Advanced Topics in Concurrent Systems
DM869

Mark Jervelund
Mark@jervelund.com
Doommius.com/notes.php

UNIVERSITY OF
SOUTHERN DENMARK

IMADA

March 12, 2019

Contents

notes

w.r.t With regards to
s.t. such that

Introduction to inference systems

Rules

Theorem 1. $\frac{\text{ffl}}{num(Z)}[Zero]$

Theorem 2. $\frac{num(Z)}{num(Sx)}[Succ]$

num(Z)

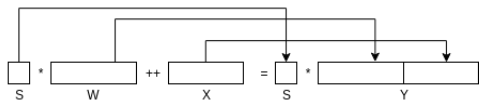
Num(Z) is derivable iff x encodes a nautral number, if any derivation for number x has exatly height n, then x encodes n
proff by induction, on the structure pf the given deviration for num(x)
Care Zero
The derivation starts with rule[Zero] hence X must be >, the height must be 1
case one
Num(Z) is derivable iff x encodes a nautral number, if any derivation for number x has exatly height n+1, then x encodes n
proff by induction, on the structure pf the given deviration for num(x)
Care Zero
The derivation starts with rule[Zero] hence X must be n+1, the height must be n+1

succ

Case Zero
The derivation rulestarts with rule[succ] hence X=Sy for somey, we have a derivation for num(Y), Its height, Say m.
By induction HP, y encodes m-1 thus Sy, encodes m-1

Add

Theorem 3. $\frac{add(w,x,y)}{add(Sw,X,Sy)}[Succ]$



add(W,X,Y) is derivable iff W+X = Y
Prove by induction on the derication of add(W,X,Y)
Case $+\mathbb{Z}$
There is no inductive step
W = Z, X = x = Y, O+X=X
Case +S W= Sw, X = x, Y = Sy,
We have a serivation for add(w,x,y)
We can apply the inductive hypothesis(ind. HP)
w+x=y, W=1+w, Y=1+y, we conclude that 1+w+x=1+y, W+X =Y

sub

Theorem 4. $\frac{num(Z)}{num(Sx)}[Succ]$

sub(w,x,y) def, rules s.t. sub(q,x,y) is derivable iff w-x=y.
It can be proved that there is no proof for this.

if then

Theorem 5. $\frac{if x+y=z}{then w-x=y}\frac{add(x,y,w)}{sub(w,x,y)}$

Conculus of the cumunication system

C is a channel
C = new channel ("IP eg 10.130.10.42")
C.open(); connect
C.send(42);
x: c.recv()
P:
Def. a labelled transition sstem is (S, L, $\rightarrow$)

- S is a set of steates (processes)
- K us a set of lables (Actions)
- $\rightarrow \subseteq S \times L \times S$ is trasition relation

Natation S $\xrightarrow{e}$ S' means (S,e, S')$\in\rightarrow$

$P ::= \emptyset$ //Termination program
$\overline{C}.P$ //send on channel c and continue as P
$\overline{C}.P$ // Recieve on channel c and continue as P

$\frac{}{C.P \xrightarrow{c} P}$ [Send]

$\frac{}{\overline{C}.P \xrightarrow{\overline{c}} P}$ [Recieve]

$\frac{P \xrightarrow{c} P' \quad Q \xrightarrow{a} Q'}{P|Q \xrightarrow{t} P'|Q'}$ [Com]

How can i see that two programs are running at the same time.
P | Q // P and Q urn concurrently
$c.P|\overline{C}.Q \rightarrow P|Q$ //If we have two nodes, c, and $\overline{c}$ and c wants to send to $\overline{c}$ and $\overline{c}$ want's to recieve from c, this is syncronys transmition. eg, a communication can't fail.
Spec R spec 1 .... Secn Rimpl
R should be an equvalence relation between two processes

- transitive: reason about chain refinemnt
- Reflexivity PRP
- Symmytry Spec R impl. -> Impl R spec

