程序验证方法
研究生课程
Chapter 11 (11.1,11.2, 11.3 , 11.4)
Distributed Programs

朱惠彪
华东师范大学 软件工程学院

11.1 Syntax

Sequential Processes

- A (*sequential*) *process* is a statement of the form

$$S \equiv S_0;\ \mathbf{do}\ \square_{j=1}^{m} g_j \rightarrow S_j\ \mathbf{od}$$

where $m \geq 0$ and

$S_0, \ldots, S_m$ are nondeterministic programs

S_0 is the *initialization part* of S

$g_1, \ldots, g_m$ are ***generalized guards*** of the form

$$\boldsymbol{g} \equiv \boldsymbol{B};\ \boldsymbol{\alpha}$$

Boolean expression Input/output command

11.1 Syntax

- An *input command* —— $c?u$
- An *output command* —— $c!t$
  - c ———— communication channel

- channels are *undirected*; that is, they can be used to transmit values in both directions;
- channels are *untyped*; that is, they can be used to transmit values of different types.

11.1 Syntax

Definition 11.1

- ***We say that <u>two i/o commands</u> match when they refer to the same channel, say c, one of them is an input command, say c?u, and the other an output command, say c!t, such that the types of u and t agree.***
- ***We say that <u>two generalized guards</u> match if their i/o commands match.***

11.1 Syntax

- The effect of a communication between two matching i/o commands $\alpha_1 \equiv c?u$ and $\alpha_2 \equiv c!t$ is the assignment $u := t$.
- We define: Effect

$$Eff(\alpha_1, \alpha_2) \equiv Eff(\alpha_2, \alpha_1) \equiv u := t.$$

- Notation:

  channel(*S*) denote the set of channel names that appear in S.

- Processes S_1 and S_2 are called disjoint **if** the following condition holds:

$$change(S_1) \cap var(S_2) = var(S_1) \cap change(S_2) = \emptyset.$$

- We say that a channel c connects two processes S_i and S_j if

$$c \in channel(S_i) \cap channel(S_j).$$

11.1 Syntax

Distributed Programs

- ***distributed programs*** are generated by the following clause for parallel composition:

$$S ::= [S_1 \parallel \ldots \parallel S_n],$$

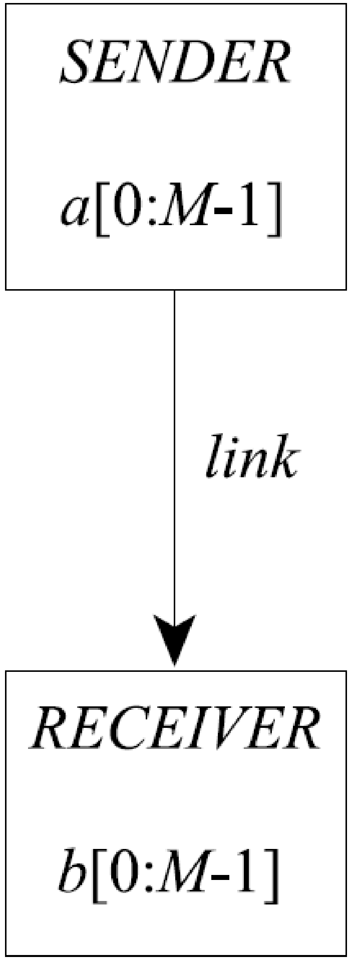
where for $n \geq 1$ and sequential processes $S_1, \ldots, S_n$ the following two conditions are satisfied:

- (i) ***Disjointness***: the processes $S_1, \ldots, S_n$ are pairwise disjoint.
- (ii) ***Point-to-Point*** *Connection*: for all i, j, k such that $1 \leq i < j < k \leq n$

$$channel(S_i) \cap channel(S_j) \cap channel(S_k) = \emptyset$$

holds.

11.1 Syntax



Example 11.1

- We now wish to write a program

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

where the process *SENDER* sends to the process *RECEIVER* a sequence of M ($M \geq 1$) characters along a channel *link*.

- $SENDER \equiv i := 0;$ **do** $i \neq M;\ \mathbf{link!a[i]} \rightarrow i := i + 1$ **od**,
- $RECEIVER \equiv j := 0;$ **do** $j \neq M;\ \mathbf{link?b[j]} \rightarrow j := j + 1$ **od**.

11.2 Semantics

(xxiv) $< \mathbf{do}\ \square_{j=1}^{m} g_j \rightarrow S_j\ \mathbf{od}, \sigma > \rightarrow < E, \sigma >$

where for $j \in \{1, \ldots, m\}$ $\boldsymbol{g_j \equiv B_j\ ;\ \alpha_j}$ and $\boldsymbol{\sigma} \models \boldsymbol{\wedge_{j=1}^{m} \neg B_j}$.

