International Journal of applied exercise physiology

ISSN: 2322-3537
Vol-13 Issue-02 July 2024

Formalization of Signalling System by Process Calculus

Nagam Aanjaneyulu, Lankala Mounika, Shaik Guntur Mahabub Subhani,

Dr. Godagala Madhava Rao

Associate Professor[1,2,3], Professor[4]

anji.amrexamcell@gmail.com[1], lankala.mounikareddy@gmail.com[2], subhanimehandi@gmail.com[3],madhavaraog175@gmail.com[4]

Department of CSE, A.M. Reddy Memorial College of Engineering and Technology, Petlurivaripalem, Narasaraopet, Andhra Pradesh

Abstract

Core finance, business resource planning, electronic medical record, telephone exchange, and other similar systems all rely heavily on software. This is true not only in the public sector but also in the commercial sector. Such essential systems should be resilient in addition to being error-free. Software verification and other formally based approaches have historically prioritized accuracy over resilience. In order to formally and analytically assess the resilience of communication methods to Denial-of-Service assaults, Ishigaki et al. suggested a calculus procedure called the Spice calculus. The central concept is a cost-benefit study of communication protocol initiators and receivers. In this article, we detail a novel approach to analysing distributed systems using the Spice calculus and expand the scope of systems intended for study by this tool. As an illustration of our approach, we analyse a telephone communication system. In this article, we demonstrate a novel approach to analysing distributed systems using the Spice calculus and broaden the scope of systems in which it can be applied. As an illustration of our approach, we analyse a telephone communication system. Within the context of the Spice calculus, we defined a basic telephone network based on ISUP of Signalling System No. 7. We then looked at the time and resources it took to create connections and the computations they required.

Keywords:

Process Calculus, Communication Protocol, Telephone Exchange System, Cost Analysis.

Introduction

Software systems play an important role in social infrastructures, and in the public and private sectors, for example in core banking, enterprise resource planning systems, electronic medical record systems, telephone exchange systems, etc. In such fundamental systems, we desire not only correctness with respect to specification requirements but also robustness to cope with unintended situations. Formal methods, such as software verification, have focused on correctness rather than robustness. Many failures have occurred in such IT systems. For example, the core banking system of Mizuho Financial Group, which is the largest banking company in Japan, failed on March 14th 2011 and did not recover until the 23rd, owing to an overload caused by numerous payments of contributions for the Tohoku Earthquake[7]. Such lengthy failures of computer systems can be mitigated by improvements in system robustness. Formal methods, such as software verification, have focused on correctness rather than robustness. Spice calculus [9], which was proposed by Nishizaki et al., is a calculus process which enables us to formalize and analyze the resistance of communication protocols against Denial-of-Service attacks, based on Milner's picalculus [6]. The key idea of the analysis is analysing costs of initiators and responders in communication protocols. Vulnerability to Denial-of-Service attacks is caused by lack of balance between an initiator and a responder. Cost analysis is important not only for resistance against Denial-of-Service attacks but also for improving a system's robustness to withstanding various unintended situations. In this paper, we therefore apply Spice calculus to analysis of robustness of distributed systems. We conduct a case study using a telephone exchange system, as an example.

Spice calculus

We introduce Spice calculus [9] in this section. In the calculus, there are two categories of expression:

International journal of applied exercise physiology
ISSN: 2322-3537
Vol-13 Issue-02 July 2024

one is a set of terms and the other is a set of processes. A term denotes data and a process denotes a series of computational operations which can be executed concurrently. Terms of spice-calculus are defined inductively as follows:

$$\begin{array}{lll} M,N ::= & & \text{term} \\ & n & \text{name} \\ | & x & \text{variable} \\ | & i & \text{integer} \\ | & (M_1,\ldots,M_n) & \text{pair} \end{array}$$

Processes of Spice calculus are defined inductively as follows:

$$\begin{array}{lll} P,Q,R ::= & & \\ & \text{out } M\,\langle N\rangle;P & \text{message sending} \\ | & \text{inp } M\,(x);P & \text{message receiving} \\ | & (P \mid Q) & \text{parallel composition} \\ | & \text{stop} & \text{stopping} \\ | & \text{end} & \text{termination} \\ | & \text{store } x = M;P & \text{memory allocation} \\ | & \text{free } x;P & \text{memory deallocation} \\ | & \text{match } M \text{ is } N \text{ err}\{R\};P & \text{matching} \\ | & \text{split } [x_1,\ldots,x_n] \text{ is } M \text{ err}\{R\};P & \text{pair decomposition} \\ | & (P+Q) & \text{selection} \end{array}$$

