Flooding Oscillations

Bruno Rijsman, 16-Oct-2018, v1.2

Current Flooding Scope Rules

	South	North	East-West
Node S-TIE	Flood if level of originator is equal to this node.	Flood if level of originator is higher than this node.	Flood only if this node is not top-of-fabric.
Non-Node S-TIE	Flood self-originated only.	Flood only if neighbor is originator of TIE.	Flood only if self-originated and this node is not top-of-fabric.
All N-TIEs	Never flood.	Flood always.	Flood only if this node is top-of-fabric.
TIDE	Include at least TIEs in flooding scope		
TIRE	Include all N-TIEs and all peer's self-originated TIEs and all node S-TIEs	Flood only if neighbor is originator of TIE.	If this node is top-of-fabric then apply north scope rules, otherwise south scope rules

Introducing a Bit of Notation

```
in_flood_scope(from_node=X, to_node=Y, tie=T)
```

- Returns true if node X **MUST** flood TIE T to node Y
- Returns false if node X **MUST NOT** flood TIE T to node Y
- If false, node Y **MUST NOT** accept and reflood TIE T from node X
  "Belt and suspenders": Y evaluates function "from perspective of X"
- "Node Y is in the flooding scope of node X for TIE T"
- We are only talking about TIEs here
  (Not discussing TIREs or TIDEs yet – will be discussed later)

Information needed to evaluate flood scope

```
in_flood_scope(from_node=X, to_node=Y, tie=T)
```

From node

- System ID
- Level
- Top of Fabric

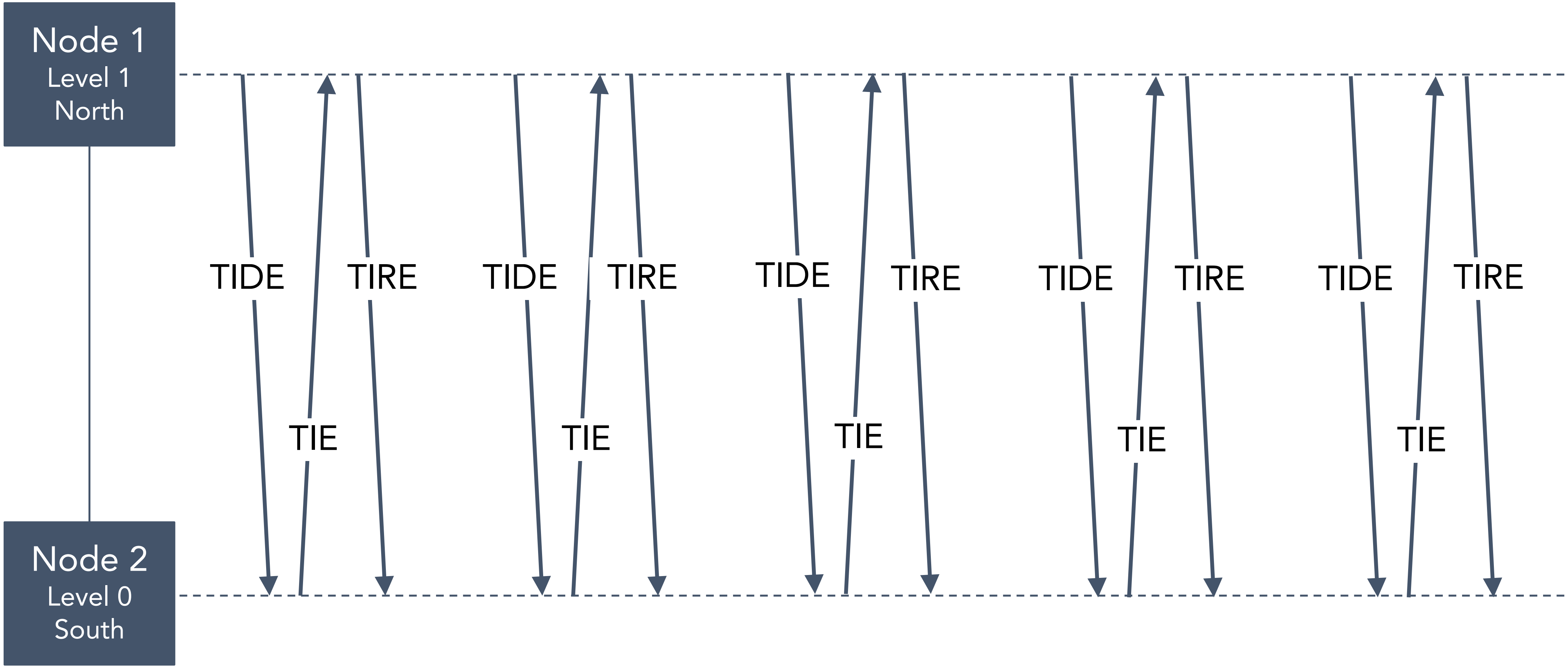
To node

- System ID
- Level
  for Direction (N,S,EW)
- Note: to_node is always a neighbor of from_node

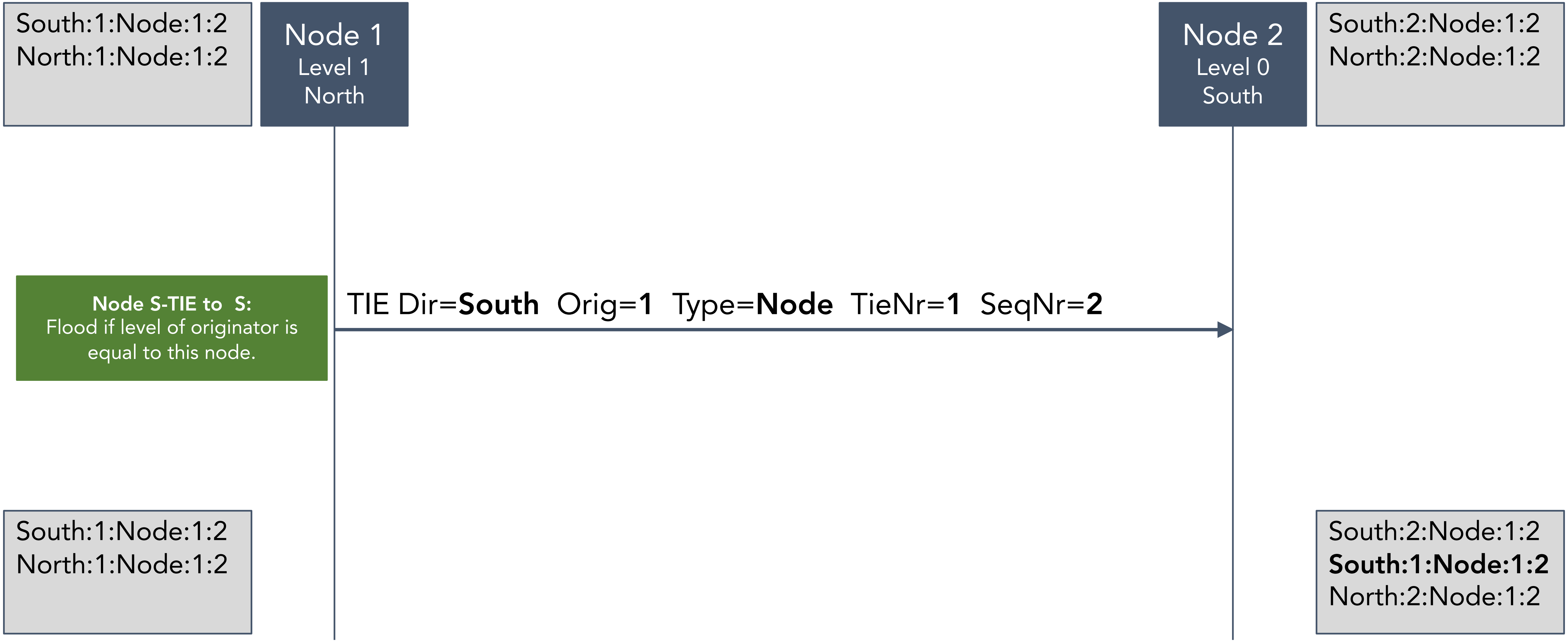
TIE

- Direction (N,S)
- Type
- Originator
  for is self-originated flag
- Originator level
  only for Node TIEs
  only thing not in header