(xxv) $< [S_1 \parallel \ldots \parallel S_n], \sigma > \rightarrow < [S'_1 \parallel \ldots \parallel S'_n], \boldsymbol{\tau} >$

where for some $k, \ell \in \{1, \ldots, n\}$, $k \neq \ell$

$$\boldsymbol{S_k} \equiv \mathbf{do}\ \square_{j=1}^{m_1} g_j \rightarrow R_j\ \mathbf{od},$$

$$\boldsymbol{S_l} \equiv \mathbf{do}\ \square_{j=1}^{m_2} h_j \rightarrow T_j\ \mathbf{od},$$

for some $j_1 \in \{1, \ldots, m_1\}$ and $j_2 \in \{1, \ldots, m_2\}$ the generalized guards

$\boldsymbol{g_{j1} \equiv B_1; \alpha_1}$ and $\boldsymbol{h_{j2} \equiv B_2; \alpha_2}$ match, and

(1) $\sigma \models B_1 \wedge B_2$,

(2) $\boldsymbol{M[[Eff(\alpha_1, \alpha_2)]](\sigma) = \{\tau\}}$,

(3) $S'_i \equiv S_i$ for $i \neq k, \ell$,

(4) $S'_k \equiv \boldsymbol{R_{j1}}\ ; S_k$,

(5) $S'_l \equiv \boldsymbol{T_{j2}}\ ; S_\ell$.

11.2 Semantics

The Variants of input/output Semantics

- **partial correctness semantics:**

$$\mathcal{M}[\![S]\!](\sigma) = \{\tau \mid < S, \sigma > \to^* < E, \tau >\},$$

- **weak total correctness semantics:**

$$\mathcal{M}_{wtot}[\![S]\!](\sigma) = \mathcal{M}[\![S]\!](\sigma) \cup \{\perp \mid S \text{ can diverge from } \tau\}$$
$$\cup \{\mathbf{fail} \mid S \text{ can fail from } \tau\},$$

- **total correctness semantics:**

$$\mathcal{M}_{tot}[\![S]\!](\sigma) = \mathcal{M}_{wtot}[\![S]\!](\sigma) \cup \{\Delta \mid S \text{ can deadlock from } \sigma\}.$$

> **Here,** $\perp$ **represents divergence.**
> **fail represents failure.**
> Δ **represents deadlock.**

11.2 Semantics

Lemma 11.1. (Bounded Nondeterminism)

- *Let S be a distributed program and σ a proper state. Then M_{tot} [[S]](σ) is either* ***finite*** *or* ***it contains*** $\perp$.

Similar to the result in chapter 10 (page 353)

Lemma 10.1. (Bounded Nondeterminism) *Let S be a nondeterministic program and σ a proper state. Then $\mathcal{M}_{tot}[\![S]\!](\sigma)$ is either finite or it contains $\perp$.*

11.3 Transformation into Nondeterministic Programs

Consider a distributed program:

$$S \equiv [S_1 \parallel . . . \parallel S_n]$$

$$S_i \equiv S_{i,0};\ \mathbf{do}\,\square_{j=1}^{m_i}\ B_{i,j};\alpha_{i,j} \rightarrow S_{i,j}\ \mathbf{od}$$

Let

$$\Gamma = \{(i,j,k,\ell) \mid \alpha_{i,j} \text{ and } \alpha_{k,\ell} \text{ match and } i < k\}.$$

We transform S into the following nondeterministic program T(S):

$$\begin{aligned} T(S) \equiv\ & S_{1,0};\ \ldots;\ S_{n,0}; \\ & \mathbf{do}\ \square_{(i,j,k,\ell)\in\Gamma}\ B_{i,j} \wedge B_{k,\ell} \rightarrow\ Eff(\alpha_{i,j},\alpha_{k,\ell}); \\ & \qquad\qquad\qquad\qquad\qquad\quad\ S_{i,j};\ S_{k,\ell} \\ & \mathbf{od}, \end{aligned}$$

Upon termination of S the assertion

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

holds.

On the other hand, upon termination of T(S) the assertion holds.

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

Note:

(1) Clearly TERM →BLOCK

but not the other way round.

(2) States that satisfy BLOCK ∧ ¬TERM are deadlock states of S.