Def
A cotntect is any term generated
$C := [-]|\alpha.C|C + P|(P|C)$
$P ::= \alpha.P|O|P + P'|(P|P')$
Def
A trace is a sequence$\alpha_1.....\alpha_n \in ACT$
(ACT)* is the finite word containing all possible programs
The set of traces of a process P is the set of traces(P) $\{\alpha_1.....\alpha_n\}$

$$Traces(User) = \{\in; \overline{P}; \overline{P}.enter; \overline{P}.enter.exit; .....\} = \{\in\} \cup \{\overline{P}t| \in traces(U_1)\} \quad (1)$$

Def a Relation R is a visimulation if
Whenever two processes PRQ it holds that :
We have two conditions
if P can proform some transition

$$\text{if } p \xrightarrow{\alpha} P' \text{ then there is } Q \xrightarrow{\alpha} Q' \wedge R'RQ' \quad (2)$$

$$\text{if } Q \xrightarrow{\alpha} Q' \text{ then there is } P \xrightarrow{\alpha} P \wedge' R'RQ' \quad (3)$$

Def P,Q are called Bisimilar (P Q) iff there is a bisimulation r st. PRQ

Desiderata for behavioval eq.

- -Equvilance relation
- -Congruence (w.r.t to the syntax for out programming language)

def. A relation R over CCS processes is a bisimulation if whenever $(P,Q) \in R$ it hold that for any lable α :

Theorem 6. *1) If $p \xrightarrow{\alpha} p'$ then there is q' s.t. $Q \xrightarrow{\alpha} Q'$ and $(P',Q') \in R$*

Theorem 7. *2) symmetric of 1)*

P and Q are called bisimilar (P Q) if there a bisimulation R. S.t. $(P,Q) \in R$
Lemma: Bisimularirt is an equivilance relation
Proof

- -Reflexivity $\forall P$, P P $p \xrightarrow{\alpha} p'$, $p \xrightarrow{\alpha} p'$ - $I \subseteq$
- -Prove that I is a bisimulartion: $(P,Q) \in I$ => P=Q => $p \xrightarrow{\alpha} p'$ then $Q \xrightarrow{\alpha} p'$ $(P',P') \in I$
- -Symmetry: P Q => Q P. - By def. of bisimilairty there is a bisimulation R such that $(P,Q) \in R$ Claim $R^{-}1$ (The relation where you flip every pair) = $\{(x,y)|(y,z) \in R\}$ is a bisimulatin $(Q,P) \in R^{-}1$ is bisimular, You can conclude Q P

Given that R is a bisimulation then R^{-1} is a bisimulation for any R.
let $(s,t) \in R^{-1}$.
1) For any $S \xrightarrow{\alpha} S'$,let T' be any s.t. $(S',T') \in R$(There is at least one since R is a Bisiulation).
(Since S is in relation with T by our definition, by this point by construction.)
It follows that $T \xrightarrow{\alpha} T'$, $(S',T') \in R^{-1}$ by The above formula. this proves the first conditioning of the def. of bisumulation.
2.) Likewise: given $T \xrightarrow{\alpha} T'$, there is a transition $S \xrightarrow{\alpha} S'$ s.t. $(T',S') \in R$ hence $(T',S') \in R^{-1}$
-Transitivity: P Q, Q S => P S,
The strategy here is to show that it is similar to what we did before.
-From P Q it follows that $\exists R_1$ s.t. R_1 is a bisumulation and $(P,Q) \in R_1$
-From Q S it follows that $\exists R_2$ s.t.R_1 is a bisumulation and $(Q,S) \in R_2$
Define $R = R_1 \dot{R}_2 = \{(x,<)|\exists y.(x,y) \in R_1 \wedge (y,z) \in R_2\}$
If R is a bisimulation by constrction $(P,S) \in R$ hence P S
$(T_1,T_2) \in R$
If t1 can preform an action, then t2 can do the same with the same lables.
1) let $T_1 \xrightarrow{\alpha} T_2$, by construction of R. T_3 s.t.