Oscilation #1



Node 1 sends its node TIE to node 2



South:1:Node:1:2
North:1:Node:1:2

Node 1
Level 1
North

Node 2
Level 0
South

South:2:Node:1:2
North:2:Node:1:2

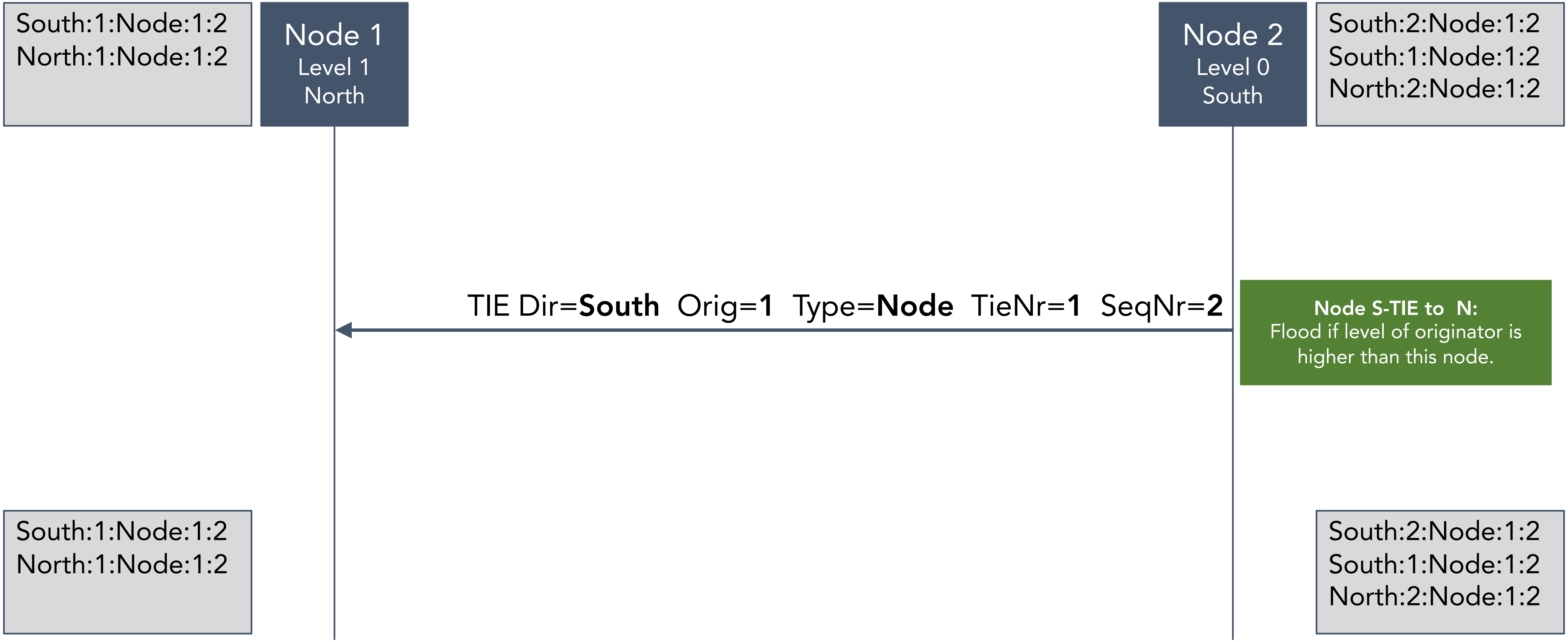
Node S-TIE to S:
Flood if level of originator is equal to this node.

TIE Dir=**South** Orig=**1** Type=**Node** TieNr=**1** SeqNr=**2**

South:1:Node:1:2
North:1:Node:1:2

South:2:Node:1:2
South:1:Node:1:2
North:2:Node:1:2

Node 2 reflects the TIE



South:1:Node:1:2
North:1:Node:1:2

Node 1
Level 1
North

Node 2
Level 0
South

South:2:Node:1:2
South:1:Node:1:2
North:2:Node:1:2

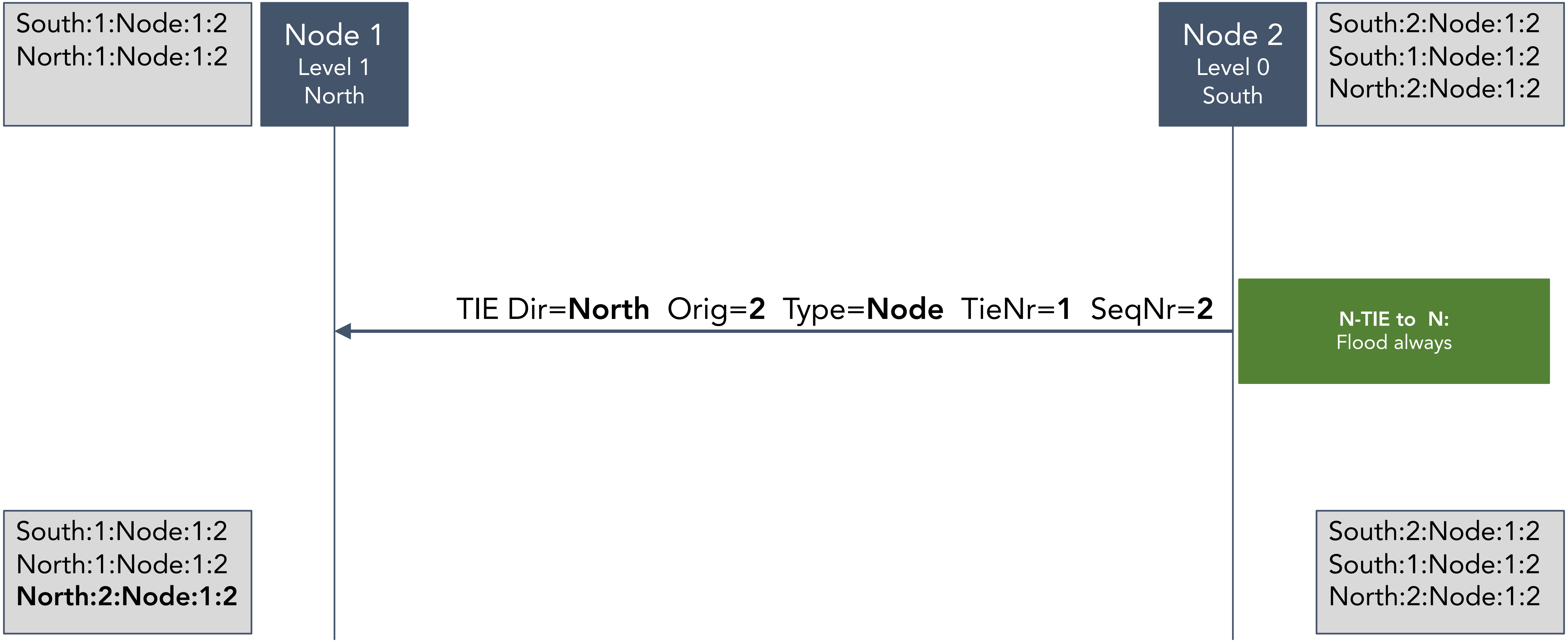
TIE Dir=**South** Orig=**1** Type=**Node** TieNr=**1** SeqNr=**2**

Node S-TIE to N:
Flood if level of originator is higher than this node.

