

8 Fault Tolerant Distributed Transactions : 2PC

- 8.1 One Phase Commit
- 8.2 3PC: nonblocking
- 8.2 Paxos consensus
- 8.6 Paxos in practice

x based on Weikum / Vossen; Valduriez / Öszu; Garcia-Molina ; Reuter/ Gray

8.1 One phase commit

Example: Calendar application

Application protocol: agreement on the date / time of some event.

e.g:

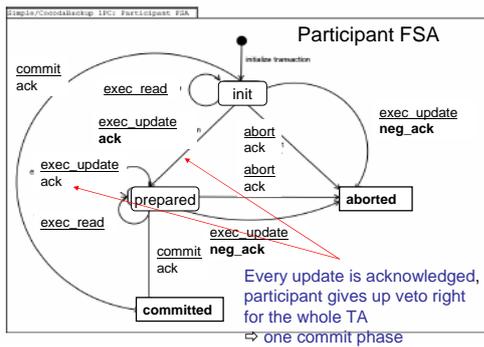
```
".. everyone happy with suggested date?
   if one participant votes no,
       coordinator makes new suggestion
   else commit (1-phase)"
```

Agreement between nodes in processing phase, not during commit.

HS-20010

HS / 08-TA-ACP2- 2

1PC: participant protocol



HS-20010

slide: J. Bross

HS / 08-TA-ACP2- 3

Characteristics of 1PC

Blocking?

Yes! When?

Two types of blocking:

- participant failure
- coordinator failure – more serious, why?

Window of uncertainty in failure free case?

Number of messages for commit /abort?
Suppose n participants.

HS-20010

HS / 08-TA-ACP2- 4

More involved task

n participants, each having a variable x_i

clients send increments ("+") to each of them
no individual ack of an increment operation, (but of msg received)

---- end of operation phase -----

Condition for successful operation: all increments successful (no overflow, or alike)

If not successful: participants reset x_i

Commit coordinator has to decide!

Commit phase? 1PC is not sufficient to come to a unanimous result! Why?

— work phase

— commit phase

HS-20010

HS / 08-TA-ACP2- 5

8.2 Three phase commit: the basic idea

Observation:

There is no distributed commit protocol (DCP) which avoids blocking with multiple failures.*

⇒ no independent recovery of failed processes (nodes) in general

⇒ External input needed for learning the fate of the TA, i.e. commit / abort decision depends on coordinator input

Basic idea: Introduce a new state which avoids the dependency on external input

HS-20010

*with least one. communication failure

HS / 08-TA-ACP2- 6

Last lecture: Useful Invariant

Goal: avoid blocking!

"If a participant P is uncertain then there is no P' which got a commit decision if P, P' are alive or not" (*)

Have shown: **Invariant does not hold for 2PC**

(*) + only site failures: uncertain participants can decide to abort the TA

⇒ Find a protocol which satisfies (*)

HS-20010

HS / 08-TA-ACP2-7

Blocking

Can blocking be avoided?

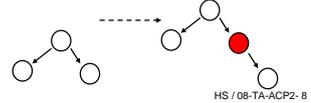
YES ...?

No "communication failure" assumption:

Process may fail during execution of commit protocol, but no messages lost

If the failure assumption (no communication failure) holds, there is a non-blocking distributed commit

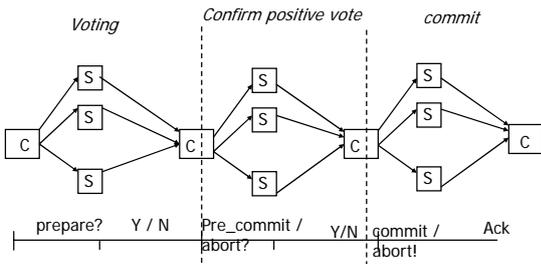
Proof idea: avoid state transitions dependent on external input



