Abstract Interpretation of the \( \pi \)-Calculus

Arnaud Venet*

LIX, École Polytechnique, 91128 Palaiseau, France.

\texttt{venet@lix.polytechnique.fr}

http://lix.polytechnique.fr/~venet

Abstract. We are concerned with the static analysis of the communication topology for systems of mobile processes. For this purpose we construct an abstract interpretation of a large fragment of the \( \pi \)-calculus which can be used as a metalanguage to specify the behaviour of these systems. The abstract domain is expressive enough to give accurate descriptions of infinite and non-uniform distributions of processes and communication channels. We design appropriate widening operators for the automatic inference of such information.

1 Introduction

The static analysis of communicating processes with dynamically changing structure - commonly called mobile processes - has been mostly studied in the particular case of CML-like programs [NN94, Col95a, Col95b]. The design of these analyses heavily depends on type and control-flow information that is specific to CML. In this paper we propose an analysis of communications in the \( \pi \)-calculus [Mil91, MPW92]. Our choice is motivated by the fact that the \( \pi \)-calculus is a widely accepted model for describing mobile processes. It is fairly simple and yet very expressive, since it can encode data structures [Mil91], higher-order communication [Mil91, San94a] and various \( \lambda \)-calculi [Mil92].

Our purpose is to analyze the communication topology of a system of mobile processes, i.e. the distribution of processes and communication channels during the evolution of the system. In the standard semantics of the \( \pi \)-calculus this information is encoded within a process algebra. We construct a refinement of this semantics where an instance of a process at a certain stage of evolution of the system is represented by the sequence of internal computations that lead to it. The communication channels are represented in turn as the congruence classes of an equivalence relation over the communication ports of all process instances. This semantic model originated in the study of data structures [Jon81] and has been successfully applied to the alias analysis of ML-like programs by A. Deutsch [Deu92a, Deu92b, Deu94] who designed a powerful analysis based on Abstract Interpretation [CC77, CC92].

C. Colby [Col95b] further extended Deutsch's work and built an analysis for a subset of CML which can discover non-uniform descriptions of infinitely growing communication topologies. However, the whole framework relies on having a

* This work was partly supported by ESPRIT BRA 8130 LOMAPS.
good approximation of the control-flow of the program before doing the analysis of communications. Whereas a closure analysis can be used for higher-order CML programs, there is no realistic solution for the \( \pi \)-calculus where the only kind of computation is communication. Therefore we design a new abstract interpretation which gets rid of this problem and still ensures a comparable level of accuracy. We use the technique of cofibered domains introduced in [Ven96] which allows us to infer simultaneously an approximation of control-flow and communications.

The paper is organized as follows. In Sect. 2 we present the syntax and semantics of the \( \pi \)-calculus. The refined semantics is described in Sect. 3. In Sect. 4 we present the basic concepts of Abstract Interpretation. We construct an abstract domain for the analysis of the \( \pi \)-calculus in Sect. 5. The abstract semantics of the \( \pi \)-calculus is described in Sect. 6. In Sect. 7 we design widening operators in order to make the analysis effective.

2 The \( \pi \)-Calculus

The basic entities in the \( \pi \)-calculus are names, which are provided to represent communication channels. The key point is that computation is restricted to the transmission and reception of names along channels. There are various possibilities of defining processes in the \( \pi \)-calculus, depending on the way communication, recursion and nondeterminism are handled. Our presentation is based on a subset of the polyadic \( \pi \)-calculus [Mil91]. Let \( \mathcal{X} = \{x, y, \ldots\} \) be an infinite set of names. The syntax of \( \pi \)-terms \( \mathcal{P} = \{P, Q, \ldots\} \) is given by the following grammar:

\[
P ::= \sum_{i \in I} \pi_i.P_i \quad \text{guarded sum of finitely many processes}
| \quad P | Q \quad \text{parallel composition}
| \quad \nu x.P \quad \text{restriction}
| \quad !t.P \quad \text{guarded replication}
\]

\( \pi ::= t | o \) atomic action

\( t ::= x(y) \) input

\( o ::= \overline{x}[y] \) output

where \( y \) is a (possibly empty) tuple \( (y_1, \ldots, y_n) \) of pairwise distinct names. We denote by 0 the empty sum of processes (that is when \( I = \emptyset \)). We call a nonempty sum \( \sum_{i \in I} \pi_i.P_i \) of processes an agent, and a replication \( !t.P \) a server. We use the common abbreviation that consists of omitting the trailing .0 in \( \pi \)-terms. The restriction \( \nu x.P \) binds the name \( x \) in \( P \) and an input guard \( x(y_1, \ldots, y_n).P \) binds the names \( y_1, \ldots, y_n \) in \( P \). We denote by \( fn(P) \) the set of names occurring free in a process \( P \). If \( x \) and \( y \) are tuples of names of the same length, we denote by \( \{y/x\} \) the substitution that maps any \( x_i \) to \( y_i \). The standard operational semantics of the \( \pi \)-calculus is given by a structural congruence \( \equiv \) and a reduction relation defined in Fig. 1 and Fig. 2.

\[\text{We use a restricted form of replication inspired from the one involved in the definition of PICT [Tur95]. Yet it is expressive enough to perform all major constructions in the } \pi \text{-calculus (data structures, higher-order communication, etc.).}\]