South:1:Node:1:2
North:1:Node:1:2

South:2:Node:1:2
South:1:Node:1:2
North:2:Node:1:2

Node 2 sends its node TIE to node 1



South:1:Node:1:2
North:1:Node:1:2

Node 1
Level 1
North

Node 2
Level 0
South

South:2:Node:1:2
South:1:Node:1:2
North:2:Node:1:2

TIE Dir=**North** Orig=**2** Type=**Node** TieNr=**1** SeqNr=**2**

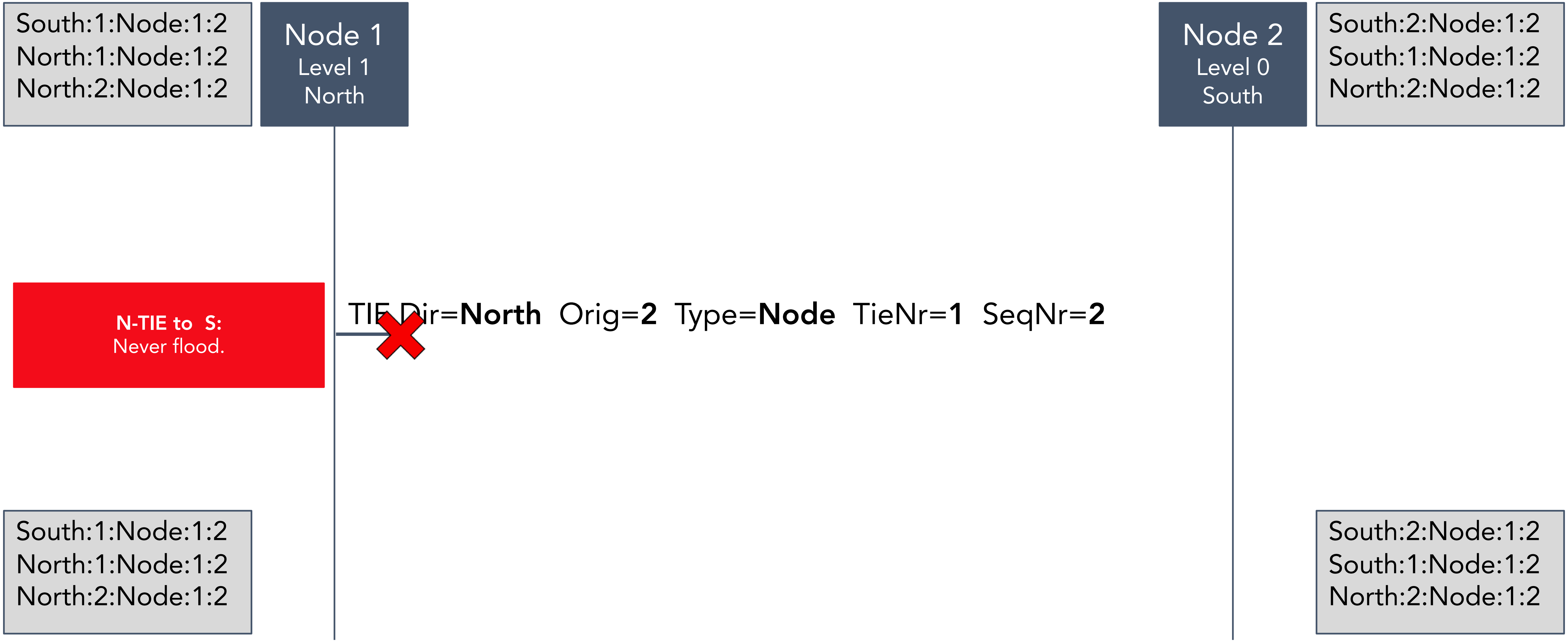
N-TIE to N:
Flood always

South:1:Node:1:2
North:1:Node:1:2
North:2:Node:1:2

South:2:Node:1:2
South:1:Node:1:2
North:2:Node:1:2

Node 1 does *not* reflect the TIE

Because the flooding scope rules don't allow it.



South:1:Node:1:2
North:1:Node:1:2
North:2:Node:1:2

Node 1
Level 1
North

Node 2
Level 0
South

South:2:Node:1:2
South:1:Node:1:2
North:2:Node:1:2

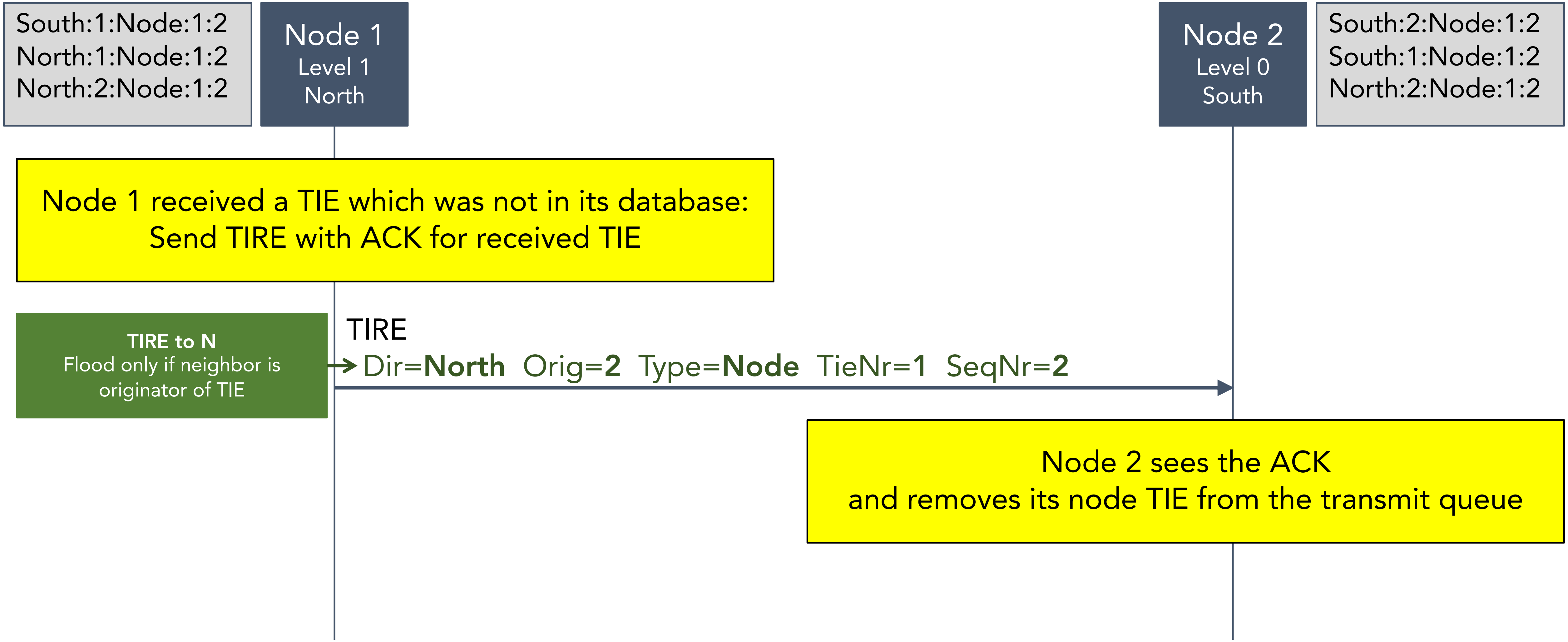
N-TIE to S:
Never flood.

