A roadmap of process algebras sorted by expressiveness - Notes

Leon Lee

September 23, 2024

Contents

1 Objective

1.1 A roadmap of process algebras sorted by expressiveness

One process algebra (or any other type of language) is said to be as least as expressive as another if there exists a valid translation, or encoding, of the latter into the former. Such a valid encoding is required to be compositional, meaning that the translation of a composed expression is completely determined by the translations of the argument expressions and a translation of the composition mechanism. In addition, the meaning of the translation of an expression should be semantically equivalent to the meaning of the expression being translated. This requires a semantic equivalence that is meaningful for both the source and the target language.
Based on this, the many hundreds of process algebras or system specification languages proposed in the literature can be ordered by their relative expressiveness. One could make a graph with such languages as the nodes, and a directed edge between two nodes if one is more expressive then another. Or a category with the languages as objects and the valid encodings as morphisms. Work on this project consists of filling in parts of this roadmap. This involves mathematically proving or disproving that a given source language can be encoded in a given target language. As the whole envisioned roadmap is very large, multiple students could work on different parts of it.

1.2 Completion

Prove and/or disprove the existence of valid encodings from some relevant source language(s) to some relevant target language(s).

2 Process Algebrai

2.1 ACP

From MCS Lecture notes,

1. ϵ: **Successful termination** (only in the optional extension ACP_ϵ)
2. δ: **Deadlock**
3. a: **Action constant** for each action a
4. $P \cdot Q$: **Sequential Composition**
5. $P + Q$: **Summation**, **Choice**, or **Alternative Composition**
6. $P||Q$: **Parallel Composition**
7. $\partial_H(P)$: **Restriction**, or **Encapsulation** for each set of visible actions H
8. $\tau_I(P)$: **Abstraction** for each set of visible actions I (only in the optional extension ACT_τ)

2.2 CCS

From MCS Lecture notes,

1. 0: **Inaction**
2. $a \cdot P$: **Action prefix** for each action a
3. $P + Q$: **Summation**, **Choice**, or **Alternative Composition**
4. $P||Q$: **Parallel Composition**
5. $P|a$: **Restriction** for each action a
6. $P[f]$: **Relabelling** for each function $f : A \to A$

2.3 CSP

From MCS Lecture notes,

1. 0: **Inaction**, originally written **STOP**
2. $a \cdot P$: **Action prefix**, originally written $a \to P$ for each visible action a
3. $P \Box Q$: **External Choice**
4. $P \sqcap Q$: **Internal Choice**
5. $P||_S Q$: **Parallel Composition**, enforcing synchronisation over the set S of visible actions
6. P/a: **Concealment**, originally written $P|a$) for each action a
7. $P[f]$: **Renaming** for each function $f : A \to A$

2.4 LOTOS

2.5 π-Calculus

3 A Theory of Encodings and Expressions

3.1 Overview of Contributions

- **Congruence Transfer Property**: If $\mathcal{L}'$ is at least as expressive as $\mathcal{L}$ up to $\sim$ then $\approx$ is also a congruence for $\mathcal{L}$
- **Reverse Congruence**: If $\mathcal{I}$ is a translation of $\mathcal{L}$ into $\mathcal{L}'$ that is valid up to $\sim$, and $\approx$ is a congruence for $\mathcal{L}$ that is coarser than or equal to $\sim$, then $\approx$ is also a congruence for the fragment of $\mathcal{L}'$ that is obtained as the image of $\mathcal{I}$
- **Congruence Closure Property**: If a translation $\mathcal{I}$ between $\mathcal{L}$ and $\mathcal{L}'$ is calid up to a semantic equivalence $\approx$ then it is valid even up to an equivalence that
  - on $\mathcal{L}$ coincides with the congruence closure of $\approx$
  - on the image of $\mathcal{L}$ within $\mathcal{L}'$ also coincides with the congruence closure of $\approx$, and
  - melts each equivalence class of $\mathcal{L}$ with exactly one of $\mathcal{L}'$
- **Congruence reflection**: (under some mild side conditions), if a translation $\mathcal{I}$ between $\mathcal{L}$ and $\mathcal{L}'$ is valid up to $\approx$, and $\sim$ is a congruence for $\mathcal{L}'$ that is finer or equal to $\approx$, then $\mathcal{I}$ is even valid up to an extension of $\sim$ to the disjoint union of $\mathcal{L}$ and $\mathcal{L}'$, still finer than or equal to $\approx$, that also is a congruence for $\mathcal{L}$ - combines the features of two languages by translating them into a common target language.