$(T_1,T_3) \in R_1$, therefor T_3' s.t. $T_3 \xrightarrow{\alpha} T_3' \wedge (T_1',T_3') \in R_1$

$(T_3,T_2) \in R_2$, since R_2 is a bimiulation, $T_2 \xrightarrow{\alpha} T_2'$ s.t. $(T_3',T_2') \in R$

Inset picture here
$T_1' R_1 T_3' R_2 T_2'$ => $T_1' R T_2'$
2) $T_2 \xrightarrow{\alpha} T_2'$,
Lemma: Is a bisimulartion
See notes

Lemma: is a congruence
$\forall C, P\ Q$ => $C[P]$ $C[Q]$
The result is a result for a weaker/eqivilant one.
Proof: by structual induction
Let P Q, Proceed by strucurtal induction on C,
Base: C C, (No holes) It follow by reflexivity of
if C= [_] then C[P] = P Q = C[Q]
induction case:

- $C = \alpha.[]\alpha.P\ \alpha.Q$
- $C = S|[]S|P\ S|Q$
- $C = []|SS|P\ S|Q$
- $C = S + []$
- $C = []+S$
- $C = []\backslash L$

Lemma: If P Q then:

- 1) a.P a.Q
- 2) $\forall SS|P\ S|Q$
- 3) $\forall S + P\ S + Q$
- 1) P Q => a.P A.Q let R be a Bisimulation s.t. $(P, Q) \in R$. Def. $R' = R \cup \{(a.P, a, Q)\}$ The two conditions
  - $a.P \xrightarrow{\alpha} p$ the other process can only reply with $a.Q \wedge a.Q \xrightarrow{\alpha} Q$. $(P, Q) \in R'$,
- 2)$P\ Q => \forall S, S|P|S|Q$.Let R be a bisimulation $(P, Q) \in R$.
  -Def. R as the following relation$R = \{(S|P, S|Q)|P\ Q$, P,Q,S are CCS Processes $\}$
  Assume that $S|P \xrightarrow{\alpha} S'|P'$. Derivations for $S|P \xrightarrow{\alpha} S'|P'$ have thererfo: the first rule applied is
  1) Lpar, 2) Rpar, 3, com
- 1) Lpar

$$\frac{S \xrightarrow{\alpha} S'}{S|P \xrightarrow{\alpha} S'|P}(P = P')\frac{S \xrightarrow{\alpha} S'}{S|q \xrightarrow{\alpha} S'|Q} \tag{4}$$

$$(S'|P, S'|Q) \in R \text{ by def of R} \tag{5}$$

- 2) Rpar

$$\frac{P \xrightarrow{\alpha} P'}{S|P \xrightarrow{\alpha} S|P'}(S = S') \text{ Since } P\ Q \exists Q \xrightarrow[(*)]{\alpha} Q' P' \dagger Q' \tag{6}$$

$$\text{From}(\star)\frac{Q \xrightarrow{\alpha} Q'}{S|Q \xrightarrow{\alpha} S|Q'}\text{And from } (\dagger)(S|PTEST', S|Q') \in R \tag{7}$$

- 3) com

$$\frac{S \xrightarrow{\alpha} S'\ P \xrightarrow{\overline{\alpha}} P'}{S|P \xrightarrow{\tau} S'|P'} \text{ Since } P\ Q, Q \xrightarrow[(\star)]{\overline{\alpha}} Q'\ P' \underset{(\dagger)}{} Q' \tag{8}$$

$$\text{From } (\star)\frac{S \xrightarrow{\alpha} S'\ Q \xrightarrow{\overline{\alpha}} Q'}{S|Q \xrightarrow{\tau} S|Q'} \text{ And from } (\dagger)(S'|P', S'|Q') \in R. \tag{9}$$

Likewise $S|Q \xrightarrow{\alpha} S'|Q'$....

PFor S+P S+Q use the relation **Picture from phone**
P|Q Q|P prove that R = { (P|Q, Q|P) } is a bisimular
For an LTS (S, ACT, ->) It's saturation is the LTS S, ACT, =>) where => is the least relation s.t.
We can ignore internail computation