TIE Dir=**North** Orig=**2** Type=**Node** TieNr=**1** SeqNr=**2**

South:1:Node:1:2
North:1:Node:1:2
North:2:Node:1:2

South:2:Node:1:2
South:1:Node:1:2
North:2:Node:1:2

TIRE from node 1 to node 2



South:1:Node:1:2
North:1:Node:1:2
North:2:Node:1:2

Node 1
Level 1
North

Node 2
Level 0
South

South:2:Node:1:2
South:1:Node:1:2
North:2:Node:1:2

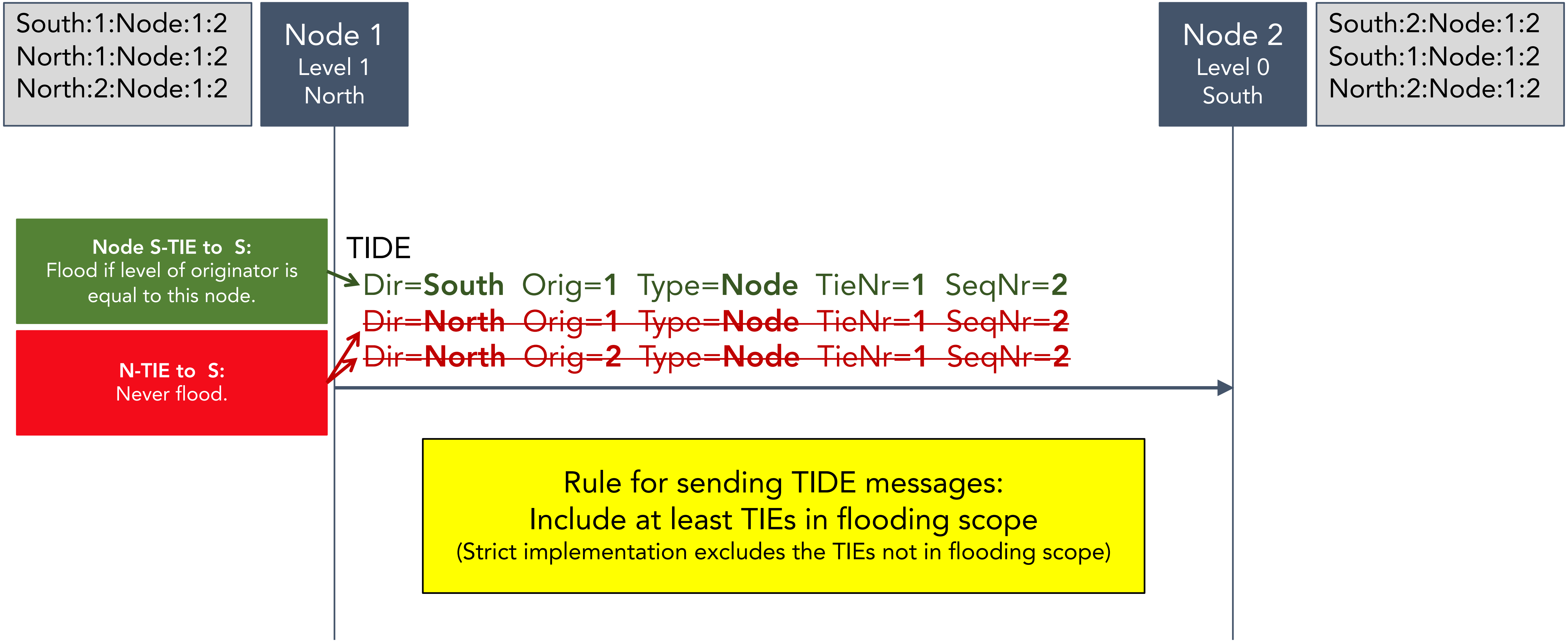
Node 1 received a TIE which was not in its database:
Send TIRE with ACK for received TIE

TIRE to N
Flood only if neighbor is
originator of TIE

TIRE
Dir=**North** Orig=**2** Type=**Node** TieNr=**1** SeqNr=**2**

Node 2 sees the ACK
and removes its node TIE from the transmit queue

TIDE from node 1 to node 2



South:1:Node:1:2
North:1:Node:1:2
North:2:Node:1:2

Node 1
Level 1
North

Node 2
Level 0
South

South:2:Node:1:2
South:1:Node:1:2
North:2:Node:1:2

Node S-TIE to S:
Flood if level of originator is equal to this node.

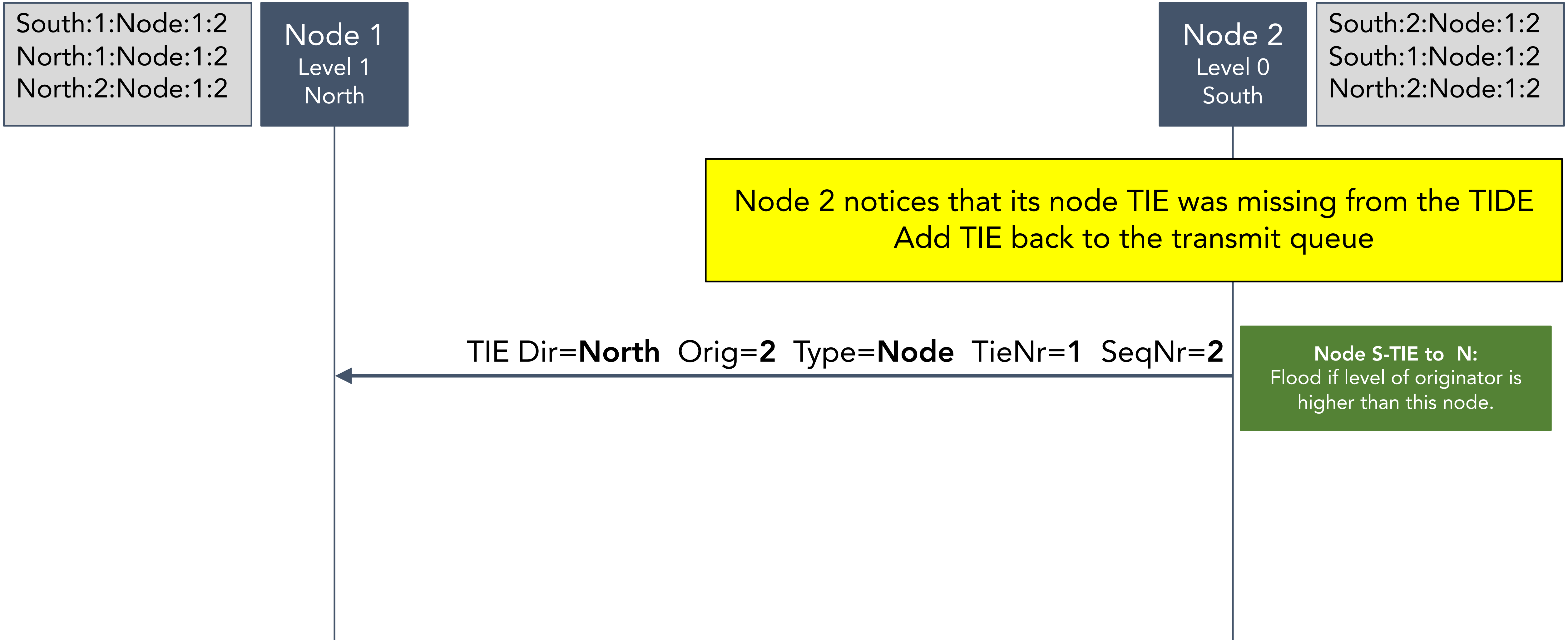
N-TIE to S:
Never flood.

