

MoQ stream mapping proposal

vasilvv@google.com
MoQ interim, 2023-10-05

Observation

Stream mapping is inherently related to dependency relationships between objects

Whenever we put object B after object A on the same stream, we effectively operate as if B depends on A

The converse is not true: we can put objects that depend on each other on different streams, in case we want to increase granularity at which we can prioritize or drop things

Principles

- In general, give actionable instructions to relays (“place object here”, not “this depends on this”)
- Some extra metadata that could be used for dropping is possible
- The goal is to not have a full dependency description, but give a simple generalized instructions for the relays

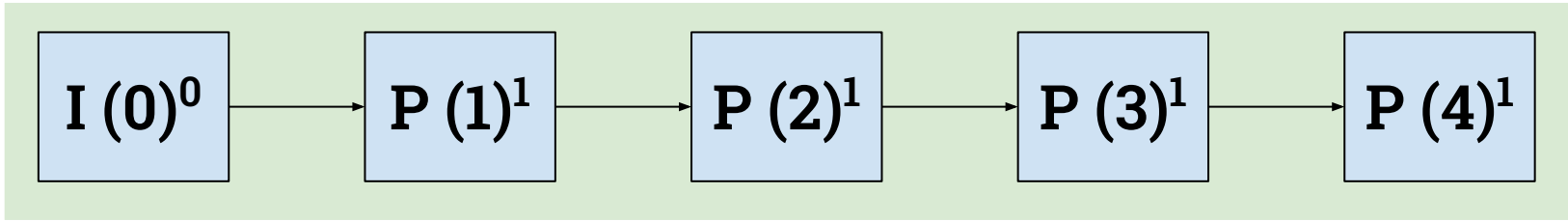
The simplest proposal

We add a new number to the object, called “stream placement marker”, that indicates what stream the object should be placed on.

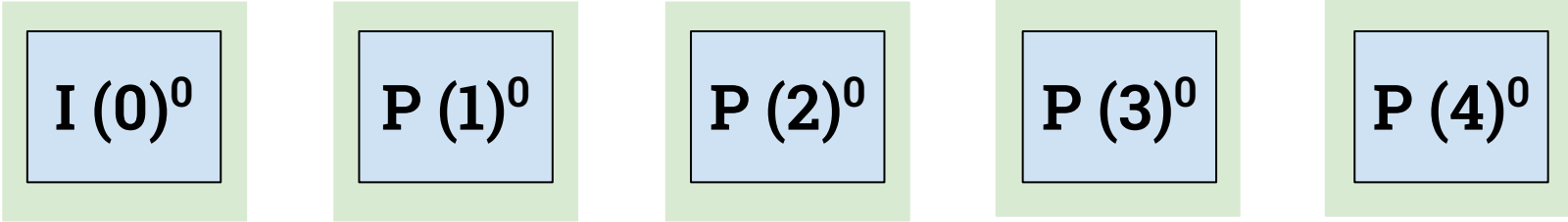
The simplest version is just one bit:

- “0” means “place this object on a new stream”
- “1” means “place this object on the same stream as the previous object” (i.e. if the object has sequence number N , that would be the same stream as $N-1$)

Implies that “1” objects depend on previous objects, and “0” objects do not.