Message sending (out M و ةÆ——— ਨ sends message N through the port M. Message receiving (in M (x); P) receives the message through port M and binds it to the variable x. Parallel decomposition (P | Q) executes the processes P and Q in parallel. The stopping process stop means an intermediate status on the way to the termination process end. In our computational system, memory allocation and de-allocation are explicitly operated as process (store x = M; P) and (free x; P), respectively. Matching (match M is N err {P}; Q) is a conditional branch: if the values of M and N are equivalent, then P is executed; otherwise, Q is executed. Selection (P+Q) is a non-deterministic choice; either P or Q is executed non-deterministically. The type system of Spice calculus is one of its distinctive features, and it enables us to specify the computational cost entailed by each computer node.

$$A \mid (B \mid A) \rhd P_1 \mid (Q \mid P_2),$$

which describes that the processes P1 and P2 are executed on the computer nodes A and Q on B. Types of Spice calculus are defined as follows:

$$A ::= \mathbf{a} : \{x_1,\ldots,x_n\} \mid (A \mid B)$$

Operational semantics are given to the Spice calculus as transition relation between processes; a computational cost is attached to each step of the transition. For example, this transition means that the processes P and Q are executed in parallel on nodes a and b respectively; the process (P ф Q) makes a transition to (P’ ф Q’);

$$(\mathbf{a} \mid \mathbf{b}) \vdash (P \mid Q) \twoheadrightarrow (P' \mid Q') : \{\mathbf{a}\cdot 2\cdot match, \mathbf{b}\cdot store,\}$$

In this transition, matching and storing operations are done with two match-costs and one store-cost, respectively. The transition relation is defined inductively by several rules (c.f. [9]). The following is one of the reduction rules defining the inner-process computation:

$$\frac{M \downarrow V : c \quad N \downarrow V : d}{(\text{match } M \text{ is } N \text{ err}\{P\};Q) \; > \; Q \; : \; c+d+match} \; \textbf{RedMatch}$$

If it is assumed that the costs for evaluating M and N are c and d respectively, and both the resulting values of M and N are V, then the expression is transited to Q and the computational cost of the transition is (cadmic). The rule below is one of the reduction rules defining the inter-process computation:

$$\frac{}{\mathbf{a}::\mathcal{V} \vdash \text{inp } n\,(x);P \xrightarrow{n} (x)P : \{\mathbf{a}\cdot store_x\}} \; \textbf{CommIn}$$

This rule means that if the process expression (in n (x); P) is executed on computer node a, then memory cost store for a variable x on a. It is transited to an expression (x)P, which denotes an intermediate form waiting for the arrival of a message.

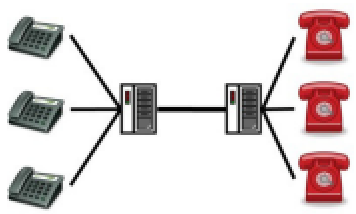


Fig. 1. Example of Switched Telephone Network

ISSN: 2322-3537
Vol-13 Issue-02 July 2024

Fig. 2. Inter-city Call

Formalization of a signalling system using spice calculus

Signalling System No. 7 and ISUP

In this section, we formalize a switched telephone network based on a telephony signalling protocol, Signalling System No. 7 (SS7)[3], using Spice calculus [9]. We focus on an example on a simple switched telephone network which consists of two switches and several telephones. Its network topology is shown in Fig. 1. In SS7, ISDN User Part (ISUP) defines the messages and protocol used in the establishment and tear down of calls over the public switched network, and to manage the trunk network on which they relay. ISUP is depicted as a sequence diagram in Fig. 2. Terminal nodes Phone and Phone are connecting, and intermediate switches Switch A and SwitchB relay the connection. Control messages –iam, acm, anm, rel, and rlc –are defined in ISUP. The messages used are as follows. x IAM (Initial address message), which is the first message sent to inform the partner switch that a call has to be established. x ACM (Address complete message), which is returned from the terminating switch when the subscriber is reached. x ANM (Answer message), which is sent when the subscriber picks up the phone. xREL (Release), which is sent to clear the call when a subscriber goes on hook. xRLC (Release completed), which is acknowledgement of the release. The state transition diagram of Fig. 3 describes the behavior of a phone. The model of the phone receives messages: dial tone, ringtone, no tone, busy tone, and check. The former four messages represent notification of status from the partner switch, and the latter message gives confirmation of connect ability. The model sends messages off hook, unhook, and dial. The former two messages describe signals that the handset is offhooking and on-hooking, respectively. The latter message represents a dialling signal.

