Sub-Synchronizing Shared Session Types in Nomos

Ishani Santurkar; Advisors: Ankush Das and Jan Hoffmann

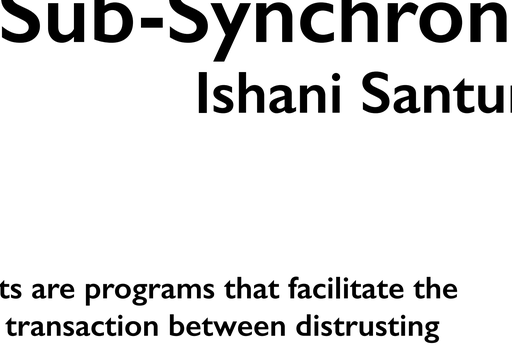
Nomos

- Smart contracts are programs that facilitate the execution of a transaction between distrusting parties.
- Nomos is a smart contract programming language that:
  - Statically guarantees protocol adherence using session types.
  - Automatically infers gas (execution cost) bounds.
  - Enforces linearity to prevent duplication of assets like money.

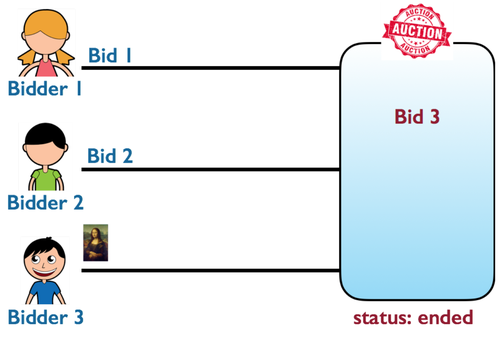
Auction Contract

- An auction contract runs in two phases:

Phase 1
RUNNING - users send bids to contract

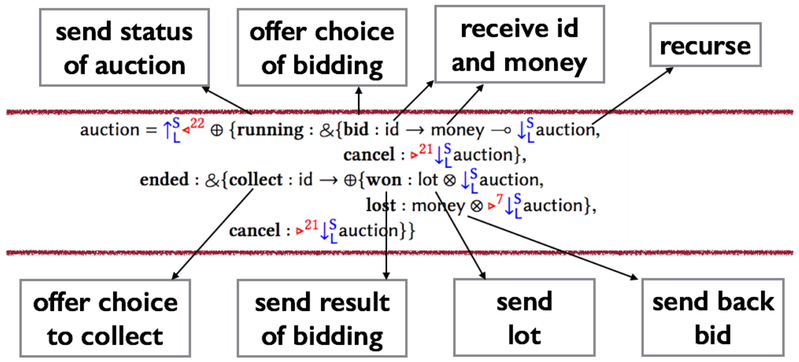


Phase 2
ENDED - winner collects lot, while others receive bids back



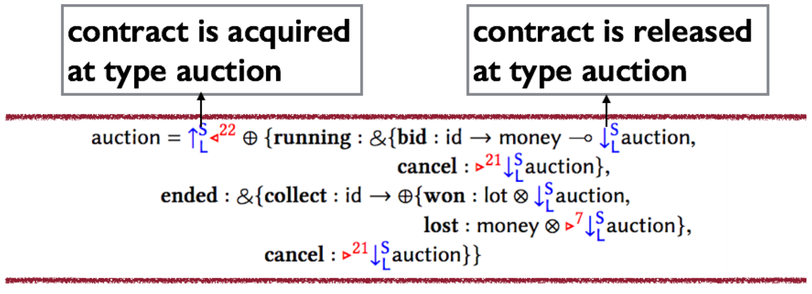
Auction Type in Nomos

- The auction process communicates with clients according to the **auction** session type below.



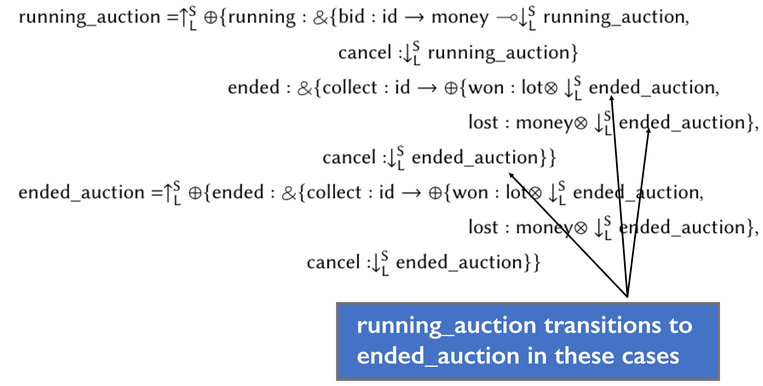
Equi-Synchronizing Session Types

- An auction is a shared session type – it offers a service to multiple clients.
- To ensure mutual exclusion, a client must:
  - acquire the shared channel
  - communicate with it in private
  - release the shared channel
- Equi-synchronizing requirement – user must release contract at the same type as it was acquired at.



Sub-Synchronizing Session Types

- Allows shared channel to be released at a subtype of its original type.
- This can be used to signal phases of an auction from its type.
- For example, on completing its running phase, a **running_auction** can transition to its subtype **ended_auction** which does not accept new bids.



Contributions

In this project, I:

- Developed rules for the algorithmic subtyping of Nomos, increasing the flexibility of its type system and allowing it to express a wider range of programs.
- Implemented the subtyping and sub-synchronizing algorithms in the Nomos type-checker.
- Proved type-safety of the extended system.

Future Work

- Extending subtyping to functional constructs in Nomos.
- Adding polymorphism to Nomos.
- Investigating the relationship between polymorphism and subtyping.