Stream per group



Stream per object

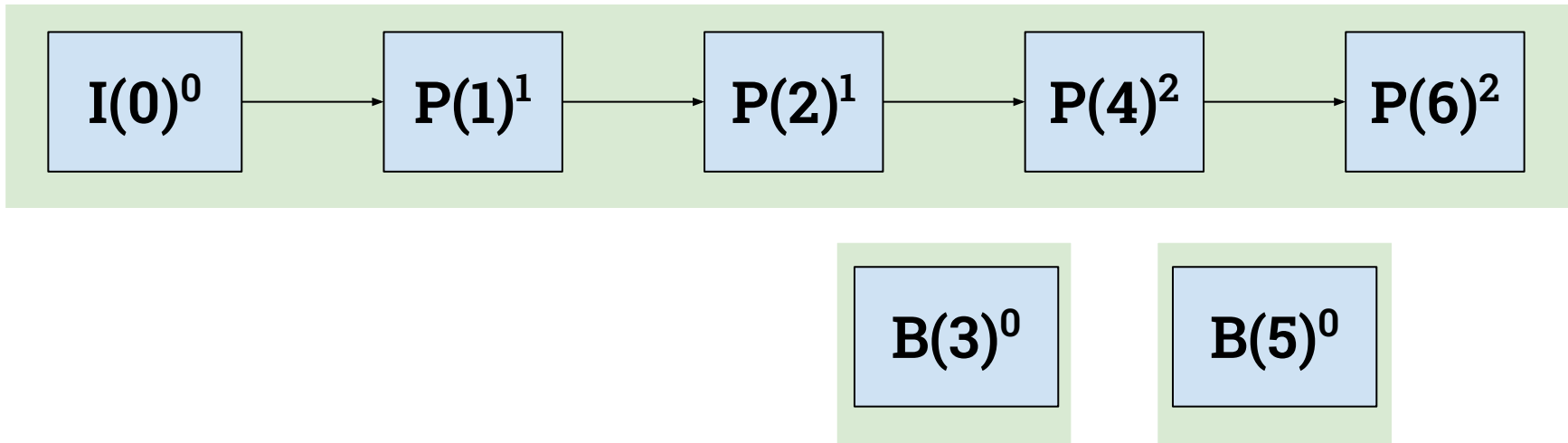
More elaborate use case

Assume that we want to do “stream per group”, but make some of the frames inside the said group optional

Proposal, first half: extend stream placement marker to accommodate for those.
For object N:

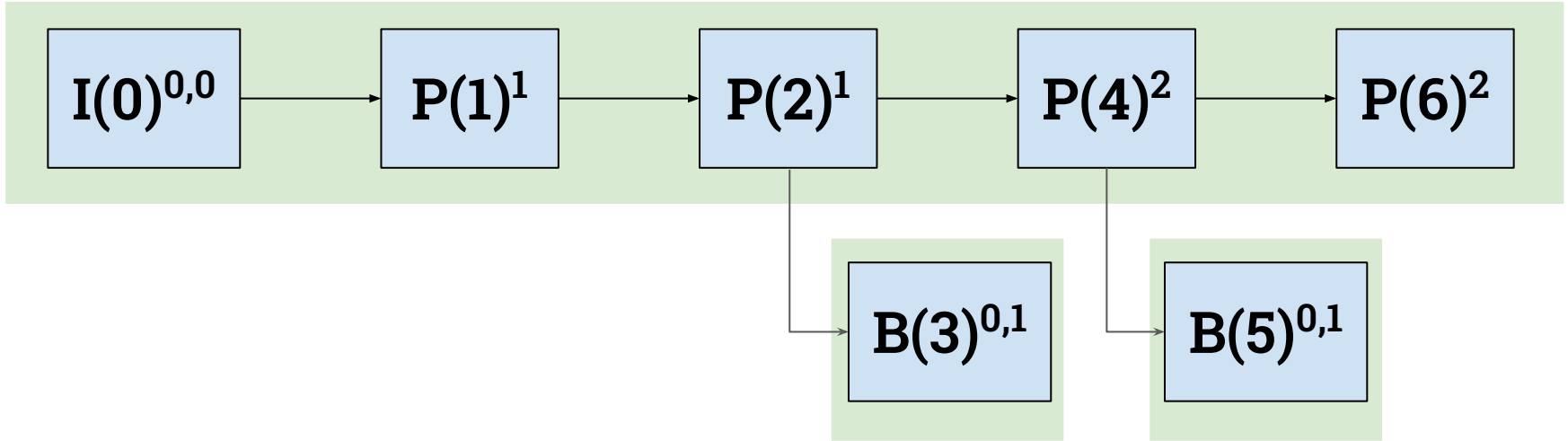
- “0” means “place this object on a new stream”
- “1” means “place this object on the same stream as N-1”
- “d” means “place this object on the same stream as N-d”

Original sequence: I (0) P (1) P (2) B (3) P (4) B (5) P (6)



Problem: we don't indicate that B (3) and B (5) depend on certain P-frames!

Original sequence: I (0) P (1) P (2) B (3) P (4) B (5) P (6)



Solution: if placement marker is “0”, add another number for “depends on”

Proposal summary

Every object has a number (“stream placement marker”) that indicates what stream it should go on

If the marker says that the object should go on a new stream, an additional marker is added to indicate the object that the current object depends on

What to do when objects arrive on a wrong stream?

Options:

- Ignore that (relay reorders things if declared incorrectly)
- Enforce it. Relatively straightforward:
 - If the marker is “0”, it has to be the first object on the stream
 - If the marker is “d”, check that the previous object on the stream is “N-d”

When to send FIN on a stream?

Options/proposals:

- At the end of the group
- When $d > 1$, close the streams for objects that are being “skipped” over

The rationale behind the second principle is as follows: if we’re branching off some objects onto a separate stream, they’re assumed to be “optional”, and thus we want to send them at a higher granularity.

Other concerns

- Subscribe in the middle of the group:
 - There is no information on what was sent before
 - Solution: assume that if client subscribes at object N in group, they have objects 0..N
- The exact strategy for priorities and dropping
 - Out of scope for this presentation