TIDE
Dir=**South** Orig=**1** Type=**Node** TieNr=**1** SeqNr=**2**
~~Dir=**North** Orig=**1** Type=**Node** TieNr=**1** SeqNr=**2**~~
~~Dir=**North** Orig=**2** Type=**Node** TieNr=**1** SeqNr=**2**~~

Rule for sending TIDE messages:
Include at least TIEs in flooding scope
(Strict implementation excludes the TIEs not in flooding scope)

Node 2 resends its node TIE

Because it was missing in the TIDE received from node 1



South:1:Node:1:2
North:1:Node:1:2
North:2:Node:1:2

Node 1
Level 1
North

Node 2
Level 0
South

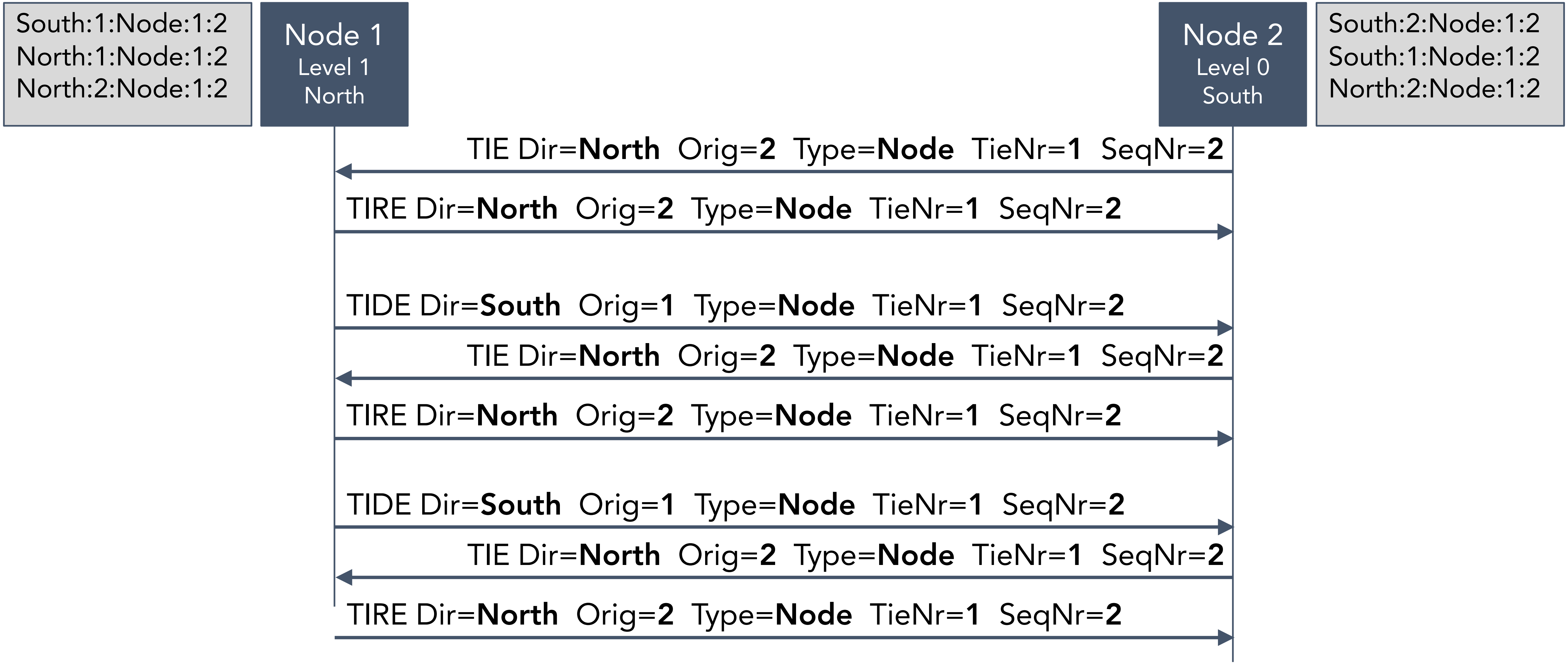
South:2:Node:1:2
South:1:Node:1:2
North:2:Node:1:2

Node 2 notices that its node TIE was missing from the TIDE
Add TIE back to the transmit queue

TIE Dir=**North** Orig=**2** Type=**Node** TieNr=**1** SeqNr=**2**

Node S-TIE to N:
Flood if level of originator is higher than this node.

Oscillation



South:1:Node:1:2
North:1:Node:1:2
North:2:Node:1:2

Node 1
Level 1
North

Node 2
Level 0
South

South:2:Node:1:2
South:1:Node:1:2
North:2:Node:1:2

TIE Dir=**North** Orig=**2** Type=**Node** TieNr=**1** SeqNr=**2**

TIRE Dir=**North** Orig=**2** Type=**Node** TieNr=**1** SeqNr=**2**

TIDE Dir=**South** Orig=**1** Type=**Node** TieNr=**1** SeqNr=**2**

TIE Dir=**North** Orig=**2** Type=**Node** TieNr=**1** SeqNr=**2**

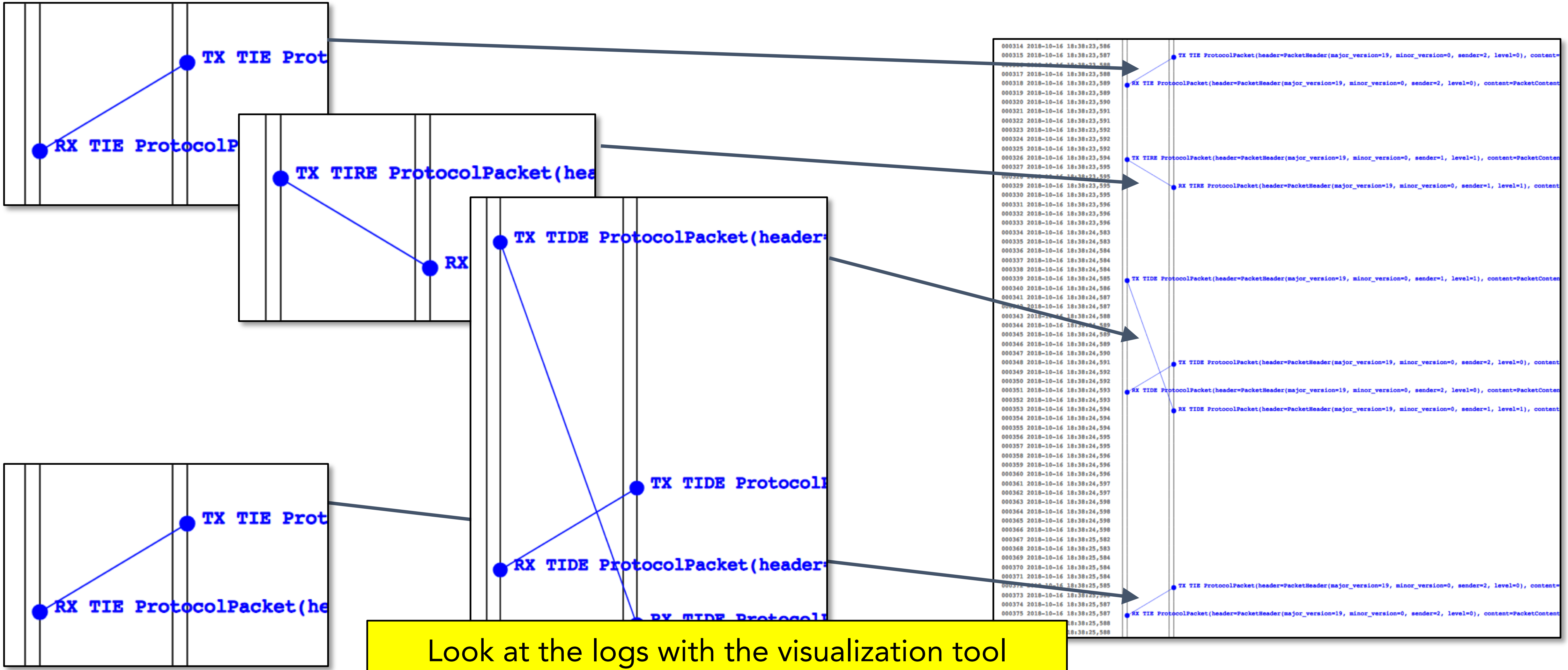
TIRE Dir=**North** Orig=**2** Type=**Node** TieNr=**1** SeqNr=**2**

