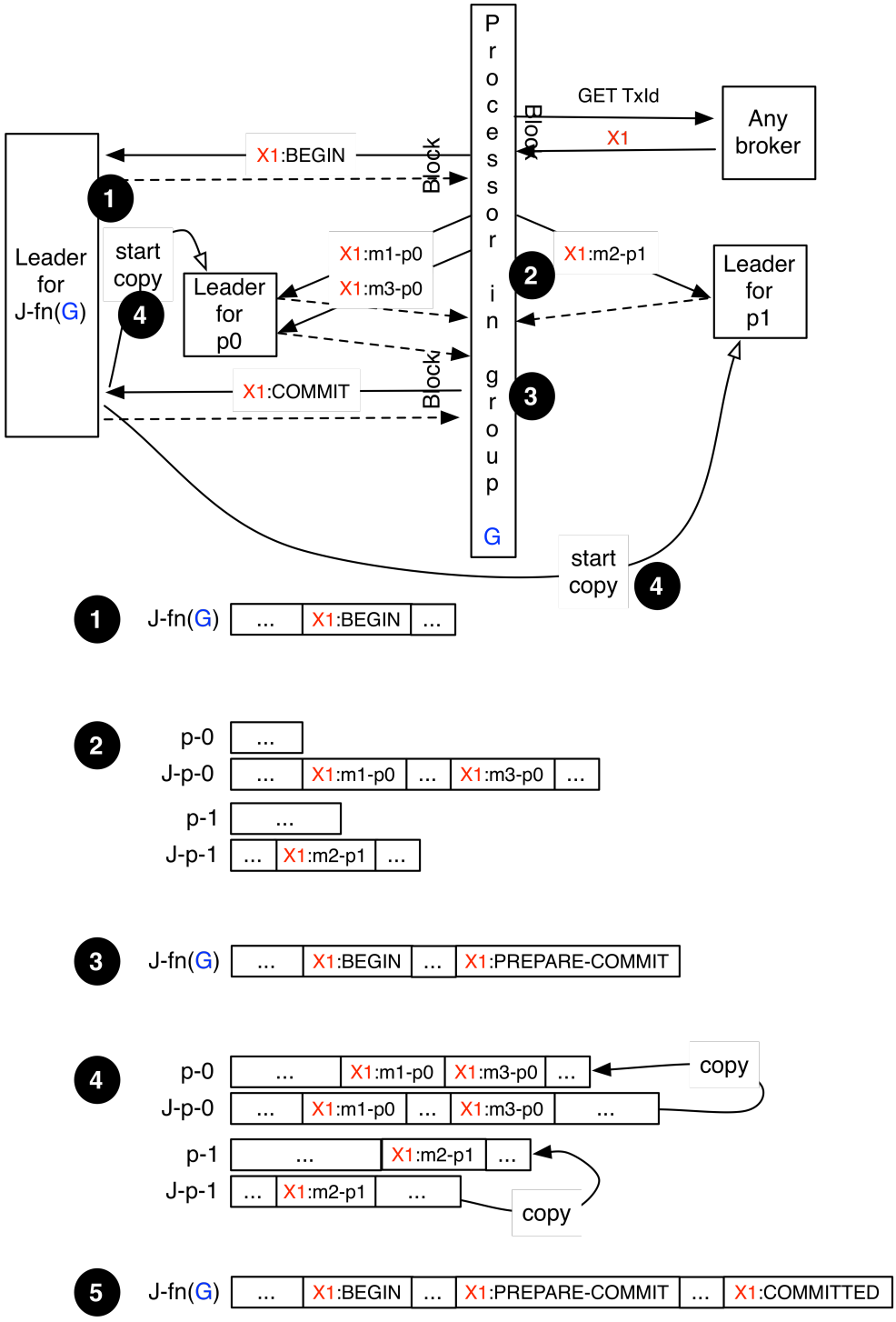
Double journaling with local data copy

In this approach, producers send metadata about each transaction to a special (replicated) journal log. Each partition of the actual data log also has a corresponding local data journal log. After the producer commits the transaction the brokers copy the data from the local data journal log (locally) into the actual data log.



Phase 1 (journaling)

- If the producer needs strict ordering then it should form a transactional group (G) with the other producers that may participate in the transaction. This could be a configured property. Otherwise it can use a random group G.Obtain a transaction ID (from any broker). These transaction IDs may be generated off a ZooKeeper queue. This is a blocking call and we could return more than one transaction ID that the producer can use in subsequent transactions, but we will anyway need to block in the commit (further down).
- Send BEGIN-TxId to the journal partition for its group based on some hash fn - i.e., J-fn(G).
- Send (pipelined) messages for the transaction to the respective leaders of the journal partitions that correspond to the data partitions that comprise the transaction.
- Send COMMIT-TxId to the J-fn(G). A successful (non-error) response to this COMMIT message indicates to the producer that the transaction will be committed.

Phase 2 (local copying)

- When the coordinator (i.e., leader of J-fn(G)) receives a COMMIT, it appends a PREPARE_COMMIT(TxId) message to J-fn(G).
- After the above append has been replicated to the follower it can ack the producer's COMMIT.
- The coordinator then sends a APPLY_COMMIT(TxId) message to all the leaders of the partitions that are in the transaction to copy the transaction data into the actual data logs.
- When a broker receives the APPLY_COMMIT(TxId) message, it does an idempotent copy of all the messages in transaction TxId into the actual data log. The mechanism of traversing over the data journal log to collect messages from the transaction will be similar to the methods described in the other double-journaling proposal. (i.e., either buffer or maintain a reverse linked-list.)

Pro

- The copying is local so throughput should be higher than the double journaling approach with remote copy.
- It seems possible to implement this without full-fledged idempotence.
- Vanilla consumers are unaffected.

Con

- Seems a little more complex than the double-journaling approach but may (in reality) be equivalent in implementation effort.
- There is still some latency from the producer ack to the point the transaction can be exposed to the consumer but it is less than the other other double-journaling approach.