Theory of Interaction
– what is a model theory of computer science?

Yuxi Fu
BASICS, Shanghai Jiao Tong University

Bologna, 22-23 April, 2013

Computation and Interaction

Thesis on Computation. All computation models share a common submodel that is physically implementable.

What is not formally treated in Turing's theory is a theory of observational equality.

Thesis on Interaction. All interaction models share a common submodel (CCS?).

What is lacking in Milner's framework is a proper treatment of computation.

I. Motivation

Why Model Theory?

Point I. In Computer Science a lot of models have been proposed. There is not yet a model theory.

- Computation Models
- Concurrency Models

Well, they are all about interactions.

Computation Theory and Process Theory have been two separated developments. An integrated treatment, Model Theory, ought to be beneficial to both theories.

Why Model Theory?

Point II. Some of the foundational assumptions of Computer Science are actually postulates in Model Theory.

- In Computability Theory, Church-Turing Thesis.
- In Complexity Theory, Extended Church-Turing Thesis.
- In Programming Theory, existence of universal program.

There is no way to formalize these foundational assumptions without a theory of models.

Why Model Theory?

Point III. Most basic concepts in Computer Science are model independent.

- expressiveness
- implementation
- correctness

Are there any basic concepts in Computer Science that are not model independent?

Model independence is basically a model theoretical concept. A model independent concept is defined in model theory.

Why Model Theory?

Point IV. Some of the fundamental problems in Computer Science are best understood when cast in the light of Model Theory.

- '$\mathbf{NP} \neq \mathbf{P}$?'
- Compare the above problem to '$\mathbf{BPP} = \mathbf{P}$?'

II. Basic Model Theory

Fundamental Relationships in Computer Science

Model Theory begins with two most fundamental relationships in Computer Science:

- the equality relationship '$=$' within a model, and
- the expressiveness relationship '$\sqsubseteq$' between models.

Both $=$ and $\sqsubseteq$ are of course model independent.

Foundational Assumption

How can we do model theory without being specific to any model?

Foundational Assumption

How can we do model theory without being specific to any model?

Model Theory is built upon four foundational principles that are just enough to define $=$, $\sqsubseteq$ in a model independent manner.

Four Principles

I. **Principle of Object**. There are two kinds of objects.

II. **Principle of Action**. There are two aspects of actions.

III. **Principle of Observation**. There are two universal operators.

IV. **Principle of Consistency**. There are two unequal objects.

Ideas from Process Theory and Computation Theory

In computation theory

- bisimulation is implicit in equivalence proofs
- divergent computation $\neq$ terminating computation

In process theory

- Milner and Park's bisimulation
- van Glabbeek and Weijland's branching bisimulation
- Milner and Sangiorgi's barbed bisimulation

Bisimulation

A binary relation $\mathcal{R}$ is a bisimulation if it validates the following bisimulation property:

1. If $Q\mathcal{R}^{-1}P \xrightarrow{\tau} P'$ then one of the following is valid:
   (i) $\exists Q'.Q \Longrightarrow Q'\mathcal{R}^{-1}P' \wedge Q'\mathcal{R}^{-1}P$.
   (ii) $\exists Q', Q''.Q \Longrightarrow Q''\mathcal{R}^{-1}P \wedge Q'' \xrightarrow{\tau} Q'\mathcal{R}^{-1}P'$.
2. If $P\mathcal{R}Q \xrightarrow{\tau} Q'$ then one of the following is valid:
   (i) $\exists P'.P \Longrightarrow P'\mathcal{R}Q' \wedge P'\mathcal{R}Q$.
   (ii) $\exists P', P''.P \Longrightarrow P''\mathcal{R}Q \wedge P'' \xrightarrow{\tau} P'\mathcal{R}Q'$.