TIDE Dir=**South** Orig=**1** Type=**Node** TieNr=**1** SeqNr=**2**

TIE Dir=**North** Orig=**2** Type=**Node** TieNr=**1** SeqNr=**2**

TIRE Dir=**North** Orig=**2** Type=**Node** TieNr=**1** SeqNr=**2**

What it looks like in “real life”



```
TX TIE Prot
RX TIE ProtocolP
```

```
TX TIRE ProtocolPacket(hea
RX
```

```
TX TIDE ProtocolPacket(header=
TX TIDE Protocol
RX TIDE ProtocolPacket(header=
RX TIDE Protocol
```

```
TX TIE Prot
RX TIE ProtocolPacket(he
```

```
000314 2018-10-16 18:38:23,586
000315 2018-10-16 18:38:23,587          TX TIE ProtocolPacket(header=PacketHeader(major_version=19, minor_version=0, sender=2, level=0), content=
000316 2018-10-16 18:38:23,588
000317 2018-10-16 18:38:23,588
000318 2018-10-16 18:38:23,589   RX TIE ProtocolPacket(header=PacketHeader(major_version=19, minor_version=0, sender=2, level=0), content=PacketContent
000319 2018-10-16 18:38:23,589
000320 2018-10-16 18:38:23,590
000321 2018-10-16 18:38:23,591
000322 2018-10-16 18:38:23,591
000323 2018-10-16 18:38:23,592
000324 2018-10-16 18:38:23,592
000325 2018-10-16 18:38:23,592
000326 2018-10-16 18:38:23,594   TX TIRE ProtocolPacket(header=PacketHeader(major_version=19, minor_version=0, sender=1, level=1), content=PacketConten
000327 2018-10-16 18:38:23,595
000328 2018-10-16 18:38:23,595
000329 2018-10-16 18:38:23,595          RX TIRE ProtocolPacket(header=PacketHeader(major_version=19, minor_version=0, sender=1, level=1), content
000330 2018-10-16 18:38:23,595
000331 2018-10-16 18:38:23,596
000332 2018-10-16 18:38:23,596
000333 2018-10-16 18:38:23,596
000334 2018-10-16 18:38:24,583
000335 2018-10-16 18:38:24,583
000336 2018-10-16 18:38:24,584
000337 2018-10-16 18:38:24,584
000338 2018-10-16 18:38:24,584
000339 2018-10-16 18:38:24,585   TX TIDE ProtocolPacket(header=PacketHeader(major_version=19, minor_version=0, sender=1, level=1), content=PacketConten
000340 2018-10-16 18:38:24,586
000341 2018-10-16 18:38:24,587
000342 2018-10-16 18:38:24,587
000343 2018-10-16 18:38:24,588
000344 2018-10-16 18:38:24,589
000345 2018-10-16 18:38:24,589
000346 2018-10-16 18:38:24,589
000347 2018-10-16 18:38:24,590
000348 2018-10-16 18:38:24,591          TX TIDE ProtocolPacket(header=PacketHeader(major_version=19, minor_version=0, sender=2, level=0), content
000349 2018-10-16 18:38:24,592
000350 2018-10-16 18:38:24,592
000351 2018-10-16 18:38:24,593   RX TIDE ProtocolPacket(header=PacketHeader(major_version=19, minor_version=0, sender=2, level=0), content=PacketConten
000352 2018-10-16 18:38:24,593
000353 2018-10-16 18:38:24,594          RX TIDE ProtocolPacket(header=PacketHeader(major_version=19, minor_version=0, sender=1, level=1), content
000354 2018-10-16 18:38:24,594
000355 2018-10-16 18:38:24,594
000356 2018-10-16 18:38:24,595
000357 2018-10-16 18:38:24,595
000358 2018-10-16 18:38:24,596
000359 2018-10-16 18:38:24,596
000360 2018-10-16 18:38:24,596
000361 2018-10-16 18:38:24,597
000362 2018-10-16 18:38:24,597
000363 2018-10-16 18:38:24,598
000364 2018-10-16 18:38:24,598
000365 2018-10-16 18:38:24,598
000366 2018-10-16 18:38:24,598
000367 2018-10-16 18:38:25,582
000368 2018-10-16 18:38:25,583
000369 2018-10-16 18:38:25,584
000370 2018-10-16 18:38:25,584
000371 2018-10-16 18:38:25,584
000372 2018-10-16 18:38:25,585          TX TIE ProtocolPacket(header=PacketHeader(major_version=19, minor_version=0, sender=2, level=0), content=
000373 2018-10-16 18:38:25,586
000374 2018-10-16 18:38:25,587
000375 2018-10-16 18:38:25,587   RX TIE ProtocolPacket(header=PacketHeader(major_version=19, minor_version=0, sender=2, level=0), content=PacketContent
                  18:38:25,588
                  18:38:25,588
```