11.4 Verification

$\Gamma = \{(i, j, k, \ell) \mid \alpha_{i,j} \text{ and } \alpha_{k,\ell} \text{ match and } i < k\}.$

Page 382

$$S \equiv [S_1 \parallel \ldots \parallel S_n]$$

$$S_i \equiv S_{i,0};\ \mathbf{do}\,\square_{j=1}^{m_i}\ B_{i,j};\alpha_{i,j} \rightarrow S_{i,j}\ \mathbf{od}$$

Partial Correctness (page 390)

- RULE 34: DISTRIBUTED PROGRAMS

$$\frac{\begin{array}{l}\{\boldsymbol{p}\}\ S_{1,0};\ \ldots;\ S_{n,0}\ \{I\},\\ \{I \wedge B_{i,j} \wedge B_{k,l}\}\ \boldsymbol{Eff(\alpha_{i,j}, \alpha_{k,l})};\ S_{i,j}\ ;\ S_{k,l}\ \{I\}\\ \qquad \text{for all } (i, j, k, \ell) \in \Gamma\end{array}}{\{p\}\ S\ \{I \wedge TERM\}}$$

I —— **global invariant relative to p**

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

11.4 Verification

$$\Gamma = \{(i, j, k, \ell) \mid \alpha_{i,j} \text{ and } \alpha_{k,\ell} \text{ match and } i < k\}.$$

Page 382

$$S \equiv [S_1 \| \ldots \| S_n]$$

$$S_i \equiv S_{i,0};\ \mathbf{do}\,\square_{j=1}^{m_i}\, B_{i,j};\alpha_{i,j} \rightarrow B_{i,j}\ \mathbf{od}$$

Weak Total Correctness (page 391)

- RULE 35: DISTRIBUTED PROGRAMS II

$$\begin{array}{l}
(1)\ \{\boldsymbol{p}\}\ S_{1,0}; \ldots; S_{n,0}\ \{I\}, \\
(2)\ \{I \wedge B_{i,j} \wedge B_{k,l}\}\ Eff(\alpha_{i,j}, \alpha_{k,l});\ S_{i,j}\ ;\ S_{k,l}\ \{I\} \\
\qquad\qquad \text{for all } (i, j, k, \ell) \in \Gamma \\
(3)\ \{I \wedge B_{i,j} \wedge B_{k,l} \wedge \mathbf{t = z}\}\ Eff(\alpha_{i,j}, \alpha_{k,l});\ S_{i,j}\ ;\ S_{k,l}\ \{\mathbf{t < z}\} \\
\qquad\qquad \text{for all } (i, j, k, \ell) \in \Gamma \\
(4)\ I \rightarrow \boldsymbol{t \geq 0} \\
\hline
\{p\}\ S\ \{I \wedge TERM\}
\end{array}$$

where t is an integer expression and z is an integer variable not appearing in p, t, I or S.

$$\Gamma = \{(i,j,k,\ell) \mid \alpha_{i,j} \text{ and } \alpha_{k,\ell} \text{ match and } i < k\}.$$

Page 382

11.4 Verification

$S \equiv [S_1 \| \ldots \| S_n]$, *where* $S_i \equiv S_{i,0};\ \mathbf{do}\square_{j=1}^{m_i} B_{i,j};\alpha_{i,j} \rightarrow B_{i,j}\ \mathbf{od}$

Total Correctness (page 391)

- RULE 36: DISTRIBUTED PROGRAMS III
- (1) $\{p\}\ S_{1,0}; \ldots; S_{n,0}\ \{I\}$,

  (2) $\{I \wedge B_{i,j} \wedge B_{k,l}\}\ Eff(\alpha_{i,j}, \alpha_{k,l}); S_{i,j}; S_{k,l}\ \{I\}$

  for all $(i, j, k, \ell) \in \Gamma$

  (3) $\{I \wedge B_{i,j} \wedge B_{k,l} \wedge \mathbf{t = z}\}\ Eff(\alpha_{i,j}, \alpha_{k,l}); S_{i,j}; S_{k,l}\ \{\mathbf{t < z}\}$

  for all $(i, j, k, \ell) \in \Gamma$

  (4) $I \rightarrow t \geq 0$

  (5) $I \wedge BLOCK \rightarrow \boldsymbol{TERM}$

  $$\frac{}{\{p\}\ S\ \{I \wedge TERM\}}$$

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

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

- where t is an integer expression and z is an integer variable not appearing in p, t, I or S. The new premise (5) allows us to deduce additionally that S is deadlock free relative to p,

11.4 Verification

Proof Systems

- RULE A8:

$$\frac{I_1 \text{ and } I_2 \text{ are global invariant relative to } p}{I_1 \wedge I_2 \text{ is a global invariant relative to } p}$$

- RULE A9:

$$\frac{\begin{array}{l} I \text{ is a global invariant relative to } p, \\ \{p\}\ S\ \{q\} \end{array}}{\{p\}\ S\ \{I \wedge q\}}$$