Codivergence

A binary relation $\mathcal{R}$ is codivergent if the following codivergence property holds whenever $P\mathcal{R}Q$:

1. If $P \xrightarrow{\tau} P_1 \xrightarrow{\tau} \ldots \xrightarrow{\tau} P_{i+1} \ldots$ is an infinite internal action sequence then $\exists Q'.\exists i{\geq}1.Q \overset{\tau}{\Longrightarrow} Q'\mathcal{R}^{-1}P_i$;
2. If $Q \xrightarrow{\tau} Q_1 \xrightarrow{\tau} \ldots \xrightarrow{\tau} Q_{i+1} \ldots$ is an infinite internal action sequence then $\exists P'.\exists i{\geq}1.P \overset{\tau}{\Longrightarrow} P'\mathcal{R}Q_i$.

Equipollence

A process P is unobservable, notation $P \not\Downarrow$, if it never interacts.

P and Q are equipollent if $P\Downarrow \Leftrightarrow Q\Downarrow$.

$\mathcal{R}$ is equipollent if P and Q are equipollent whenever $P \mathcal{R} Q$.

Extensionality

$\mathcal{R}$ is extensional if the following extensionality property holds:

1. If $M \mathcal{R} N$ and $P \mathcal{R} Q$ then $(M \mid P) \mathrel{\mathcal{R}} (N \mid Q)$;
2. If $P \mathcal{R} Q$ then $(a)P \mathrel{\mathcal{R}} (a)Q$ for every name a.

Absolute Equality

The absolute equality $=_{\mathbb{M}}$ is the largest relation on $\mathbb{M}$-processes that validates the following statements:

1. It is reflexive.
2. It is equipollent, extensional, codivergent and bisimilar.

Subbisimilarity

A relation $\mathfrak{R}$ from the set of $\mathbb{M}_0$-processes to the set of $\mathbb{M}_1$-processes is a subbisimilarity, notation $\mathfrak{R} : \mathbb{M}_0 \rightarrow \mathbb{M}_1$, if it validates the following statements:

1. It is total and sound.
2. It is equipollent, extensional, codivergent and bisimilar.

We write $\mathbb{M}_0 \sqsubseteq \mathbb{M}_1$ if there is some subbisimilarity from $\mathbb{M}_0$ to $\mathbb{M}_1$.

Remark

$P =_{\mathbb{M}} Q$ means that P, Q are equal objects/processes of model $\mathbb{M}$.

$\mathbb{M} \sqsubseteq \mathbb{N}$ means that $\mathbb{N}$ is at least as expressive as $\mathbb{M}$.

Remark

$P =_{\mathbb{M}} Q$ means that P, Q are equal objects/processes of model $\mathbb{M}$.

$\mathbb{M} \sqsubseteq \mathbb{N}$ means that $\mathbb{N}$ is at least as expressive as $\mathbb{M}$.

Now we can write a logical formula in terms of $=$ and $\sqsubseteq$.

For example we may assume that the class $\mathfrak{M}$ of models to be dense by imposing the following postulate

$$\forall \mathbb{L}, \mathbb{N} \in \mathfrak{M}. \exists \mathbb{M} \in \mathfrak{M}. \mathbb{L} \sqsubset \mathbb{M} \sqsubset \mathbb{N}.$$

III. Axiom of Completeness

A correct formulation of Church-Turing Thesis is the starting point of Model Theory. Model Theory would be a failure if it could not support such a formalization.

Initial Model $\mathbb{C}$

Grammar of $\mathbb{C}$:

$$P \quad := \quad \mathbf{0} \mid \Omega \mid F_a^b(\mathrm{f}(x)) \mid \overline{a}(\underline{i}) \mid P \mid P,$$

where f is a computable function and $\underline{i}$ is a natural number.

Semantics of $\mathbb{C}$:

- $F_a^b(\mathrm{f}(x)) \xrightarrow{a(\underline{i})} \overline{b}(\underline{j})$ if $\mathrm{f}(\underline{i}) = \underline{j}$;
- $F_a^b(\mathrm{f}(x)) \xrightarrow{a(\underline{i})} \Omega$ if $\mathrm{f}(\underline{i})\uparrow$;
- $\overline{a}(\underline{j}) \xrightarrow{\overline{a}(\underline{j})} \mathbf{0}$;
- $\Omega \xrightarrow{\tau} \Omega$.