Look at the logs with the visualization tool

In a stable topology I should not see any TIEs or TIREs

It is easy to see oscillation patterns in the visualization

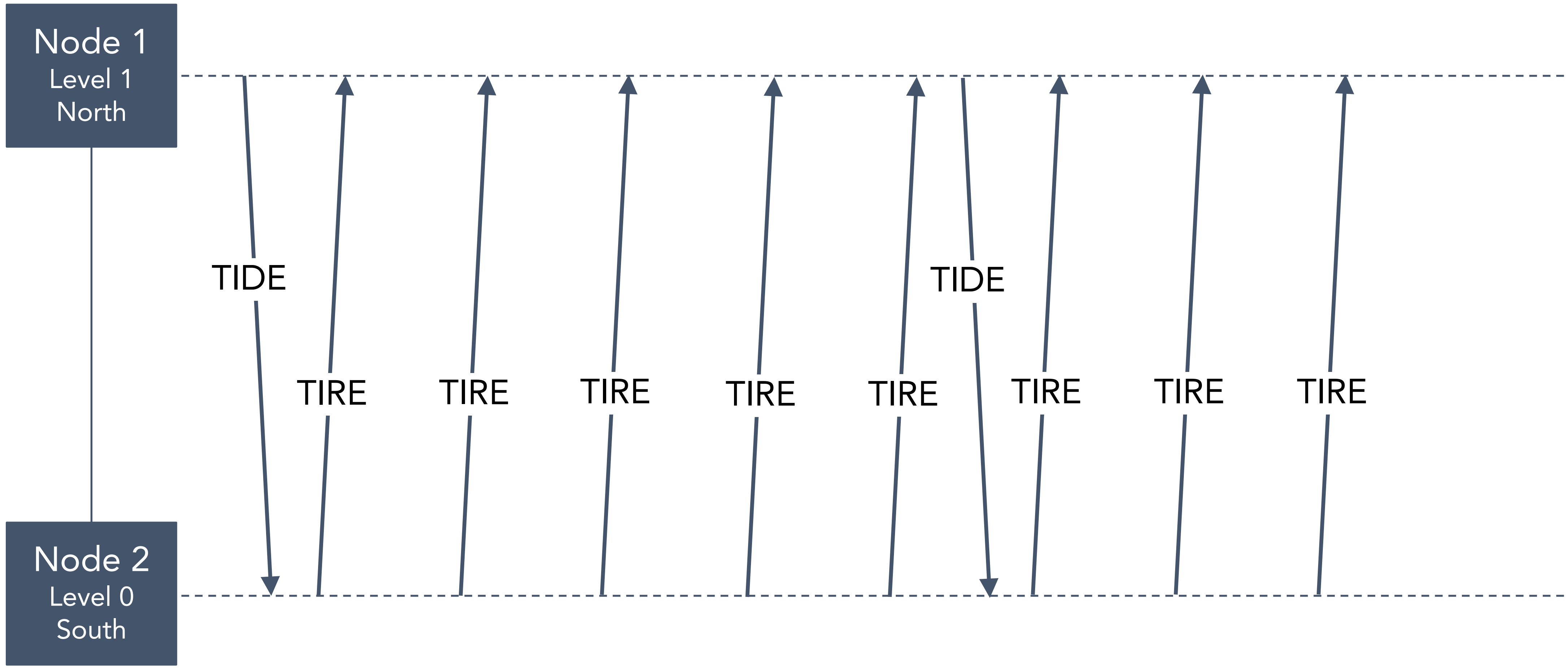
Root cause of the oscillation #1

- A TIDE message sent from node X to node Y serves two purposes:
- Purpose 1: Announce TIEs that X wants to flood to Y
  - TIDE **MUST** contain the TIEs that X **MUST** flood to Y
  - TIDE **MAY** contain additional TIEs that X **MUST NOT** flood to Y
  - To avoid having to encode a separate TIDE for each neighbor
  - Y must apply flooding scope rules, and ignore ("not accept") extra TIEs in TIDE
- Purpose 2: Acknowledge acceptance of TIEs that Y has flooded to X
  - If Y has a TIE that it must flood to X and it is missing in the TIDE received from X, then Y will (re)send the TIE to X
  - The TIDE **MUST** contain all TIEs that X has received and accepted from Y (even if X has not intention of sending the TIE to Y, i.e. even if not in flooding scope)
- The current RIFT specification only captures the first purpose

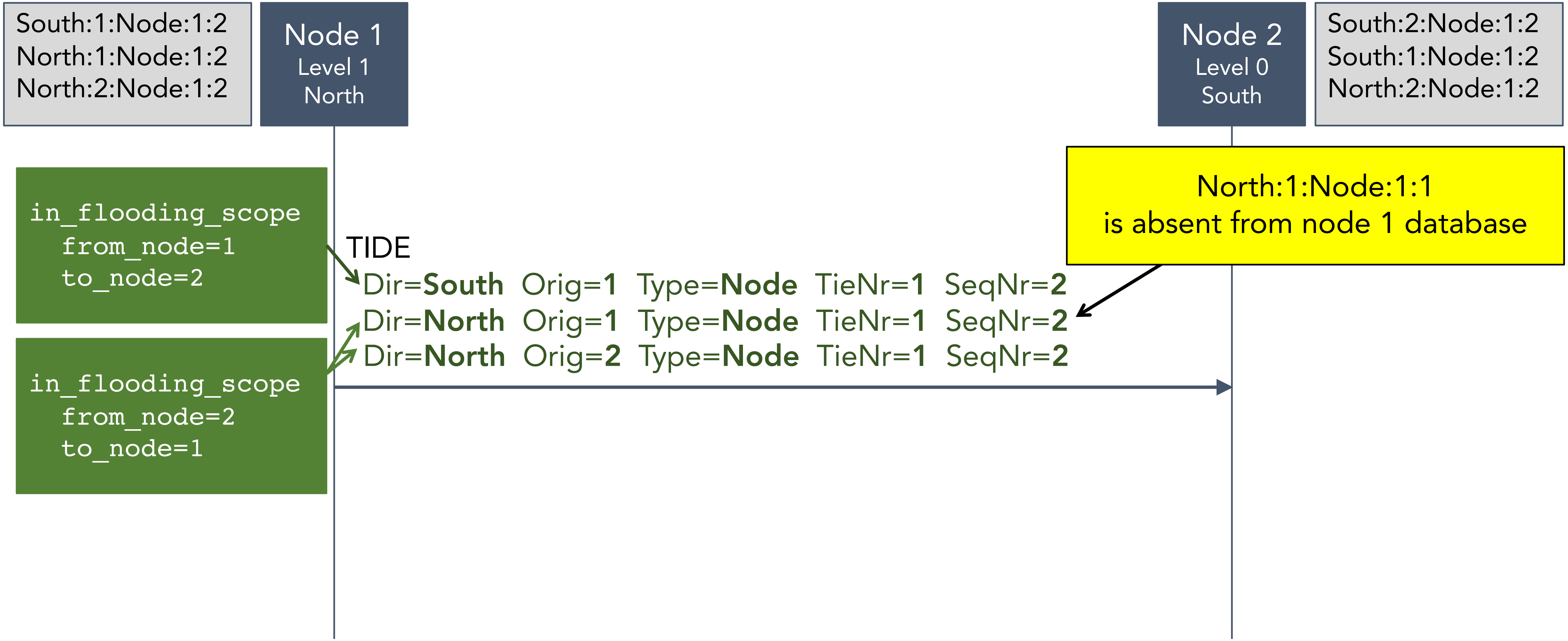
Short-term / ad-hoc solution for oscillation #1

- Update rule for sending TIDEs
- A TIDE message sent from node X to node Y:
  - Purpose 1: MUST include TIEs for which:
    ```
    in_flood_scope(from_node=X, to_node=Y, tie_header=T) == true
    ```
  - Purpose 2: MUST include TIEs for which:
    ```
    in_flood_scope(from_node=Y, to_node=X, tie_header=T) == true
    ```
  - MAY include additional TIEs

Oscillation #2 (after fixing oscillation #1)



