Postgraduate Course: Process Algebra

Introduction Part

Huibiao Zhu

Software Engineering Institute, East China Normal University

What is Process Algebra?

(1) The process calculi (or process algebras) are a diverse family of related approaches to formally modelling concurrent systems.

(2) Process calculi provide a tool for the high-level description of interactions, communications, and synchronizations between a collection of independent agents or processes.

(3) They also provide algebraic laws that allow process descriptions to be manipulated and analyzed, and permit formal reasoning about equivalences between processes (e.g., using bisimulation).

(4) Leading examples of process calculi include CSP, CCS, ACP, LOTOS, π-calculus.

History of Process Algebra

(1) Research on process calculi began in earnest with Robin Milner's seminal work on the Calculus of Communicating Systems (CCS) during the period from 1973 to 1980.

(2) C.A.R. Hoare's Communicating Sequential Processes (CSP) first appeared in 1978, and was subsequently developed into a fully-fledged process calculus during the early 1980s. There was much cross-fertilization of ideas between CCS and CSP as they developed.

(3) In 1982 Jan Bergstra and Jan Willem Klop began work on what came to be known as the Algebra of Communicating Processes (ACP), and introduced the term process algebra to describe their work.

(4) CCS, CSP, and ACP constitute the three major branches of the process calculi family: the majority of the other process calculi can trace their roots to one of these three calculi.

Communicating Sequential processes (1)

(1) In computer science, Communicating Sequential Processes (CSP) is a formal language for describing patterns of interaction in concurrent systems. CSP was influential in the development of the occam programming language.

(2) CSP was first described in a 1978 paper by C. A. R. Hoare, but has since evolved substantially. CSP has been practically applied in industry as a tool for specifying and verifying the concurrent aspects of a variety of different systems such as the T9000 Transputer,and a secure ecommerce system.

Reference: Hoare, C. A. R. (1978). "Communicating sequential processes". Communications of the ACM 21 (8): 666–677

(3) The theory of CSP itself is also still the subject of active

research, including work to increase its range of practical applicability (e.g. increasing the scale of the systems that can be tractably analyzed).

The Development of CSP

(1) Following the publication of the original version of CSP, Hoare, Stephen Brookes, and A. W. Roscoe developed and refined the theory of CSP into its modern, process algebraic form.

(2) The theoretical version of CSP was initially presented in a 1984 article by Brookes, Hoare, and Roscoe, and later in Hoare's book Communicating Sequential Processes[7], which was published in 1985. In September 2006, that book was still the third-most cited computer science reference of all time according to Citeseer.

Reference:

[a] Hoare, C. A. R.. Communicating Sequential Processes. Prentice Hall. ISBN 0-13-153289-8.

[b] Brookes, Stephen; C. A. R. Hoare and A. W. Roscoe (1984). "A

Theory of Communicating Sequential Processes". Journal of the ACM 31 (3): 560C599.

(c) Stephen D. Brookes, A. W. Roscoe: An Improved Failures Model for Communicating Processes. Seminar on Concurrency 1984: 281-305

(3) The theory of CSP has undergone a few minor changes since the publication of Hoare's book. Most of these changes were motivated by the advent of automated tools for CSP process analysis and verification. Roscoe's The Theory and Practice of Concurrency describes this newer version of CSP.

Reference:

[a] Roscoe A. W. (1994). "Model-checking CSP". In A Classical Mind: essays in Honour of C.A.R. Hoare. Prentice Hall.

[b] Roscoe, A. W. (1997). The Theory and Practice of Concurrency. Prentice Hall. ISBN 0-13-674409-5.

Example of System Described using CSP

Example 1 (Bank Accounts): East customer of a bank first opens an account. He then makes any number of deposits and withdrawals, and finally may terminate his account. Let us initially ignore the amount of each deposit or withdrawal, and not worry whether the account is in credit or debit. Construct the process.

(1) Events

$$\alpha ACC =_{df} \{open,\ deposit,\ withdraw,\ terminate\}$$

(a) *open*——Opening a bank account.

(b) *deposit*——Depositing the money into the bank account.

(c) *withdraw*——Withdrawing the money from the bank account.

(e) *teminate*——Closing the bank account.

(2) Processes

$$ACC \;=\; open \rightarrow ACC11$$

$$ACC11 \;=\; deposit \rightarrow ACC11 \mid withdraw \rightarrow ACC11 \mid terminate \rightarrow STOP$$

Example 2 (Bank accounts again): A customer may query his account at any time before the account is terminated. Please add one new process in parallel with the original process (Example 1). Please construct the new process and the whole system.

(1) **The New Process:**

$\alpha ACC12 =_{df} \{query,\ terminate\}$

query——Querying the bank account.

