Intent Machines (in 15 min)

Christopher Goes
Inaugural Intents Day!

Goals of an intent formalism

---

1. Capture commonalities of intent systems
2. Capture structure, not implementation detail
3. Aid in analysis of:
    a. Similarities and differences
    b. Conditions for and behavior of composition
    c. Relationship to other concepts (e.g. MEV)

Candidate formalism

Candidate: “Intent machine”

---

1. Fix a state type **T**.
2. An *intent* is a function of type **T -> T -> 0 | 1**.
3. An *intent machine* is a potentially non-deterministic function of type **(T, Set I) -> (T, Set I)**
   a. First tuple: prior state and candidate intents
   b. Second tuple: posterior state and processed intents
4. Key property: *intent adherence*
   a. forall i in processed . i prior posterior = 1

Intent machines: decomposition

Without loss of generality, this function can be decomposed into two steps:

1. *Enumeration*: computing a set of (candidate state, processed) tuples which satisfy intent adherence.
2. *Selection*: choosing one of the tuples to return.

Intent machines: constraints

This function may additionally constrain which state transitions are considered to be valid. This can be modelled as a “system intent” which must always be satisfied.

Examples:

- Interior EVM state transition function satisfied
- Resource linearity & logics satisfied

Intent machines: selection

Selection picks one pair from the set of valid options.

choose :: Set (T, Set I) -> (T, Set I)

All of the interesting structure lies here.

Selection functions

Example selection functions I

- “Pure chaos”
  - Select a valid return pair at random
- “Pareto-efficient chaos”
  - Select the return pair which satisfies the most intents; break ties with randomness.
- “Utility maximization”
  - Select the return pair which maximizes some scalar function **T -> Nat**.

Example selection functions II

- “Profit maximization”
  - Utility maximization with a utility function that calculates the balance of some specific token owned by the operator’s address.
- “Welfare maximization”
  - Utility maximization with a utility function set to the welfare function of some community.
- “Expected utility maximization”
  - Select the return pair which maximizes expected future utility, given some probability distribution over future intents conditional on the posterior state.

Composition: selection functions

- “Optimistic preferred”
  - If both selection functions agree, return that, else use the solution chosen by one of them.
- “Optimistic random”
  - If both selection functions agree, return that, else choose randomly between options the Pareto frontier.
- “Weighted welfare”
  - (works for scalar utility functions only)
  - Select according to some weighted sum.

Distribution

Analysis: distribution

- ➔ Anoma (& many others) effectively implement a distributed intent machine
  - Different parties performing enumeration and selection.
  - Consensus to agree on which new state will be chosen.
- ➔ Everything is distributed!
  - State
  - Computation
  - Enumeration
  - Selection
  - Verification

Analysis: composition under distribution

- ➔ This distributed intent machine is composed of … other intent machines (with different select functions)
  - e.g. profit-maximizing operators
- ➔ One could understand cryptoeconomic mechanism design as trying to set incentives to provide a particular composed selection function.
- ➔ MEV & co. enter here
- ➔ (needs more work)

Survey questions

Questions for the audience

---

1. What would your goals for an intent formalism be?
2. Do you think this option makes sense?
    a. Which parts are clear / unclear?
3. Are there other compelling candidate formalisms?

Thanks!