HS-20010

HS / 08-TA-ACP2-8

Non-blocking commit, three phases

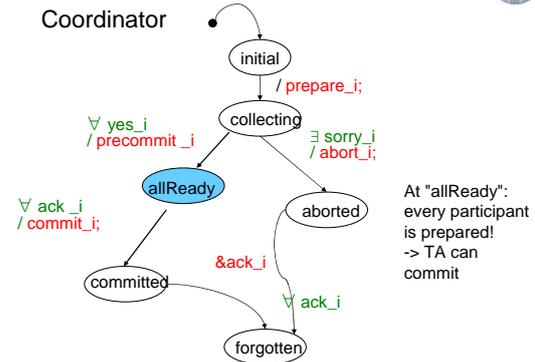


HS-20010

HS / 08-TA-ACP2-9

3 PC - coordinator

Coordinator

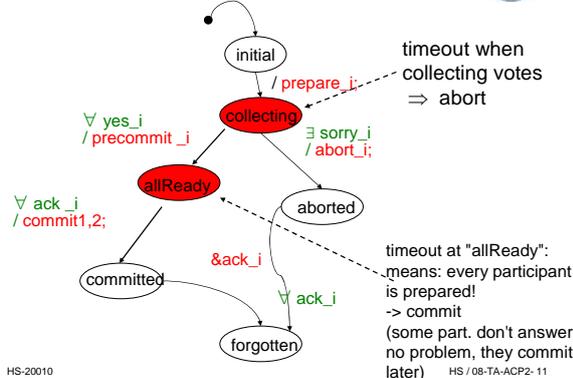


At "allReady": every participant is prepared! -> TA can commit

HS-20010

HS / 08-TA-ACP2-10

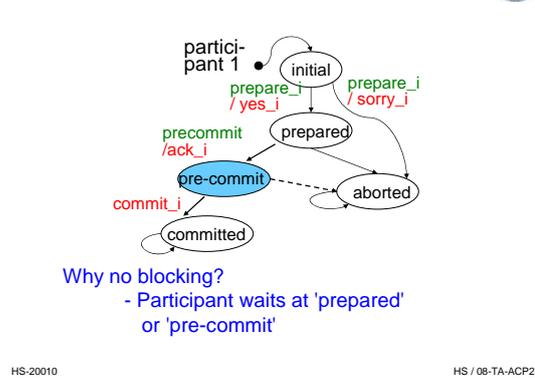
Coordinator termination



HS-20010

HS / 08-TA-ACP2-11

3PC - participant



Why no blocking?
- Participant waits at 'prepared' or 'pre-commit'

HS-20010

HS / 08-TA-ACP2-12

Termination for prepared participants

Participants

particip-
ant 1

initial

prepare_i / yes_i

prepare_i / sorry_i

prepared

?

Timeout in "prepared":
Find a **new coordinator**
and terminate according
to state of other participants

NewCoord:
"abort" ⇒ Action "abort"

Timeout in "pre-commit": in an analogue manner

NewCoord:
"prepared" ⇒ other participants
in "prep" | "abort" |"
Action: abort

NewCoord:
"pre-c" ⇒ other participants in
"prep" | "pre-c" | "commit"
Action: "commit"
idempotent? ⇒ continue
3PC-coordinator protocol

HS-20010 HS / 08-TA-ACP2- 13

Net Partitioning

Why does protocol not work in partitioned net?

prepared

prepared

pre-commit

pre-commit

prepared

Decides: abort

Decides: commit

HS-20010 HS / 08-TA-ACP2- 14

How important is 3PC?

Three phase commit challenging...

... but not really important in practice – up to now (?)

Message overhead
("Make typical case fast", is failure typical?)

Practice today:

- make 2 PC fast,
- reduce blocking probability
- blocking seems to be important (see Oracle's commit point), but no serious analysis

HS-20010 HS / 08-TA-ACP2- 16

All kinds of variants

e.g

Early Commit:

- Send commit with last operation
- leads to unblocking, but uncertainty
- compensation if final decision is abort

"Early Prepare"

- Send prepare with last operation
- may lead to longer blocking phase
- what else? (-> Übungen)

HS-20010 HS / 08-TA-ACP2- 17