I think the frame of promises

may be helpful to analyze what kinds of properties we want distributed controller operation and state synchronization to have. Specifically, when I talk about promises in this context, I mean promises by

a controller to accept

inclusion of certain future histories under certain conditions. Controllers always have control over what histories they accept - what transactions they sign over - so these are promises which they can make, and promises on which users will at least implicitly rely in order to make decisions about how to move their resources around the system - and I think it may help to make them explicit

.

I think it's also helpful to keep in mind that resources in the Anoma system will always be tied to resources in the real world by a single originating external identity, who is - at least implicitly - promising to honor the digital representation (for example, allowing redemption for a physical good). When users are using resources "downstream" of that originating external identity (possibly also a controller), they ultimately care whether or not the resources which they're using can be redeemed back "upstream", since that upstream path is what connects them to the real world. Accordingly, users of downstream resources care about the existence of a promise chain

which they could rely on to redeem the resource if desired. Typically, this redemption should not actually have to be enacted - but it must be clearly possible

.

Let's assume that this originating identity enforces, at minimum, local linearity

: no more promises can be redeemed than were originally created. The first question, then, is - considering that there may be potentially conflicting histories, which one is accepted?

As we have only local time, if we want to eventually accept a history, all of our options look like "FIFO + x":

- FIFO only - as soon as a valid history is provided, accept it, and reject all future conflicting histories (this is what IBC does, with the specific case of fungible tokens).
- FIFO + delay - as soon as a valid history is provided, start a timer. When the delay period has passed, if no other valid histories were provided, accept the history. If another valid history is submitted during the delay period, run a resolution protocol, which may rely on other controllers, further delay periods, etc.
- FIFO + delay + endorsements - same as the delay period, but endorsements from various controllers can shorten the delay period (as they provide additional evidence that a particular history is canonical in other parts of the network).

I think these are all the possible options.

No more time for writing today, to be continued…