$ACC12 \ = \ query \rightarrow ACC12 \mid terminate \rightarrow STOP$

(2) **The Whole System:**

$ACC' \ = \ ACC \parallel ACC12$

World Leading Research in CSP

1. Oxford University (Computing Lab, Programming Research Group)

Bill Roscoe (http://web.comlab.ox.ac.uk/people/Bill.Roscoe/):

(a) Concurrency
(b) Verification
(c) FDR
(d) Computer Security

2. Other institutes, including University of Surrey (Steven Schneider)

(a) Theory of Concurrency and its Applications
(b) Integration of Formal Methods, specifically CSP and B
(c) Security (in secure electronic voting)

Extending CSP (1)

(1) Timed-CSP

New statements: *wait t*

(a) Mike Reed's work (Oxford University, 1986; Now at UNU/IIST)

Paper: George M. Reed, A. W. Roscoe. A Timed Model for Communicating Sequential Processes. Theor. Comput. Sci. 58: 249-261 (1988)

(b) Jim Davies and Steve Schneider's work (Oxford University, 1993)

Book: Jim Davies: Specification and proof in real-time CSP, September 1993

Paper: Jim Davies, Steve Schneider. A brief history of Timed CSP, February 1995, Theoretical Computer Science

Extending CSP (2)

(2) Probability CSP

(a) Karen Seidel's work (Oxford University, 1992)

New statements: $P \ {}_r \sqcap \ Q$

Paper: Karen Seidel. Probabilistic Communicating Processes. Theor. Comput. Sci. 152(2): 219-249 (1995)

(b) Gavin Lowe's work (Oxford University, 1993)

Dphil Thesis: Gavin Lowe. Probabilities and priorities in timed CSP, University of Oxford, Oxford, 1993

Combining CSP with Other Formal Methods

(1) Michael Butler (CSP2B)

The csp2B tool provides a means of combining CSP-like descriptions with standard B specifications. The notation of CSP provides a convenient way of describing the order in which the operations of a B machine may occur. The function of the tool is to convert CSP-like specifications into standard machine-readable B specifications which means that they may be animated and appropriate proof obligations may be generated. Use of csp2B means that abstract specifications and refinements may be specified purely using CSP or using a combination of CSP and B.

(2) Steve Schneider ($CSP \parallel B$)

This combination of CSP and B is appropriate for systems whose description involves both control flow or interaction, and a significant state or data aspect.

The resulting method, $CSP \parallel B$, is designed to enable maximum use of the mature industrial strength tools for CSP and for B, within a single framework.

(3) Dong Jin Song (TCOZ=Timed-CSP+Object Z)

Timed CSP and Object-Z complement each other in their expressiveness. Object-Z has strong data and state modeling capabilities. Timed CSP has strong process control modeling capabilities. The multi-threading and synchronisation primitives of CSP are extended with timing primitives. The approach taken in TCOZ is to identify operation schemas (both syntactically and semantically) with (terminating) CSP processes that perform only state update events; to identify active classes with non-terminating CSP processes; and to allow arbitrary (channel-based) communications interfaces between objects.

Calculus of communicating systems (CCS)

The Calculus of Communicating Systems (CCS) is a process calculus introduced by Robin Milner in around 1980. Its actions model indivisible communications between exactly two participants.

The expressions of the language are interpreted as a labelled transition system. Between these models, bisimulation is used as a semantic equivalence.

References:

[a] R. Milner. A Calculus of Communicating Systems, volume 18 of Lecture Notes in Computer Science. Springer-Verlag, 1980.

[b] R. Milner. Communication and Concurrency. Prentice Hall International Series in Computer Science, 1990.

Examples of CCS

(1) An Agent C (a cell which may hold a single data item)

$$\begin{aligned} C &=_{df} in(x).C'(x) \\ C'(x) &=_{df} \hat{out}(x).C \end{aligned}$$

(2) $Buff_n$ (a buffer of capacity n)

$$\begin{aligned} Buff_n\langle\rangle &=_{df} Buff_n\langle x\rangle \\ Buff_n\langle v_1,...,v_n\rangle &=_{df} \hat{out}(v_n).Buff_n\langle v_1,...,v_{n-1}\rangle \\ Buff_n\langle v_1,...,v_k\rangle &=_{df} in(x).Buff_n\langle x,v_1,...,v_k\rangle \ + \\ &\qquad \hat{out}(v_n).Buff_n\langle v_1,...,v_{k-1}\rangle \ \ (0<k<n) \end{aligned}$$

π-calculus

(1) In theoretical computer science, the π-calculus is a process calculus originally developed by Robin Milner, Joachim Parrow and David Walker as a continuation of work on the process calculus CCS (Calculus of Communicating Systems).

(2) The π-calculus allows channel names to be communicated along the channels themselves, and in this way it is able to describe concurrent computations whose network configuration may change during the computation.

Algebra of Communicating Processes

(1) The Algebra of Communicating Processes (ACP) is an algebraic approach to reasoning about concurrent systems. It is a member of the family of mathematical theories of concurrency known as process algebras or process calculi.

(2) ACP was initially developed by Jan Bergstra and Jan Willem Klop in 1982,as part of an effort to investigate the solutions of unguarded recursive equations.

(3) More so than the other seminal process calculi (CCS and CSP), the development of ACP focused on the algebra of processes, and sought to create an abstract, generalized axiomatic system for processes, and in fact the term process algebra was coined during the research that led to ACP.