11.4 Verification

- Proof system PDP —— *p*artial correctness of *d*istributed *p*rograms
- Proof system WDP —— *w*eak total correctness of *d*istributed *p*rograms
- Proof system TDP —— *t*otal correctness of *d*istributed *p*rograms
  - **PROOF SYSTEM *PDP*** :

    This system consists of the proof system *PN* (P357) augmented

    by the group of axioms and rules 34 (P390), A8 (P392) and A9 (P392).
  - **PROOF SYSTEM *WDP*** :

    This system consists of the proof system *TN* (P358) augmented

    by the group of axioms and rules 35 (P391) and A9 (P392).
  - **PROOF SYSTEM *TDP*** :

    This system consists of the proof system *TN* (P358) augmented

    by the group of axioms and rules 36 (P391) and A9 (P392).

11.4 Verification

Example 11.3.

- We prove the correctness of the program *SR* from Example 11.1. (P377)

$$SR \equiv [SENDER \,\|\, RECEIVER],$$

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

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

- More precisely, we prove

$$\{M \geq 1\}\ SR\ \{a[0 : M - 1] = b[0 : M - 1]\}$$

in the sense of total correctness.

$$\boxed{\forall (0 \leq k < j) : a[k] = b[k] \wedge i = j.}$$

- Global invariant relative to $M \geq 1$ we choose

$$I \equiv a[0 : i - 1] = b[0 : j - 1] \wedge 0 \leq i \leq M,$$

- One joint transition: $b[j] := a[i];\ i := i + 1;\ j := j + 1$

11.4 Verification

Example 11.3.

$I \equiv a[0:i-1] = b[0:j-1] \wedge 0 \le i \le M$

$t = M - i$

- The premises of the distributed programs III rule 36 amount to the following:

(1) $\{M \geq 1\}\ i := 0;\ j := 0\ \{I\}$,

(2) $\{I \wedge \mathrm{i} \neq M \wedge j \neq M\}\ b[j] := a[i];\ i := i+1;\ j := j+1\ \{I\}$,

(3) $\{I \wedge \mathrm{i} \neq M \wedge j \neq M \wedge \boldsymbol{t = z}\}$

$b[j] := a[i];\ i := i+1;\ j := j+1$

$\{t < z\}$,

(4) $I \rightarrow t \geq 0$,

(5) $(I \wedge \neg(i \neq M \wedge j \neq M)) \rightarrow i = M \wedge j = M$.

- All these premises can be easily verified.
- Thus, the desired correctness result can be yielded.

- RULE 36: DISTRIBUTED PROGRAMS III

(1) $\{p\}\ S_{1,0};\ldots;S_{n,0}\ \{I\}$,

(2) $\{I \wedge B_{i,j} \wedge B_{k,l}\}\ Eff(\alpha_{i,j}, \alpha_{k,l});\ S_{i,j};\ S_{k,l}\ \{I\}$

for all $(i, j, k, \ell) \in \Gamma$

(3) $\{I \wedge B_{i,j} \wedge B_{k,l} \wedge \mathrm{t} = \mathrm{z}\}\ Eff(\alpha_{i,j}, \alpha_{k,l});\ S_{i,j};\ S_{k,l}\ \{\mathrm{t} < \mathrm{z}\}$

for all $(i, j, k, \ell) \in \Gamma$

(4) $I \rightarrow t \geq 0$

(5) $I \wedge BLOCK \rightarrow TERM$

$$\frac{}{\{p\}\ S\ \{I \wedge TERM\}}$$

11.4 Verification

Soundness

- **Theorem 11.2. (Distributed Programs I)**

  The distributed programs rule 34 is sound for partial correctness.

- **Theorem 11.3. (Distributed Programs II)**

  The distributed programs II rule 35 is sound for weak total correctness.

- **Lemma 11.4. (Deadlock Freedom)**

  Assume that I is a global invariant relative to p; that is, I satisfies premises (1) and (2) above in the sense of partial correctness, and assume that premise (5) holds as well; that is, $I \wedge BLOCK \rightarrow TERM$. Then S is deadlock free relative to p.

- **Theorem 11.4. (Distributed Programs III)**

  The distributed programs III rule 36 is sound for total correctness.

11.4 Verification

Soundness

- **Theorem 11.5. (Soundness of PDP, WDP and TDP)**

  (i) The proof system PDP is sound for partial correctness of distributed programs.

  (ii) The proof system WDP is sound for weak total correctness of distributed programs.

  (iii) The proof system TDP is sound for total correctness of distributed programs.

Thanks!