Formalizing Church-Turing Thesis

Axiom of Completeness. $\forall \mathbb{M} \in \mathfrak{M}.\ \mathbb{C} \sqsubseteq \mathbb{M}$.

A model $\mathbb{M}$ is said to be complete if $\mathbb{C} \sqsubseteq \mathbb{M}$.

Some Results

Theorem. Both $\mathbb{VPC}$ and π are complete.

Some Results

Theorem. Both $\mathbb{VPC}$ and π are complete.

Theorem. CCS is not complete.

Theorem. The higher order process calculus is not complete.

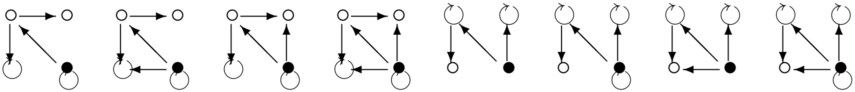
IV. Computation Theory

Nondeterminism

A one-step deterministic computation $A \to B$ is an internal action $A \xrightarrow{\tau} B$ such that $A = B$.

A one-step nondeterministic computation $A \xrightarrow{\iota} B$ is an internal action $A \xrightarrow{\tau} B$ such that $A \neq B$.

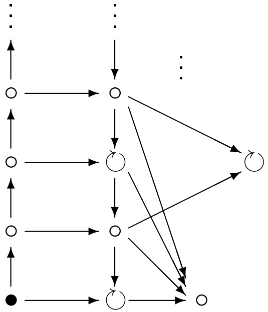
C-graph



There is a complete axiomatic system for the finite computations.

Infinite C-Graph

The following structure is definable for example by a π-process.



Infinite C-Graph

The infinite C-graph is defined by *Centipeda*:

$$Centipeda \quad = \quad (inc)(dec)(o)(e)(Cp \mid Cnt \mid o.O \mid e.E),$$

where

$$\begin{aligned}
Cp &= \tau.\Upsilon_0 + \tau.(\tau.\Upsilon_1 + \tau.(\overline{o} \mid !o.\overline{inc}.\overline{e} \mid !e.\overline{inc}.\overline{o})),\\
Cnt &= inc.(d)(A(d) \mid d),\\
A(x) &= dec.\overline{x} + inc.(d)(A(d) \mid d.A(x)),\\
E &= \mu X.(\tau.X + \tau + \overline{dec}.O),\\
O &= \tau + \tau.\Omega + \overline{dec}.E.
\end{aligned}$$

Nondeterminism is Model Independent

A model of interaction is a Turing-Milner model if it enjoys the following properties:

- The $\mathbb{M}$-processes are Gödel enumerable.
- The transition tree of every process is computable.

Theorem. Suppose $\mathbb{M}$ is a Turing-Milner model. Then

$$\forall P \in \mathbb{M}.\neg(P\Downarrow) \Rightarrow \exists Q \in \mathbb{C}.\neg(Q\Downarrow) \wedge Q = P.$$

Axiom of Computation.

$$\forall \mathbb{M} \in \mathfrak{M}. \forall P \in \mathbb{M}. \neg(P\Downarrow) \Rightarrow \exists Q \in \mathbb{C}. \neg(Q\Downarrow) \wedge Q = P.$$

V. Process Theory

Model Theory provides a basis for a systematic study and classification of the ‘700 process calculi’.

In fact most of these calculi are incomplete.

Largest Subbisimilarity?

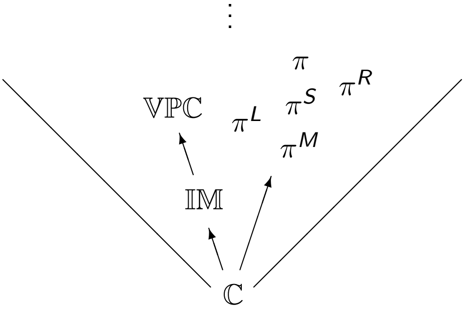
Theorem. There are an infinite number of subbisimilarities from $\mathbb{VPC}^!$ to $\mathbb{VPC}^!$.