3.2 Section 2

Definition 3.2.1: Language

- the **syntax** of a language determines the valid expressions in the language
- the **sementics** of a language is given by a mapping $[\![\,]\!]$ that associates with each valid expression the meaning, e.g. object, concept, or statement

Define a language $\mathcal{L}$ as a pair $(\mathbb{T}_{\mathcal{L}}, [\![\,]\!]_{\mathcal{L}})$ of a set $\mathbb{T}_{\mathcal{L}}$ of valid expressions in $\mathcal{L}$, and a mapping $[\![]\!]_{\mathcal{L}} : \mathbb{T}_{\mathcal{L}} \to \mathcal{D}_{\mathcal{L}}$ from $\mathbb{T}_{\mathcal{L}}$ in some set of meanings $\mathcal{D}_{\mathcal{L}}$

Definition 3.2.2: Single sorted Language

Single-sorted languages are ones where *expressions* or *terms* are built from variables (taken from an infinite set $\mathcal{X}$) by means of operators (including constants) and additional (e.g. recursion) constructs

---

Differentiates from other languages which are *multi-sorted* e.g.

- AWN - has sequential processes, parallel processes, network
- mCRL2 has multiple types of *data*

Definition 3.2.3: Translation

A **translation** from a language $\mathcal{L}$ into a language $\mathcal{L}'$ is a mapping $\mathcal{I} : \mathbb{T}_{\mathcal{L}} \to \mathcal{T}_{\mathcal{L}'}$

NOTE: *encoding* to be used as a synonym for translation

Definition 3.2.4: Expressiveness

A langauge $\mathcal{L}'$ is at least as **expressive** as $\mathcal{L}$ iff a valid translation from $\mathcal{L}$ to $\mathcal{L}'$ exists

This way *expressiveness results* can be obtained by exhibiting a valid transition between two languages and *separation results* by giving an argument that no valid translation between two given languages exists

Definition 3.2.5: Simple and Closed-term language

A language $\mathcal{L} = (\mathbb{T}_{\mathcal{L}}, [\![\,]\!]_{\mathcal{L}})$ is called simple if the following three conditions are met:

1. They have no process variables. So $\mathbb{T}_{\mathcal{L}} = T_{\mathcal{L}}$
2. They do not feature any additional syntactic constructs (beyond operators)
3. $\mathcal{D}_{\mathcal{L}} = T_{\mathcal{L}}$ and $[\![\,]\!]_{\mathcal{L}} : T_{\mathcal{L}} \to T_{\mathcal{L}}$ is the identity function

Languages satisfying (3) are called **closed term languages**. In such languages there is no separation between syntax and semantics

Definition 3.2.6: Compositional

Let $\mathcal{L}$ and $\mathcal{L}'$ be languages without variables or additional syntactic constructs (beyond operators). A translation $\mathcal{I}$ from $\mathcal{L}$ into a language $\mathcal{L}'$ is **compositional** if for every n-ary operator f of $\mathcal{L}$ there exists an n-ary $\mathcal{L}$'-context C_f such that $\mathcal{I}(f(E_1, \ldots, E_n)) = C_f[\mathcal{I}(E_1), \ldots, \mathcal{I}(E_n)]$

Theorem 3.2.7: Compositional

Let $\mathcal{L}$ and $\mathcal{L}'$ be languages without variables or additional syntactic constructs (beyond operators). A translation $\mathcal{I}$ from $\mathcal{L}$ into a language $\mathcal{L}'$ is **compositional** iff for every n-ary context D of $\mathcal{L}$ there exists an n-ary $\mathcal{L}$'-context C such that $\mathcal{I}(D(E_1, \ldots, E_n)) = C[\mathcal{I}(E_1), \ldots, \mathcal{I}(E_n)]$

