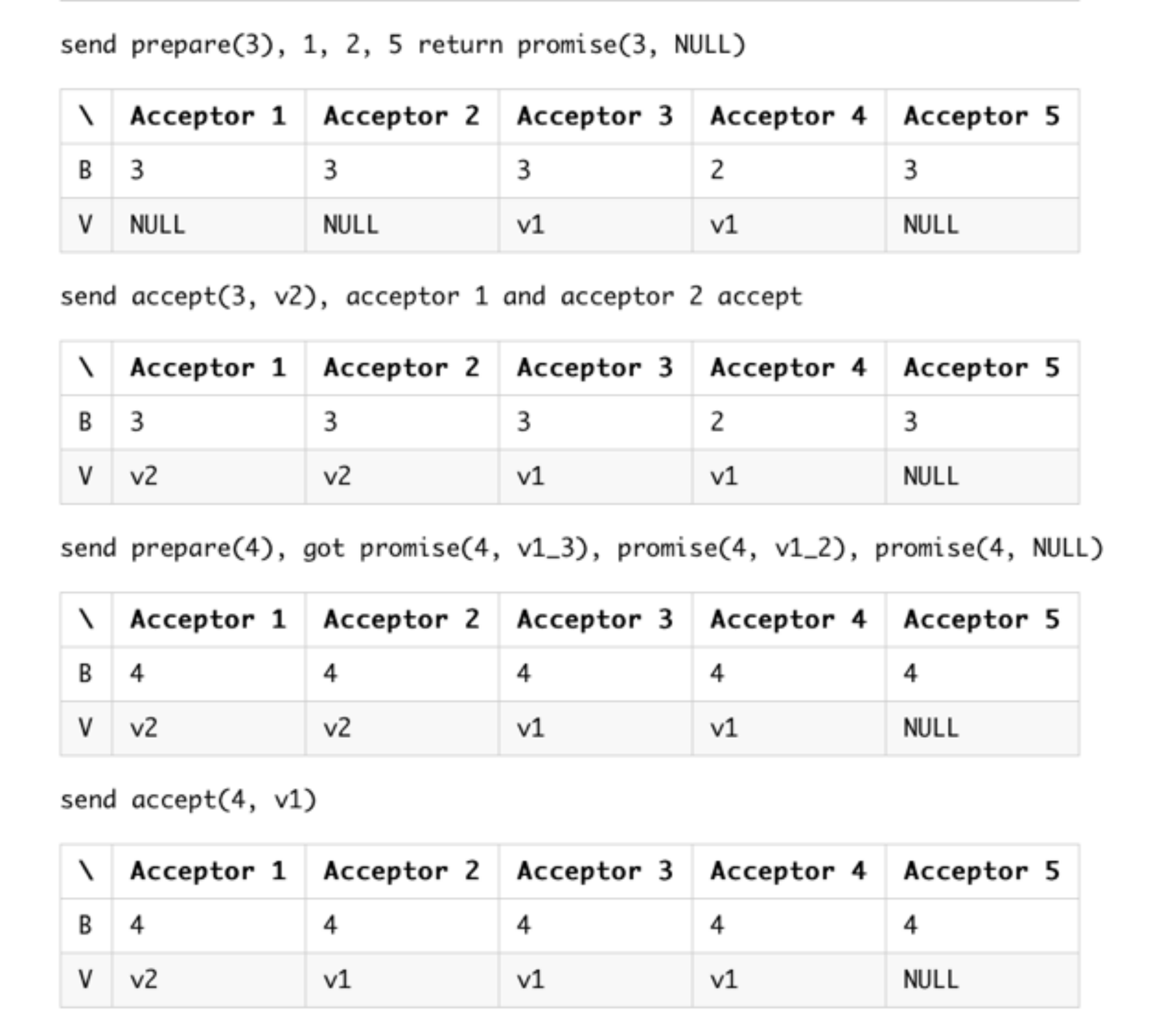
send prepare(2), got three promise(2, NULL)

\	Acceptor 1	Acceptor 2	Acceptor 3	Acceptor 4	Acceptor 5
B	2	1	2	2	0
V	NULL	NULL	NULL	NULL	NULL

send accept(2, v1), acceptor 3 and acceptor 4 accept.

\	Acceptor 1	Acceptor 2	Acceptor 3	Acceptor 4	Acceptor 5
B	2	1	2	2	0
V	NULL	NULL	v1	v1	NULL

Example:conflict



send prepare(3), 1, 2, 5 return promise(3, NULL)

\	Acceptor 1	Acceptor 2	Acceptor 3	Acceptor 4	Acceptor 5
B	3	3	3	2	3
V	NULL	NULL	v1	v1	NULL

send accept(3, v2), acceptor 1 and acceptor 2 accept

\	Acceptor 1	Acceptor 2	Acceptor 3	Acceptor 4	Acceptor 5
B	3	3	3	2	3
V	v2	v2	v1	v1	NULL

send prepare(4), got promise(4, v1_3), promise(4, v1_2), promise(4, NULL)

\	Acceptor 1	Acceptor 2	Acceptor 3	Acceptor 4	Acceptor 5
B	4	4	4	4	4
V	v2	v2	v1	v1	NULL

send accept(4, v1)

\	Acceptor 1	Acceptor 2	Acceptor 3	Acceptor 4	Acceptor 5
B	4	4	4	4	4
V	v2	v1	v1	v1	NULL

Deduction

- Only a Value be accepted in a Paxos example
- If a Value has accepted by more than half of the Acceptors, later acceptors can only accept this value
- Later proposer must propose this value
- If a value v promise by more than half of the Acceptors, later proposer can only proposer value v
- Before Proposer propose a value v, there must be no value been accept, or there is a set more than half of the acceptors a value

Usage

- Paxos in Aerospike
- Paxos in Zookeeper/Chubby
- Paxos in Megastore

Limit

- Need two phase
- Live Lock

Reference

- Paxos make simple
- Paxos make live
- lamport Paxos

Next

- Multi-Paxos
- Rafter

Thanks