TIDE from node 1 to node 2



South:1:Node:1:2
North:1:Node:1:2
North:2:Node:1:2

Node 1
Level 1
North

Node 2
Level 0
South

South:2:Node:1:2
South:1:Node:1:2
North:2:Node:1:2

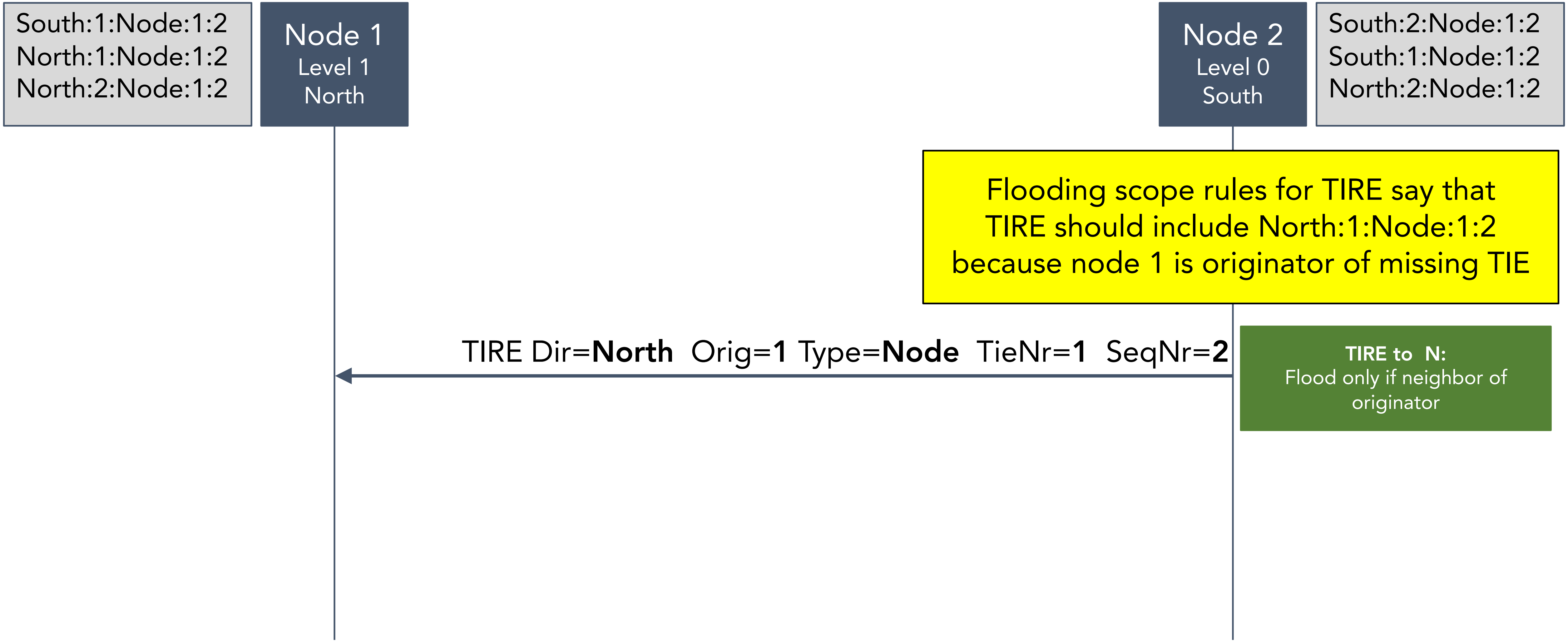
North:1:Node:1:1
is absent from node 1 database

```
in_flooding_scope
  from_node=1
  to_node=2
```

```
in_flooding_scope
  from_node=2
  to_node=1
```

TIDE
Dir=**South** Orig=**1** Type=**Node** TieNr=**1** SeqNr=**2**
Dir=**North** Orig=**1** Type=**Node** TieNr=**1** SeqNr=**2**
Dir=**North** Orig=**2** Type=**Node** TieNr=**1** SeqNr=**2**

Node 2 requests the missing TIE



South:1:Node:1:2
North:1:Node:1:2
North:2:Node:1:2

Node 1
Level 1
North

Node 2
Level 0
South

South:2:Node:1:2
South:1:Node:1:2
North:2:Node:1:2

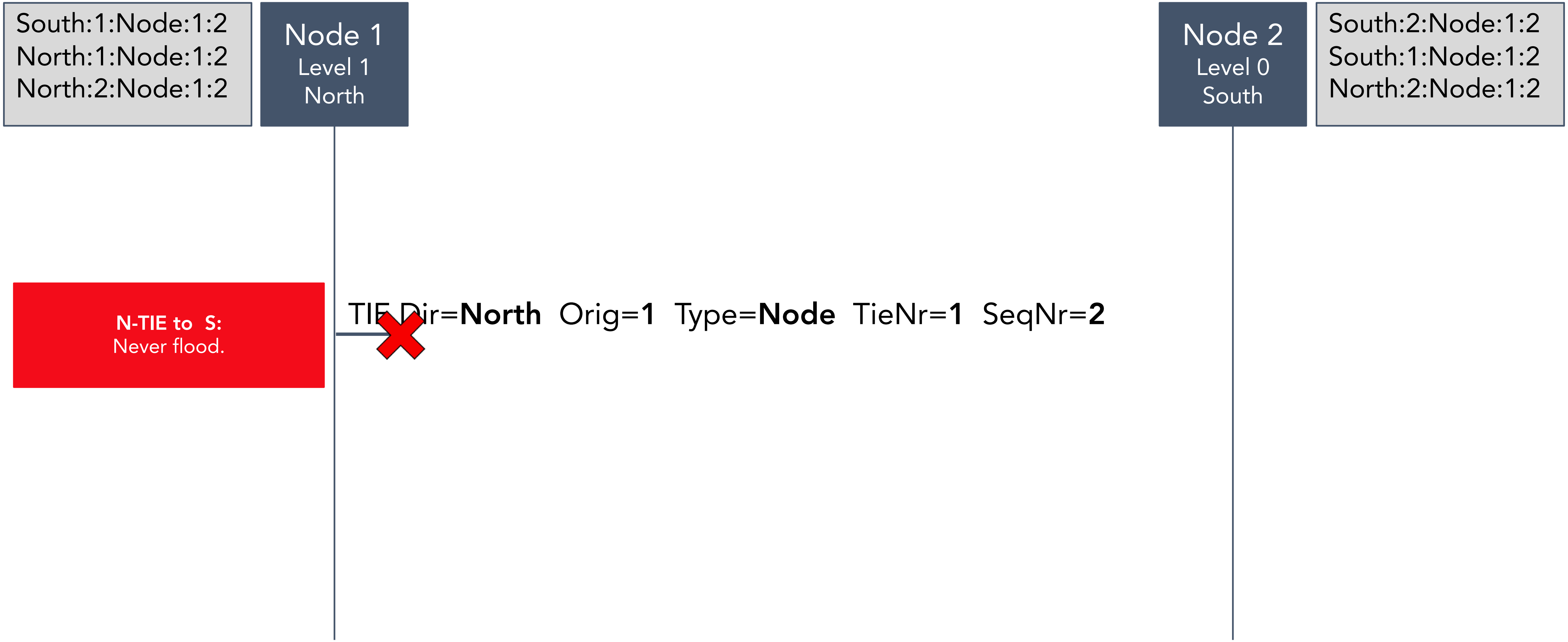
Flooding scope rules for TIRE say that TIRE should include North:1:Node:1:2 because node 1 is originator of missing TIE

TIRE Dir=**North** Orig=**1** Type=**Node** TieNr=**1** SeqNr=**2**

TIRE to N:
Flood only if neighbor of originator

Node 1 does *not* send the requested TIE

Because the flooding scope rules don't allow it.



South:1:Node:1:2
North:1:Node:1:2
North:2:Node:1:2

Node 1
Level 1
North

Node 2
Level 0
South

South:2:Node:1:2
South:1:Node:1:2
North:2:Node:1:2

N-TIE to S:
Never flood.

TIE Dir=**North** Orig=**1** Type=**Node** TieNr=**1** SeqNr=**2**

Analysis of oscillation #2

Root cause of oscillation #2

- A TIRE message sent from node X to node Y serves two purposes:
- Purpose 1: X is requesting a missing TIEs it wants Y to send
- Purpose 2: X is acknowledging acceptance of TIEs it has received from Y
- The current TIRE flooding rule only captures the second purpose

Potential short-term / ad-hoc solution:

- Different TIRE rules for request missing / acknowledge
- Not (yet) implemented – want to step back and consider more drastic measures
- Note: so far we have only considered a trivial 2-node topology and not even looked at more complex topologies

A game of “whack-a-mole”

1. Find an oscillation scenario
2. Tweak the flooding scope rules to fix it.
3. Find a new oscillation scenario which is a result of the tweaked rules.
4. Go to step 2.

System behavior (oscillations) extremely sensitive to rule details

Porposed long-term / fundamental solution

- Basic idea: encode target flooding scope into TIE header, e.g.:
  - Flood to “node 4 and direct south neighbors”
  - Flood to “node 18 and south-cone from there”
  - Flood to “level 0 and all north levels”
  - Flood to “level 2 and direct south level”
- Just a few bytes in the TIE header (from-where, direction, how-far)
- Advantages:
  - Explicitly signal intent, instead of trying to reverse-engineer intent from rules
  - I expect this to be simpler to implement and to understand behavior
  - Originator can control scope for individual TIEs (e.g. different keys in KV)
- **More detailed proposal and analysis to follow**