Theorem. The largest subbisimilarity from a π-variant to itself exists and coincides with the absolute equality.

Old Result in New Theory

$$
\begin{tikzcd}[row sep=large, column sep=large]
\pi^{\mathrm{def}} \arrow[r, tail, dashed, shift left, "?"] \arrow[d, tail, shift right, "/" marking] & \pi \arrow[l, tail, shift left] \arrow[r, tail, shift left, "\backslash" marking] \arrow[d, tail, shift right, "/" marking] & \pi^{!} \arrow[l, tail, shift left] \arrow[d, tail, shift right, "/" marking] \\
\pi^{m\mathrm{def}} \arrow[u, tail, shift right] \arrow[r, tail, shift left] & \pi^{m} \arrow[u, tail, shift right] \arrow[l, tail, shift left] \arrow[r, tail, shift left] \arrow[d, tail, shift right, "/" marking] & \pi^{m!} \arrow[u, tail, shift right] \arrow[l, tail, shift left] \\
\pi^{s\mathrm{def}} \arrow[r, tail, shift left] & \pi^{s} \arrow[u, tail, shift right] \arrow[l, tail, shift left] \arrow[r, tail, shift left] & \pi^{s!} \arrow[l, tail, shift left]
\end{tikzcd}
$$

World of Model



Theorem. $\mathbb{VPC}^{\mathrm{def}} \not\sqsubseteq \pi \not\sqsubseteq \mathbb{VPC}^{\mathrm{def}}$.

Theorem. polyadic $\pi \not\sqsubseteq$ monadic π

VI. Programming Theory

Programming Theory is based on the existence of interpreter/universal process.

Interpreter

Suppose $\mathbb{L}, \mathbb{M}$ are complete and $\mathbb{L} \sqsubseteq \mathbb{M}$.

We intend to formalize the relationship saying that $\mathbb{M}$ is capable of interpreting all the $\mathbb{L}$-processes within $\mathbb{M}$.

Interpreter

An interpreter of $\mathbb{L}$ in $\mathbb{M}$ at c is a tuple $\langle [\![\_]\!], \propto, \{\mathcal{I}_c^{i,\jmath}\}_{i\in\mathbb{N},\jmath\in\mathcal{N}^*}\rangle$ where

- $[\![\_]\!]$ is an encoding of $\mathbb{C}$ into $\mathbb{M}$;
- $\propto$: $\mathbb{L} \sqsubseteq \mathbb{M}$ is a subbisimilarity;
- if k is a Gödel index of an $\mathbb{L}$-process P that has at most i distinct local names and contains no more global names than those appearing in $\jmath$, then some Q exists such that

$$[\![k]\!]_c^{\mathbb{M}} \,|\, \mathcal{I}_c^{i,a_1\ldots a_k} \xrightarrow{\iota} Q \propto^{-1} P.$$

We write $\mathbb{L} \in \mathbb{M}$ if there is an interpreter of $\mathbb{L}$ in $\mathbb{M}$.

$\mathbb{VPC}^! \in \mathbb{VPC}^{\mathrm{def}}$

Let $\mathbb{VPC}^!$ be the value-passing calculus with replication, and $\mathbb{VPC}^{\mathrm{def}}$ the value-passing calculus with parametric definition.

Theorem. $\mathbb{VPC}^! \in \mathbb{VPC}^{\mathrm{def}}$.

$\mathbb{VPC}^! \in \mathbb{VPC}^{\mathrm{def}}$

An index of a $\mathbb{VPC}^!$-process can be defined as follows:

