Distributed Programs

Shangping Ren

Department of Computer Science
Illinois Institute of Technology

April 28, 2014
Outline

1. Basics
2. Semantics
3. Transformation into Nondeterminism
Objectives

1. Know the syntax and semantics for distributed systems.
2. Be able to define the term *channel*.
3. Know how to represent a distributed program as a nondeterministic program.
4. Be able to verify deadlock freedom of a distributed program.
Distributed Systems

- What are distributed systems?
- How do they compare to “normal” parallel programs?
- How do they compare to nondeterministic programs?
$S \equiv S_0; \text{do } \Box_j^{m} g_j \rightarrow S_j \text{ od}$

- $S_0$ is the initialization part.
- $S_j$ are nondeterministic programs.
- $g_j$ have the form $B_j; \alpha_j$.
- $B_j$ is the guard.
- $\alpha_j$ is an i/o command of the form $c ? u$ or $c ! t$.
- $c$ is a bidirectional, blocking, synchronous, typeless channel.

$\text{SENDER} \equiv i := 0; \text{do } i \neq M; \text{link ! } a[i] \rightarrow i := i + 1 \text{ od}$

$\text{RECEIVER} \equiv j := 0; \text{do } j \neq M; \text{link ? } b[j] \rightarrow j := j + 1 \text{ od}$
Two i/o commands *match* when they refer to the same channel, one is a read, the other is write, and the types agree. E.g. $c! u$ and $c? t$ match, but $d? x$ and $d? y$ don’t. Neither do $d? x$, $e! y$.

The effect of $\alpha_1 \equiv c? u$ and $\alpha_2 \equiv c! t$ is $u := t$.

Define this as $\text{Eff}(\alpha_1, \alpha_2) \equiv \text{Eff}(\alpha_2, \alpha_1) \equiv u := t$.

Processes are *disjoint* if

$$\text{change}(S_1) \cap \text{var}(S_2) = \text{var}(S_1) \cap \text{change}(S_2) = \emptyset$$

A channel *connects* $S_1$ and $S_2$ if $c \in \text{channel}(S_1) \cap \text{channel}(S_2)$. 
Distributed Programs

\[ S \equiv [S_1 \| \cdots \| S_n] \]

where \( n \geq 1 \) and for \( S_1, \ldots, S_n \) we have

- **Disjointness**: \( S_1, \ldots, S_n \) are pairwise disjoint. (why?)
- **Point to Point**: for all \( 1 \leq i < j < k \leq n \)
  
  \[ \text{channel}(S_i) \cap \text{channel}(S_j) \cap \text{channel}(S_k) = \phi \]

- Nested parallelism is not allowed.
Distributed Programs

\[ S \equiv [S_1 \parallel \cdots \parallel S_n] \]

- It terminates when all of its processes \( S_i \) terminate;
- It may fail to terminate due to
  - divergence of a process
  - abortion of a process
  - deadlock
Example — Message Passing

\[ SENDER \ a[0:M-1] \]

\[ RECEIVER \ b[0:M-1] \]

\[ input \]

\[ SR \equiv [SENDER \parallel RECEIVER] \]

\[ SENDER \equiv i := 0; \ do \ i \neq M; \ link! \ a[i] \rightarrow i := i + 1 \ od \]

\[ RECEIVER \equiv j := 0; \ do \ j \neq M; \ link? \ b[j] \rightarrow j := j + 1 \ od \]
Example — Message Passing

- **SENDER** $a[0:M-1]$ 
- **FILTER** $b[0:M-1]$ 
- **RECEIVER** $c[0:M-1]$
Example — Message Passing

\[\text{TRANS} \equiv [\text{SENDER} \parallel \text{FILTER} \parallel \text{RECEIVER}]\]

**SENDER** \( \equiv i := 0; \) \( \text{do } \) \( i \neq M; \) \( \text{input}! \ a[i] \rightarrow i := i + 1 \) \( \text{od} \)

**FILTER** \( \equiv \text{in} := 0; \) \( \text{out} := 0; \) \( x := "\ "; \)
\( \text{do } x \neq "\ "; \) \( \text{input}? \ x \rightarrow \)
\( \quad \text{if } x = "\ " \rightarrow \text{skip} \)
\( \quad \square x \neq "\ " \rightarrow b[\text{in}] := x; \) \( \text{in} := \text{in} + 1 \)
\( \quad \text{fi} \)
\( \square \text{out} \neq \text{in}; \) \( \text{output}! \ b[\text{out}] \rightarrow \text{out} := \text{out} + 1 \)
\( \text{od} \)

**RECEIVER** \( \equiv j := 0; \) \( y := "\ "; \)
\( \text{do } y \neq "\ "; \) \( \text{output}? \ y \rightarrow c[j] := y; j := j + 1 \) \( \text{od} \)
Termination

\[
< \text{do } \bigwedge_{j=1}^{m} g_j \rightarrow S_j \text{ od } , \sigma > \rightarrow < E, \sigma >
\]

where for \( j \in \{1, \ldots, m\} \) \( g_j \equiv B_j \); \( \alpha_j \) and \( \sigma \models \bigwedge_{j=1}^{m} \neg B_j \)
Effects of Communication

\[ < [S_1 \parallel \cdots \parallel S_n], \sigma > \rightarrow < [S'_1 \parallel \cdots \parallel S'_n], \tau > \]

where for some \( k, l \in \{1, \ldots, m\}, k \neq l \)

\[ S_k \equiv \text{do } \Box_{j=1}^{m_1} g_j \rightarrow R_j \text{ od} \]
\[ S_l \equiv \text{do } \Box_{j=1}^{m_2} h_j \rightarrow T_j \text{ od} \]

for some \( j_1 \in \{1, \ldots, m_1\} \) and \( j_2 \in \{1, \ldots, m_2\} \)

the guards \( g_{j_1} \equiv B_1; \alpha_1 \) and \( h_{j_2} \equiv B_2; \alpha_2 \) match, and

