Computational (Higher and Lower) Type Theory

Robert Harper

MURI Meeting October 15, 2021
New and Coming PhD’s

Evan Cavallo defended February, 2021, post-doc at Stockholm w/Mörtberg.
   Higher Inductive Types and Parametricity in Cubical Type Theory.

Jon Sterling defended September, 2021, post-doc at Aarhus w/Birkedal.
   First Steps in Synthetic Tait Computability.

Yue Niu proposing soon.
   Cost-Aware Logical Framework
Two Themes

Synthetic Tait Computability

- Jon: Normalization for Cubical Type Theory.
- Yue: Cost-Aware Logical Framework.
- [Jon and B: Parametricity for Program Modules.]

Parametricity = Relational Invariance

- Carlo: Data Abstraction and Univalence.
- Evan: Higher Parametricity and Cohesion.
First Steps in STC: Normalization for Cartesian Cubical Type Theory.

- Normal cubes are **unstable** under substitution: eg compose to $x$, or endpoints.
- Universes require **structure**, not **property**.
- Synthetic separation: *open/closed modalities* $\mathcal{O}_{\text{syn}} A$, $\blacksquare_{\text{syn}} A$ corresponding to a subterminal syn.

Direct application to **CoolTT** implementation of CCTT.

- Practical decision method for judgmental equality.
- cf Favonia and Reed’s talks.
Second steps in STC: Rep Ind for Program Modules. [cf Crary]

- Program modules are **phase separated**:
  - **Static**: types = elements of universe.
  - **Dynamic**: programs as elements of types.

- Dependent types with $\mathcal{U}$, $\Sigma$, $\Pi$, plus
  - **Extension types**: $\{ A \mid \text{stat} \hookrightarrow V \}$. (cf Riehl and Shulman, Stone and H)
  - **Open/closed modalities**: $\mathcal{O}_{\text{stat}} A$, $\bullet_{\text{stat}} A$.

**Parametricity structures:**

- Internal language of topos constructed by glueing.
- Phase-separated, proof-relevant logical relations.
- Open/closed modalities: $\mathcal{O}_{\text{syn}} A$, $\bullet_{\text{syn}} A$. 

Phase distinctions abound!

- Syntax/semantics: computability structures / glueing.
- Static/dynamic: type checking vs execution time.
- Development/compilation: respecting/violating abstraction.
- Public/private: ensuring confidentiality/integrity.
- **Extension/intension**: behavior/resource usage.
Internal/intrinsic account of both behavior and cost.

- **Extensional:** \( \text{mergesort} = \text{inssort} : \text{seq} \rightarrow \text{seq} \).
- **Intensional:**
  - \( \text{mergesort} : \text{seq} \xrightarrow{n \lg n} \text{seq} \).
  - \( \text{inssort} : \text{seq} \xrightarrow{n^2} \text{seq} \).

Idea: use synthetic methods to isolate extension from intension.

- Open modality \( \mathcal{O}_{\text{ext}} A \): disregard cost.
- Closed modality \( \mathcal{C}_{\text{ext}} A \): isolate cost.

Notion of cost is abstract!

- Number of comparisons, number of hcom’s (potentially).
- Steps of execution: adequacy wrt operational semantics (cf. Niu and H 2021)
Reynolds: relational invariance / parametricity ensures representation independence.

- eg, relate two implementations of queues.
- show that operations preserve relation.
- Thm: no client can distinguish them.

Smells a lot like Structure Identity Principle!

- Use quotients via HIT’s to represent simulation relation as an equivalence.
- Apply univalence to ensure interchangeability in all contexts.
Cubical, proof-relevant parametricity expresses relational invariance:

- **Bridges** correspond to paths.
- **Relativity** corresponds to univalence.

Associativity of smash product is readily proved via parametricity.

- Relational invariance gives rise to paths.
- But it is a strong requirement (natural for computing though).

Cohesive modalities distinguish **pointwise** from **parametric**, and allow **transfer** of paths.