$$
\begin{aligned}
[\![\mathbf{0}]\!]_{\mathfrak{v}} &\stackrel{\mathrm{def}}{=} 0, \\
[\![a(x).T]\!]_{\mathfrak{v}} &\stackrel{\mathrm{def}}{=} 7 * \langle \varsigma(a), \varsigma(x), [\![T]\!]_{\mathfrak{v}} \rangle + 1, \\
[\![\overline{a}(t).T]\!]_{\mathfrak{v}} &\stackrel{\mathrm{def}}{=} 7 * \langle \varsigma(a), [\![t]\!]_{\varsigma}, [\![T]\!]_{\mathfrak{v}} \rangle + 2, \\
[\![T \mid T']\!]_{\mathfrak{v}} &\stackrel{\mathrm{def}}{=} 7 * \langle [\![T]\!]_{\mathfrak{v}}, [\![T']\!]_{\mathfrak{v}} \rangle + 3, \\
[\![(c)T]\!]_{\mathfrak{v}} &\stackrel{\mathrm{def}}{=} 7 * \langle \varsigma(c), [\![T]\!]_{\mathfrak{v}} \rangle + 4, \\
[\![\mathit{if}\ \varphi\ \mathit{then}\ T]\!]_{\mathfrak{v}} &\stackrel{\mathrm{def}}{=} 7 * \langle [\![\varphi]\!]_{\varsigma}, [\![T]\!]_{\mathfrak{v}} \rangle + 5, \\
[\![!a(x).T]\!]_{\mathfrak{v}} &\stackrel{\mathrm{def}}{=} 7 * \langle \varsigma(a), \varsigma(x), [\![T]\!]_{\mathfrak{v}} \rangle + 6, \\
[\![!\overline{a}(t).T]\!]_{\mathfrak{v}} &\stackrel{\mathrm{def}}{=} 7 * \langle \varsigma(a), [\![t]\!]_{\varsigma}, [\![T]\!]_{\mathfrak{v}} \rangle + 7.
\end{aligned}
$$

The simulator $\mathcal{S}_{\mathfrak{v}}(z)$ is defined by the following

$$
\begin{array}{lll}
\multicolumn{3}{l}{\textbf{case } z \textbf{ of}} \\
\quad \mathrm{r}_7(z){=}0 & \Rightarrow & \mathbf{0}; \\
\quad \mathrm{r}_7(z){=}1 & \Rightarrow & \mathit{Nth}(\mathrm{d}_7(z)_0, j).a_j(x).\mathcal{S}_{\mathfrak{v}}([x/\mathrm{d}_7(z)_1]\mathrm{d}_7(z)_2); \\
\quad \mathrm{r}_7(z){=}2 & \Rightarrow & \mathit{Nth}(\mathrm{d}_7(z)_0, j).\overline{a_j}(\mathit{val}(\mathrm{d}_7(z)_1)).\mathcal{S}_{\mathfrak{v}}(\mathrm{d}_7(z)_2); \\
\quad \mathrm{r}_7(z){=}3 & \Rightarrow & \mathcal{S}_{\mathfrak{v}}(\mathrm{d}_7(z)_0) \mid \mathcal{S}_{\mathfrak{v}}(\mathrm{d}_7(z)_1); \\
\quad \mathrm{r}_7(z){=}4 & \Rightarrow & \mathit{Nth}(\mathrm{d}_7(z)_0, j).(a_j)\mathcal{S}_{\mathfrak{v}}(\mathrm{d}_7(z)_1); \\
\quad \mathrm{r}_7(z){=}5 & \Rightarrow & \mathit{if}\ \mathit{val}(\mathrm{d}_7(z)_0)\ \mathit{then}\ \mathcal{S}_{\mathfrak{v}}(\mathrm{d}_7(z)_1); \\
\quad \mathrm{r}_7(z){=}6 & \Rightarrow & \mathit{Nth}(\mathrm{d}_7(z)_0, j).!a_j(x).\mathcal{S}_{\mathfrak{v}}([x/\mathrm{d}_7(z)_1]\mathrm{d}_7(z)_2); \\
\quad \mathrm{r}_7(z){=}7 & \Rightarrow & \mathit{Nth}(\mathrm{d}_7(z)_0, j).!\overline{a_j}(\mathit{val}(\mathrm{d}_7(z)_1)).\mathcal{S}_{\mathfrak{v}}(\mathrm{d}_7(z)_2). \\
\multicolumn{3}{l}{\textbf{end case}}
\end{array}
$$

Parametric definition plays an essential role in the simulator.