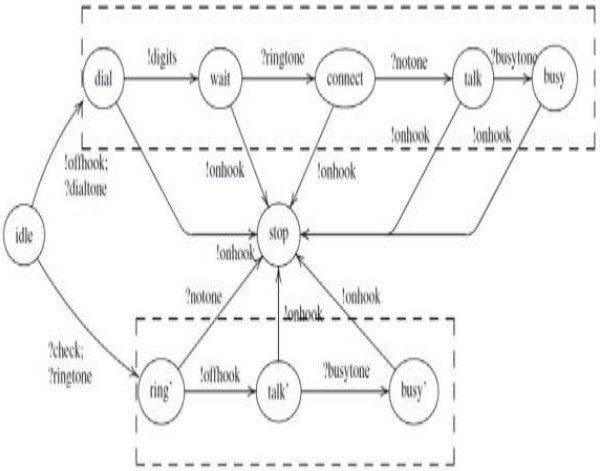


Fig. 3. Transition diagram of phone

Formalization of Phones and Switches in Spice Calculus

In Spice calculus, the behaviour of phones (i.e., subscribers) and switches is formalized as processes. The behaviour of a phone a with its partner switch A which has a connection with phone b with its partner switch B, is described as Subscriber(a,A,B,b), which is divided in two sub-processes, Sender(a,A,B,b) and Receiver(a, A):

$$Subscriber(a,A,B,b) = Sender(a,A,B,b) + Receiver(a,A)$$

$$\begin{aligned}
&Sender(a,A,B,b)\\
&= \text{out } localport(A,A)\ \langle(\text{“}offhook\text{”},A)\rangle; \text{store } port = localport(a,A);\\
&\quad \text{inp } port\ (msg);\\
&\quad \text{match } msg \text{ is “}dialtone\text{”} \ \text{err}\{\text{free } msg,port\}; \text{free } msg;\\
&\qquad \text{out } port\ \langle\text{“}onhook\text{”}\rangle; \text{stop};\\
&\quad + \text{out } port\ \langle(\text{“}digits\text{”},B,b)\rangle; \text{out } port\ \langle\text{“}onhook\text{”}\rangle; \text{stop}\\
&\quad + \text{inp } port\ (\text{“}msg\text{”});\\
&\qquad \text{match } msg \text{ is “}ringtone\text{”} \ \text{err}\{\text{free } msg,port\}; \text{free } msg;\\
&\qquad \text{out } port\ \langle\text{“}onhook\text{”}\rangle; \text{stop};\\
&\quad + \text{inp } port\ (msg);\\
&\qquad \text{match } msg \text{ is “}notone\text{”} \ \text{err}\{\text{free } msg,port\}; \text{free } msg;\\
&\qquad \text{out } port\ \langle\text{“}onhook\text{”}\rangle; \text{stop};\\
&\quad + \text{inp } port\ (msg);\\
&\qquad \text{match } msg \text{ is “}busytone\text{”} \ \text{err}\{\text{free } msg,port\}; \text{free } msg;\\
&\qquad \text{out } port\ \langle\text{“}onhook\text{”}\rangle; \text{stop};
\end{aligned}$$

ISSN: 2322-3537
Vol-13 Issue-02 July 2024

```
Receiver(a,A)
= store port = localport(a,A);
  inp port (msg);
  match msg is "check" err{free msg,port};free msg;
  inp port (msg);
  match msg is "ringtone" err{free msg,port};free msg;
    (inp port (msg);
     match msg is "notone" err{free msg,port};free msg;stop;)
  + out port ("offhook");
      (out port ("onhook");stop);
    + inp port ("msg");
      match msg is "busytone" err{free msg,port};free msg;
      out port ("onhook");stop;
```

The ports which are used in the processes for communication are depicted in Figs. 4 and 5.

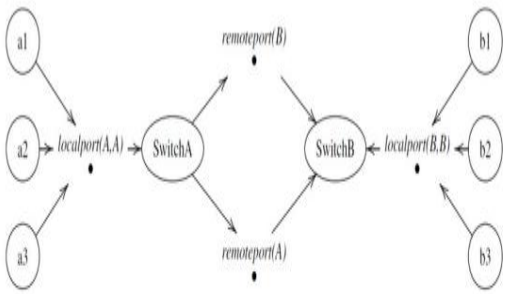


Fig. 4. Intermediate Ports (before establishing connection)

Fig. 5. Intermediate Ports (after establishing connection)

Here introduces the paper, and add nomenclature, if necessary, in a box with the same font size as the rest of the paper. The paragraphs continue from here and are only separated by headings, subheadings, images and formulae. The section headings are arranged by numbers, and are in bold and 10 pt. Here follows further instructions for authors.