Definition 3.2.8: Preorder operator

Define $\sim$ (this has the circle on it) on $T_{\mathcal{L}} \uplus T_{\mathcal{L}'}$. Intuitively, $P' \sim P$ with $P \in T_{\mathcal{L}}$ and $P' \in T_{\mathcal{L}'}$ means that the processes P and P' are sufficiently like for our purposes, so that one can accept a translation of P into P'

A translation $\mathcal{I}$ is **valid up to** $\sim$ iff it is compositional and $\mathcal{I}(P) \sim P$ for each $P \in T_{\mathcal{L}}$

Definition 3.2.9: Finer

An equivalence or preorder $\sim$ on a class Z is said to be **finer**, **stronger**, or **more discriminating** than another equivalence or preorder $\approx$ on Z if $v \sim w \implies v \approx w$ for all $v, w \in Z$

Using a finer equivalence or preorder yields a stronger claim that one language can be encoded in another. On the other hand, when separating two languages $\mathcal{L}$ and L' by showing that $\mathcal{L}$ **cannot** be encoded in $\mathcal{L}'$, a coarser equivalence or preorder yields a stronger claim

Theorem 3.2.10: Associativity property

If valid translatiosn up to $\sim$ exists from $\mathcal{L}$ into $\mathcal{L}'$, and from $\mathcal{L}$ to $\mathcal{L}''$, then there is a valid translation up to $\sim$ from $\mathcal{L}$ to $\mathcal{L}''$

Definition 3.2.11: Congruence

Let $\mathcal{L}$ be a simple language. An equivalence relation $\sim$ on $T_{\mathcal{L}}$ is a **congruence** for $\mathcal{L}$ if

$$P_1 \sim Q_1 \wedge \cdots \wedge P_n \sim Q_n \implies f(P_1, \ldots, P_n) \sim f(Q_1, \ldots, Q_n)$$

for each n-ary operator f of $\mathcal{L}$, and each $P_i, Q_i \in T_{\mathcal{L}} (i = 1, \ldots, n)$

Let $\mathcal{L}$ be a simple language. An equivalence relation $\sim$ on $T_{\mathcal{L}}$ is a **congruence** for $\mathcal{L}$ if

$$P_1 \sim Q_1 \wedge \cdots \wedge P_n \sim Q_n \implies C[P_1, \ldots, P_n] \sim C[Q_1, \ldots, Q_n]$$

for each n-ary context C, and each $P_i, Q_i \in T_{\mathcal{L}} (i = 1, \ldots, n)$

Theorem 3.2.12: Coarser Expressions

Suppose $\mathcal{L}'$ is at least as expressive as $\mathcal{L}$ up to an equivalene or preorder $\sim$, and let $\approx \supseteq \sim$ be a congruence for $\mathcal{L}'$ that is coarser or equal to $\sim$. Then $\approx$ is also a congruence for $\mathcal{L}$

Let $\mathcal{I}$ be a translation of $\mathcal{L}$ into $\mathcal{L}'$ that is valid up to $\sim$, and let $\approx \supseteq \sim$ be a congruence for $\mathcal{L}'$ that is coarser or equal to $\sim$. Then $\approx$ is also a congruence for $\mathcal{I}(\mathcal{L})$

Let $\mathcal{I}$ be a translation of $\mathcal{L}$ into $\mathcal{L}'$ that is valid up to $\sim$, and let $\approx \supseteq \sim$ be a congruence for $\mathcal{I}(\mathcal{L})$ that is coarser or equal to $\sim$. Then $\approx$ is also a congruence for $\mathcal{L}$

Definition 3.2.13: Translation Congruence

Let $\mathcal{I} : \mathbb{T}_{\mathcal{L}} \to \mathbb{T}_{\mathcal{L}'}$ be a translation from $\mathcal{L}$ to $\mathcal{L}'$. An equivalence relation $\sim$ on $T_{\mathcal{I}(\mathcal{L})}$ is a **congruence** for $\mathcal{I}(\mathcal{L})$ if

$$\mathcal{I}(P) \sim \mathcal{I}(Q) \implies \mathcal{I}(D)[\mathcal{I}(P)] \sim \mathcal{I}(D)[\mathcal{I}(Q)]$$

for each unary $\mathcal{L}$-context D, and each $P, Q \in T_{\mathcal{L}}$

3.3 Look into

- Bisimularity (strong / weak)
- pi calculi
- Barbed bisimilarity ?