A model $\mathbb{M}$ admits a universal process if $\mathbb{M} \in \mathbb{M}$.

Theorem. $\pi \in \pi$.

Theorem. $\mathbb{VPC}^{\mathrm{def}} \in \mathbb{VPC}^{\mathrm{def}}$.

However it is unlikely that $\mathbb{VPC}^{!} \in \mathbb{VPC}^{!}$.

Programming Theory

We say that $\mathbb{M}$ is a programming model if $\mathbb{M} \in \mathbb{M}$.

Fact. In the world of programming models, $\mathbb{L} \sqsubseteq \mathbb{M}$ iff $\mathbb{L} \in \mathbb{M}$.

Axiom of Programming. $\forall \mathbb{M}.\ \mathbb{M} \in \mathbb{M}$.

Application

1. Modeling security protocol
2. Process-Passing as Value-Passing

VII. Recursion Theory

Recursion Theory for Process

The existence of a universal process allows one to develop a Recursion Theory for an interaction model (say π).

Enumeration Theorem, S-m-n Theorem, Recursion Theorem, and then the rest of it.

Problem Classification, the Functional Case

A problem f is reducible to a problem g if there is a pair (e, d) of total computable functions such that

$$f = e; g; d. \tag{1}$$

We write $f \preccurlyeq_{(e,d)} g$, or simply $f \preccurlyeq g$, if (1) holds.

Let $\succcurlyeq$ denote the reverse relation of $\preccurlyeq$ and let $\approx$ denote $\preccurlyeq \cap \succcurlyeq$.

A problem type is an equivalence class of $\approx$.

Process Classification

Suppose e, d are total computable functions. A codivergent bisimulation $\mathcal{R}$ on $\mathbb{VPC}$ processes is a reduction via e, d if the following statements are valid:

1. If $Q\mathcal{R}^{-1}P \xrightarrow{a(i)} P'$ then $Q \Longrightarrow Q'' \xrightarrow{a(e(i))} Q'\mathcal{R}^{-1}P'$ and $P\mathcal{R}Q''$ for some Q', Q''.
2. If $P\mathcal{R}Q \xrightarrow{\overline{a}(j)} Q'$ then $P \Longrightarrow P'' \xrightarrow{\overline{a}(d(j))} P'\mathcal{R}Q'$ and $P''\mathcal{R}Q$ for some P', P''.
3. If $Q\mathcal{R}^{-1}P \xrightarrow{\overline{a}(i)} P'$ then $Q \Longrightarrow Q'' \xrightarrow{\overline{a}(j)} Q'\mathcal{R}^{-1}P'$, $P\mathcal{R}Q''$ and $i = d(j)$ for some Q', Q'', j.

We write $P \preceq_{(e,d)} Q$ if there is a functional simulation via e, d that contains the pair (P, Q).

Process Classification

$P \preceq Q$ if $P \preceq_{(e,d)} Q$ for some e, d.

Let $\simeq$ be $\preceq \cap \succeq$. A process type is an equivalence class w.r.t. $\simeq$.

We can then investigate the order structure of process type.

Process Degree

Given a model we can classify the processes that can interact with the processes of the model but are not definable in the model.

$$A \lesssim_\pi B \text{ iff } \exists P, \widetilde{c}. A = (\widetilde{c})(B \mid P).$$

Using oracle B we can implement A in π-calculus.

$[A]_\pi$, the degree of A, is the class $\{B \mid B \lesssim_\pi A \lesssim_\pi B\}$.

We can then investigate the order structure of process degree.

VIII. Complexity Theory

An Example

Let $\mathbf{P}_\pi$ denote the set of all functional processes whose computation paths are of polynomial length.

Fact. $\mathbf{P}_\pi = \mathbf{NP}$.

An Example

Let $\mathbf{P}_\pi$ denote the set of all functional processes whose computation paths are of polynomial length.

Fact. $\mathbf{P}_\pi = \mathbf{NP}$.

Theorem. Suppose $\mathbb{M}$ is a Turing-Milner model. Then $\mathbf{P}_\mathbb{M} = \mathbf{NP}$.

Thanks!