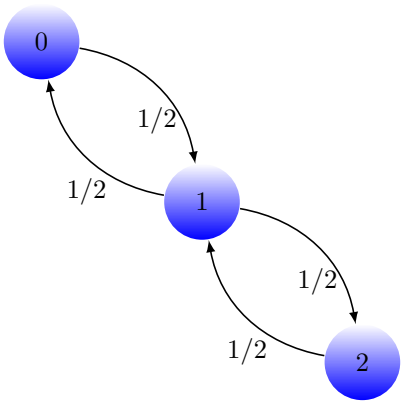
$$\frac{\text{ffl}}{S \to []\tau S}\frac{S_1 \to []\tau S_1'\ S_1' \xrightarrow{[]\alpha} S_2'\ S_2' \xrightarrow{[]\alpha S_2}}{S_1 \to []\alpha S_2} \tag{10}$$

March 4th 2019

Assignment 1

$P = \overline{in}(b).out(\neg b).p$
$Q = \overline{in\_O}.out\_1.Q + \overline{in\_1}.out\_O.Q$

- Write(d1)
- write_O
- write_1



Assignment 2

$P = \overline{inA}(x).\overline{inB}(y).out(x \wedge y).p$

March 4th 2019

Shared memory vs Message parsing.
why it's bad etc.
What's blocking, nonblocking, etc...
Detributed logic,

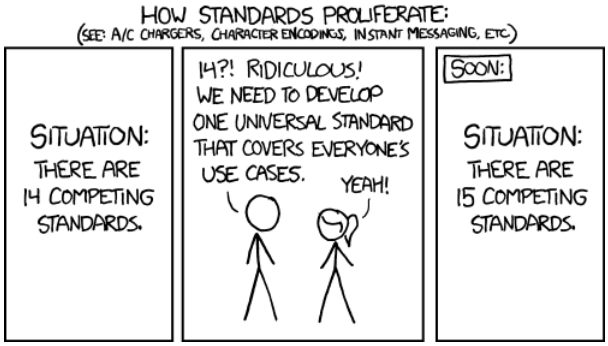
March 11th 2019

What is the general context of the paper? (try to describe both the general field (programming, distributed systems, etc.) and the specific application (functional implementations, consensus algorithms, etc.))

The problem of ditributing and replicating the and to guarenteed the data across the services.
To fill the gab between currently programming langues, designs and frameworks regarding service orientated microservies.

What are the problems the authors want to address?

"The adoption of SOC, however, has led to a problem of fragmentation."
We have a problem now where everyone has their own implementation in their own language.



Why are those problems important (impact, theoretical and/or practical needs, etc.)?

What are the main contributions of the paper?

Are there alternatives? In which way do they differ from this contribution?

Is the paper well-written, i.e., is it clear from the paper how to respond to the previous points?

Form and prepare to discuss your opinion on the paper, e.g., do you think the contributions solve the problems? To which extent (completely, what parts)? Why?

March 14th 2019 - Choreographies in practice

1) What is the general context of the paper? Try to describe both the general field (programming, distributed systems, etc.) and the specific application (functional implementations, consensus algorithms, etc.))
{distributed,programming languages, Concurrently, systems}

2) What are the problems the authors want to address?
Expressiveness of choreographic programming, The correct notation disallows mismatched I/o to communicate, which can cause problems such as deadlocks and so on..

3) Why are those problems important (impact, theoretical and/or practical needs, etc.)?
a lot of people are working on the issues within choreographic programming.
As they allow a program to be correct based on construction.

4) What are the main contributions of the paper? Describing of a design methodology that guaranteed correctness by design/construction and is able to help implement concurrent processes.

5) How do the authors substantiate their contributions? E.g., (running) examples, formalization, theorems, case studies, benchmarks, statistics, ...
They provide examples for quick sort, Gaussian elimination and fast Fourier transformation.

6) Are there alternatives? In which way do they differ from this contribution?
From what is listed in the paper this is the fisrt attempt to solve it in this way, but there has been other work ased in MPST that have explored the use of this as a protocol specification for the coordination of message exchanges in some real world scenarios.

7) Is the paper well-written, i.e., is it clear from the paper how to respond to the previous points?
It's pretty clear, i had to google some words but that was due to my inexperience in the topic.

8) What is your opinion on the paper? E.g., do you think the contributions solve the problems? To which extent (completely, what parts)? Why?
I follow and like of the ideas and solutions. I've worked with MPI and i know how prone this format can be to deadlocks and other issues with data, and communication.