Cost Estimation by Spice Calculus

In this section, we show a cost analysis of a case in which a subscriber a with a partner switch A requests a connection to a subscriber b with a partner switch B. This is simpler than the situation mentioned in Figs. 4 and 5. Subscribers a1, a2 and a3 are simplified into only one subscriber a; b1, b2, and b3 are simplified into b. The behaviour of this case is described as the following process.

$$Sender(a,A,B,b) \mid Switch(A) \mid Switch(B) \mid Receiver(b,B).$$

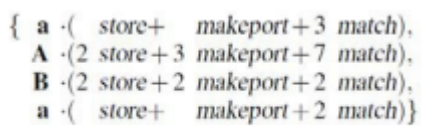
By reducing the process described the system; we can see not only the computational result but also the total cost.

$$\begin{aligned}
&Sender(a,A,B,b) \mid Switch(A) \mid Switch(B) \mid Receiver(b,B)\\
&\gg Sender(a,A,B,b) \mid LocalSwitch(A) \mid RemotelSwitch(B) \mid Receiver(b,B)\\
&\twoheadrightarrow (\text{out } port\ \langle(\text{“}digits\text{”},B,b)\rangle;\cdots) \mid (\text{inp } localport\ (msg);\cdots)\\
&\qquad \mid RemotelSwitch(B) \mid Receiver(b,B)\\
&\quad : \{\mathbf{a}\cdot(\text{store}+\text{makeport}+\text{match}),\mathbf{A}\cdot(3\text{store}+\text{makeport}+\text{match})\}\\
&\quad \text{Let } port \text{ be } port_a \text{ and } localport\ port_a \text{ below.}\\
&\twoheadrightarrow (\text{inp } port_a\ (msg);\cdots) \mid (\text{out}\cdots\langle(\text{“}iam\text{”},\cdots)\rangle;\cdots)\\
&\qquad \mid RemotelSwitch(B) \mid Receiver(b,B)\\
&\quad : \{\mathbf{A}\cdot(2\text{store}+\text{makeport}+3\text{match})\}\\
&\twoheadrightarrow \cdots \mid (\text{inp } localport\ (\text{“}check\text{”});\cdots)\\
&\qquad \mid (\text{out } localport\ \langle(\text{“}check\text{”},\cdots)\rangle;\cdots) \mid Receiver(b,B)\\
&\quad : \{\mathbf{A}\cdot(2\text{store}+\text{makeport}+3\text{match})\}\\
&\twoheadrightarrow \cdots \mid (\text{inp } localport\ (\text{“}check\text{”});\cdots)\\
&\qquad \mid (\text{out } localport\ \langle(\text{“}check\text{”},\cdots)\rangle;\cdots) \mid Receiver(b,B)\\
&\quad : \{\mathbf{A}\cdot(2\text{store}+\text{makeport}+3\text{match})\}\\
&\twoheadrightarrow \cdots \mid \cdots\\
&\qquad \mid (\text{out } localport\ \langle\text{“}check\text{”}\rangle;\cdots) \mid (\text{inp } port\ (msg);\cdots)\\
&\quad : \{\mathbf{b}\cdot(\text{store}+\text{makeport})\}\\
&\quad \text{Let } port \text{ be } port_b \text{ and } localport\ port_b \text{ below.}\\
&\twoheadrightarrow \cdots \mid (\text{inp } remoteport\ (msg);\cdots)\\
&\qquad \mid (\text{out } localport\ \langle\text{“}acm\text{”}\rangle;\cdots) \mid (\text{inp } port_b\ (msg);\cdots)\\
&\quad : \{\mathbf{b}\cdot match\}\\
&\twoheadrightarrow (\text{inp } port_a\ (msg);\cdots) \mid (\text{out } port_a\ \langle\text{“}ringtone\text{”}\rangle;\cdots)\\
&\qquad \mid (\text{out } port_b\ \langle\text{“}ringtone\text{”}\rangle;\cdots) \mid \cdots\\
&\quad : \{\mathbf{A}\cdot 2match\}
\end{aligned}$$

