程序验证方法
研究生课程
Chapter 9 (9.1-9.3)
Parallel Programs with Synchronization

朱惠彪

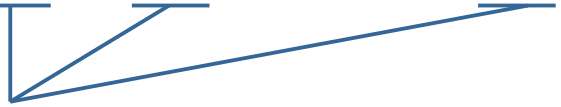
华东师范大学软件工程学院

9.1 Syntax

Syntax

Parallel programs with synchronization

$$S ::= [S_1 \parallel S_2 \parallel \ldots \parallel S_n]$$



while programs + the **await** statement

$$S ::= \textbf{await } B \textbf{ then } S_0 \textbf{ end}$$

1. loop free,
2. does not contain any await statements.

Syntax

$$S ::= \text{await } B \text{ then } S_0 \text{ end}$$

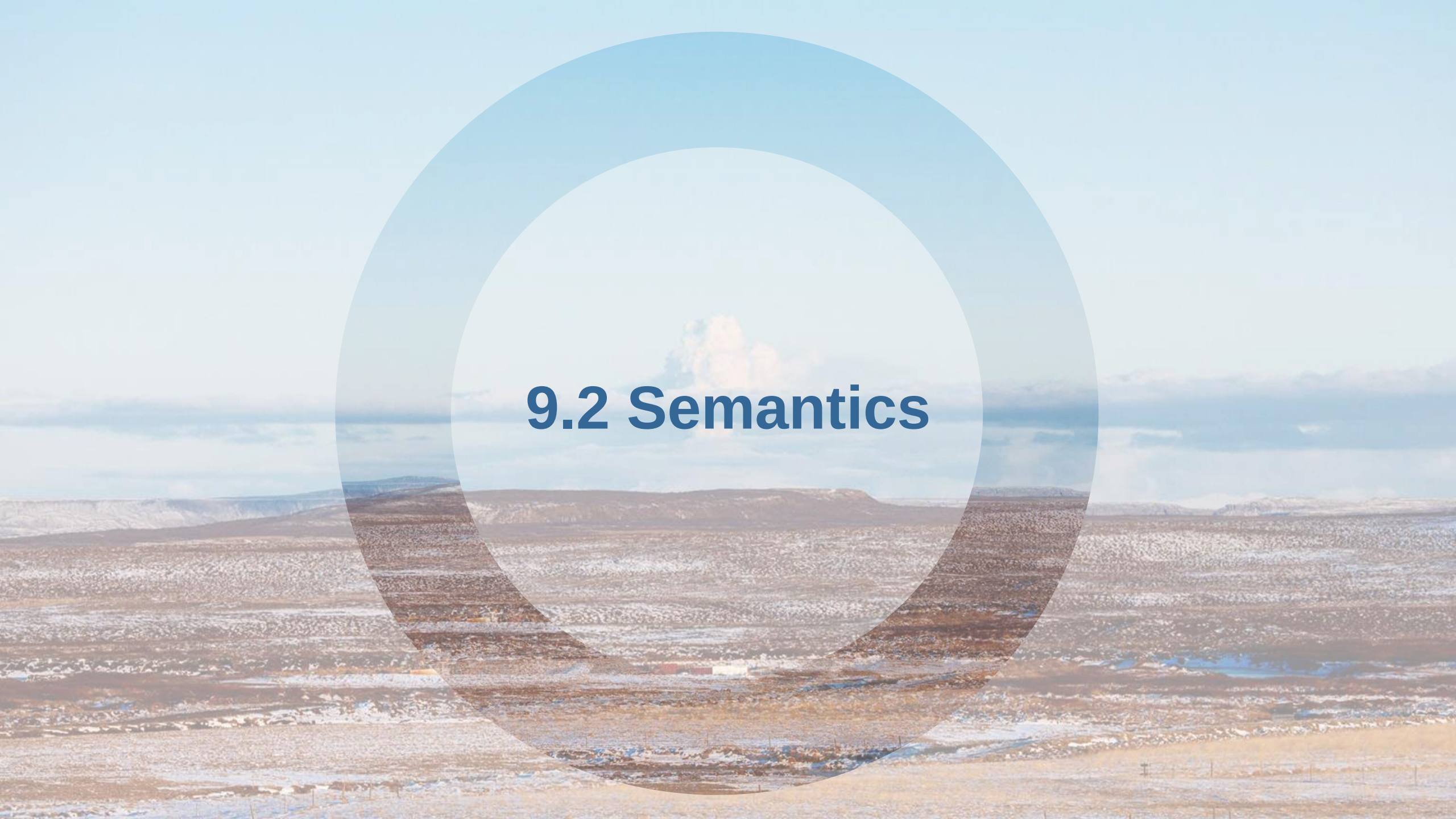
- **Meaning of await statements**

  If B evaluates to true, then S is executed as an atomic region whose activation cannot be interrupted by the other components. If B evaluates to false, the component gets blocked and the other components take over the execution. If during their execution B becomes true, the blocked component can resume its execution. Otherwise, it remains blocked forever.

- **await true then S end ≡ < S >.**

- **Abbreviation:**

$$\text{wait } B \equiv \text{await } B \text{ then } skip \text{ end}$$



9.2 Semantics

Semantics

(xix) $$\frac{< S, \sigma > \rightarrow^* < E, \tau >}{< \text{await } B \text{ then } S \text{ end}, \sigma > \rightarrow < E, \tau >} \quad \text{where } \sigma \models B.$$

𝔛 If B evaluates to **false**, the rule does **not** allow us to derive any transition for await B then S end.

Definition 9.1. Consider a parallel program *S*, a proper state σ and an assertion *p*.

(i) A configuration $< S, \sigma >$ is called *deadlock* if $S \not\equiv E$ and there is no successor configuration of $< S, \sigma >$ in the transition relation $\rightarrow$.

(ii) The program *S* can deadlock from σ if there exists a computation of *S* starting in σ and ending in a deadlock.

(iii) The program *S* is *deadlock free (relative to p)* if there is no state σ (satisfying *p*) *from which S can deadlock*.

Semantics

When started in a proper state σ, a parallel program S can now terminate, diverge or deadlock.

- **partial correctness semantics:**