\( \sigma \models B_1 \land B_2 \)

\[ \mathcal{M}[\text{Eff}(\alpha_1, \alpha_2)](\sigma) = \{\tau\} \]

\( S'_i \equiv S_i \) for \( i \neq k, l \)

\( S'_k \equiv R_{j_1}; S_k, \)
\( S'_l \equiv T_{j_2}; S_l. \)
Correctness

- Partial
  \[ M[S](\sigma) = \{ \tau \mid < S, \sigma > \rightarrow^* < E, \tau > \} \]

- Weak Total
  \[ M_{w\text{tot}}[S](\sigma) = M[S](\sigma) \cup \{ \bot \mid S \text{ can diverge from } \sigma \} \]
  \[ \cup \{ \text{fail} \mid S \text{ can fail from } \sigma \} \]

- Total
  \[ M_{\text{tot}}[S](\sigma) = M_{w\text{tot}}[S](\sigma) \cup \{ \Delta \mid S \text{ can deadlock from } \sigma \} \]
Transformation to Nondeterministic Program

Suppose \( S \equiv [S_1 \parallel \cdots \parallel S_n] \), where each process \( S_i \) is of the form

\[
S_i \equiv S_{i,0}; \text{do } \square^{m_i} B_{i,j}; a_{i,j} \rightarrow S_{i,j} \text{ od}.
\]

Let \( \Gamma = \{(i, j, k, l) | a_{i,j} \text{ and } a_{k,l} \text{ match and } i < k\} \).

Then we have

\[
T(S) \equiv S_{1,0}; \cdots ; S_{n,0}; \\
\text{do } \square_{(i,j,k,l)\in \Gamma} B_{i,j} \land B_{k,l} \rightarrow \text{Eff}(\alpha_{i,j}, \alpha_{k,l}); S_{i,j}; S_{k,l} \text{ od}
\]
Transformation into Nondeterminism

TERM and BLOCK

Upon termination of $S$, we have

$$TERM \equiv \bigwedge_{i=1}^{n} \bigwedge_{j=1}^{m_i} \neg B_{i,j}$$

Upon termination of $T(S)$, we have

$$BLOCK \equiv \bigwedge_{(i,j,k,l) \in \Gamma} \neg (B_{i,j} \land B_{k,l})$$

Relation: $TERM \to BLOCK$. What about the other direction?

Deadlock states of $S$: $\sigma \models BLOCK \land \neg TERM$, $S$ is deadlock
The rest of the slides are for your info
Transformation into Nondeterminism

Rule 34: Partial Correctness

\[
\{ p \} \mathcal{S}_{1,0}; \cdots; \mathcal{S}_{n,0} \{ I \} \\
\{ I \land B_{i,j} \land B_{k,l} \} \text{Eff}(\alpha_{i,j}, \alpha_{k,l}); \mathcal{S}_{i,j}; \mathcal{S}_{k,l} \{ I \} \\
\text{for all } (i, j, k, l) \in \Gamma
\]

\[
\{ p \} \{ I \land \text{TERM} \} 
\]

Where

- Assertion \( I \) is called **global invariant** relative to \( p \);
- \( B_{i,j} \land B_{k,l} \) is referred to **Boolean condition** of the transition; and
- \( \text{Eff}(\alpha_{i,j}, \alpha_{k,l}); \mathcal{S}_{i,j}; \mathcal{S}_{k,l} \{ I \} \) is called **joint transition** (within \( \mathcal{S} \)).
Rule 35: Weak Total Correctness

\[
\begin{align*}
\{p\} S_{1,0}; \cdots ; S_{n,0}\{l\} \\
\{l \land B_{i,j} \land B_{k,l}\} \text{Eff}(\alpha_{i,j}, \alpha_{k,l}); S_{i,j}; S_{k,l}\{l\} \\
\text{for all } (i, j, k, l) \in \Gamma \\
\{l \land B_{i,j} \land B_{k,l} \land t = z\} \text{Eff}(\alpha_{i,j}, \alpha_{k,l}); S_{i,j}; S_{k,l}\{t < z\} \\
\text{for all } (i, j, k, l) \in \Gamma \\
l \land t \geq 0 \\
\end{align*}
\]

\[
\{p\} S\{l \land \text{TERM}\}
\]
Rule 36: Total Correctness

\[
\begin{align*}
\{p\} S_{1,0}; \cdots ; S_{n,0}\{l\} \\
\{l \land B_{i,j} \land B_{k,l}\} \text{Eff}(\alpha_{i,j}, \alpha_{k,l}); S_{i,j}; S_{k,l}\{l\} \\
\text{for all } (i, j, k, l) \in \Gamma \\
\{l \land B_{i,j} \land B_{k,l} \land t = z\} \text{Eff}(\alpha_{i,j}, \alpha_{k,l}); S_{i,j}; S_{k,l}\{t < z\} \\
\text{for all } (i, j, k, l) \in \Gamma \\
l \land t \geq 0 \\
l \land \text{BLOCK} \rightarrow \text{TERM} \\
\hline
\{p\} S\{l \land \text{TERM}\}
\end{align*}
\]
Example: Sender and Receiver

What are the equations for

\[ \text{SENDER} \equiv i := 0; \text{do } i \neq M; \text{link} ! \ \text{ba}[i] \rightarrow i := i + 1 \ \text{od} \]

\[ \text{RECEIVER} \equiv j := 0; \text{do } j \neq M; \text{link} ? \ b[j] \rightarrow j := j + 1 \ \text{od} \]