$$\begin{aligned}
&\twoheadrightarrow (\cdots) \mid (\text{inp } remoteport\ (msg);\cdots)\\
&\qquad \mid (\text{inp } port_b\ (msg);\cdots) \mid (\text{out } port_b\ (\text{“}offhook\text{”});\cdots)\\
&\quad : \{\mathbf{b}\cdot match\}\\
&\twoheadrightarrow (\cdots) \mid (\text{inp } remoteport\ (msg);\cdots)\\
&\qquad \mid (\text{out } port_b\ (\text{“}anm\text{”});\cdots) \mid (\cdots session\ started\cdots)\\
&\quad : \{\mathbf{B}\cdot match\}\\
&\twoheadrightarrow ((\text{inp } port_a\ (msg);\cdots) \mid (\text{out } port_a\ (\text{“}notone\text{”});\cdots)\\
&\qquad \mid (\cdots session\ started\cdots) \mid (\cdots session\ started\cdots)\\
&\quad : \{\mathbf{A}\cdot match\}\\
&\twoheadrightarrow (\cdots session\ started\cdots) \mid (\cdots session\ started\cdots) \mid (\cdots session\ started\cdots) \mid (\cdots session\ started\cdots)\\
&\quad : \{\mathbf{a}\cdot match\}
\end{aligned}$$

Summing up the costs attached to each transition step, we can see the total computation cost is as follows:

International Journal of applied



ISSN: 2322-3537
Vol-13 Issue-02 July 2024

Related Works

Protocol verification has been studied for the last decade using several kinds of framework. For example, Bruns et al. [1] studied the MSMIE protocol, which allows processors in a distributed system to communicate via shared memory. They provided a formal model of the MSMIE protocol in Milner's CCS and analysed the formal model using an automated verification tool, the Concurrency Workbench. Quantitative analysis of communication and secure protocols was studied by one of the authors [9], but also by Cervelat [2], which enables evaluation of protocol resilience to various forms of denial of service, guessing attacks, and resource limitation using the MSR (Multi-Set Rewriting) specification language. Liu et al. [5] formalized and verified in the framework of a coloured petri net the capability exchange signalling (CES) protocol, which is a sub-protocol of the H.245 control protocol for multimedia communication.

Conclusion

Due to the time and labour-intensive nature of human Appraisal labelling, there are not many such datasets accessible. While sentiment analysis has made great strides in determining a text's overarching meaning slant, the Attitude Appraisal component is still holding out. In order to pinpoint the source of the issues and offer suggestions for how to fix them, a simple automated recognizer was developed and tried for this research. No distinction is made between authorial and non-authorial assessment; the focus is solely on written Appraisal.

References

[1] Bruns, G., Anderson, S.: The formalization and analysis of a communications protocol. Formal Aspects of Computing 6(1), 92—112 (1994).

[2] Cervelat, I.: Towards a Notion of Quantitative Security Analysis. In: Dolman, F. Masaccio, A.Yautsiukhin (eds.) Quality of Protection: Security Measurements and Metrics — QoP'05, pp. 131—144. Springer-Verlag Advances in Information Security 23 (2006)

[3] Dryburgh, L., Hewitt, J.: Signalling System No. 7 (SS7/C7): Protocol, Architecture, and Services. Cisco Press (2004)

[4] Ikeda, R., Narita, K., Ishigaki, S.: Cooperation of model checking and network simulation of cost analyses of distributed systems. International Journal of Computers and Applications 33(4) (2011)

[5] Liu, L., Billington, J.: Verification of the capability exchange signalling protocol. International Journal on Software Tools for Technology Transfer 9(3—4), 305—326 (2007).

[6] Milner, R., Parrow, J., Walker, D.: A calculus of mobile processes, part I and part ii. Information and Computation 100(1), 1—77 (1992)

[7] Causes and Plans for Improvements and Counter—Measures based on the Recent Computer System Failures. Mizuho Financial Group (2011). http://www.mizuho-fg.co.jp/english/csr/mizuhocsr/calendar/2010/highlight_system/plan.html

[8] Ishigaki, S.: Polymorphic environment calculus and its type inference algorithm. Higher-Order and Symbolic Computation, Kluwer 13(3), 239—278 (2000)

[9] Tomioka, D., Nishizaki, S., Ikeda, R.: A cost estimation calculus for analysing the resistance to denial-ofservice attack. In: Software Security — Theories and Systems., Lecture Notes in Computer Science, vol. 3233, pp. 25—44. Springer Berlin Heidelberg (2004).