現状報告

斉藤 新
shin

1999 年 3 月 4 日

参考文献

[1] Eijiro Sumii and Naoki Kobayashi. **A generalized deadlock-free process calculus.** In Uwe Nestmann and Benjamin C. Pierce, editors, *HLCL '98: High-Level Concurrent Languages*, Vol. 16.3 of *entcs*, Nice, France, September 1998. Elsevier Science Publishers.

---

1 進捗状況

今まで π-calculus とその型システムの勉強をしてきました。ある程度のまとめとして先日 (2/23)、 π-calculus の型システムに関する (あやしい) サーベイをやりました。

今は、住井さんの Deadlock-free π-calculus [1] の型推論についての研究を始めたところです。が、まだ特に進展はありません。

1.1 Deadlock-free π-calculus の型推論

ここで用いられる π-calculus のプロセスの文法は以下のようになります (U_i については後述)。

$$
\begin{array}{rcll}
P & ::= & \textbf{new}\ x : [\tau_1, \ldots, \tau_n]/\{U_1, \ldots, U_m\}\ \textbf{in}\ P & \text{(channel creation)} \\
 & | & c!\,[a_1, \ldots, a_n] & \text{(asynchronous output)} \\
 & | & c?[x_1, \ldots, x_n].P & \text{(input prefix)} \\
 & | & P_1 \,|\, P_2 & \text{(parallel execution)} \\
 & | & \textbf{def}\ f[x_1 : \tau_1, \ldots, x_n : \tau_n] = P_1\ \textbf{in}\ P_2 & \text{(process definition)} \\
 & | & \textbf{if}\ x\ \textbf{then}\ P_1\ \textbf{else}\ P_2 & \text{(conditional branch)}
\end{array}
$$

型 τ としては論理型 *bool* と channel type $[\tau_1, \ldots, \tau_n]$ を考えます。プロセス定義は P_2 にローカルなプロセス (サブルーチン) P_1 を定義します。 P_2 の中からの P_1 の呼び出しは $f!\,[\tilde{a}] \,|\, \ldots$ と書けてこれは $\{\tilde{a}/\tilde{x}\}P_1 \,|\, \ldots$ とリダクションされます。

channel creation ではチャネルの型に *usage* U というものを付記し、 P の中でチャネル x がどう使われるか指定します。上の定義では x が $U_1, \ldots, U_n$ のいずれかに従って使われることを意味します。 usage の文法は以下。

$$
U \quad ::= \quad 0 \quad | \quad \mathrm{O}_a^t \quad | \quad \mathrm{I}_a^t.U \quad | \quad U_1 \,|\, U_2 \quad | \quad !U
$$

t は time tag、 O (I) はチャネルが出力 (入力) に使われることを表わし、 a は *attribute* を表わします。

attribute には c (*capability*) と o (*obligation*) があります。 c が意味するのは、そこで入出力を「もし行えば」成功するということであり、 o が意味するのは、そこで入出力を「成否にかかわらず」実行しなければいけないということです。例えば mutex channel の usage は $\mathrm{O}_o \,|\, !(\mathrm{I}_c.\mathrm{O}_o)$ であり、 linear channel では $\mathrm{I}_{co} \,|\, \mathrm{O}_{co}$ となります。例えば以下のような宣言は正しい usage の宣言になります (ただし deadlock する)。

$$
\textbf{new}\ x : [\tau]/\{(\mathrm{I}_c \,|\, \mathrm{I}_c \,|\, \mathrm{O}_o)\}\ \textbf{in}\ (x?[v_1].P_1 \,|\, x?[v_2].P_2 \,|\, x!\,[w])
$$

よってこの calculus ではチャネルの型と usage および time tag を推論することになります。もちろん全部推論することはできないので、いったいどこまでプログラマが指定すればよいかを考えていくことになります。

目次

1 進捗状況 **1**

1.1 Deadlock-free π-calculus の型推論 2