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

- **weak total correctness semantics:**

$$M_{\text{wtot}}\,[[S]](\sigma) = M[[S]](\sigma) \cup \{\perp \mid S \text{ can diverge from } \sigma\},$$

- **total correctness semantics:**

$$M_{\text{tot}}\,[[S]](\sigma) = M_{\text{wtot}}\,[[S]](\sigma) \cup \{\Delta \mid S \text{ can deadlock from } \sigma\}.$$

9.3 Verification

Verification

Each of the above three variants of semantics induces in the standard way a corresponding notion of program correctness. For example, weak total correctness is defined as follows:

$$\models_{\text{wtot}} \{p\}\ S\ \{q\} \text{ iff } M_{\text{wtot}}\,[[S]]([[p]]) \subseteq [[q]]$$

$M_{\text{wtot}}\,[[S]](\sigma) = M[[S]](\sigma) \cup \{\perp \mid S \text{ can diverge from } \sigma\}$

(xix) $$\frac{< S, \sigma > \rightarrow^* < E, \tau >}{< \textbf{await } B \textbf{ then } S \textbf{ end}, \sigma > \rightarrow < E, \tau >}$$ where $\sigma \models B$.

RULE 28: SYNCHRONIZATION

$$\frac{\{p \wedge B\}\ S\ \{q\}}{\{p\}\ \textbf{await } B \textbf{ then } S \textbf{ end}\ \{q\}}$$

Note 1: The soundness of the synchronization rule is an immediate consequence of the transition rule (xix)

Proof outlines:

(xii) $$\frac{\{p \wedge B\}\ S^*\ \{q\}}{\{p\}\ \textbf{await } B \textbf{ then } \{p \wedge B\}\ S^*\ \{q\}\ \textbf{end}\ \{q\}}$$

where S^* stands for an annotated version of S.

Partial Correctness

Notions:

at(T, S) —— **introduced in Definition 3.7 (p 81)**

—— **the remainder of S that is executed when the control is at subprogram T.**

Lemma 9.1. **(Strong Soundness for Component Programs)** ***Consider a component program S with a standard proof outline*** $\{p\}\ S^*\ \{q\}$ ***for partial correctness. Suppose that***

$$< S, \sigma > \rightarrow^* < R, \tau >$$

for a proper state σ ***satisfying*** p***, a program R and a proper state*** τ ***. Then***

- ***either*** $R \equiv at(T, S)$ ***for some normal subprogram T of S and*** $\tau \models pre(T)$
- ***or*** $R \equiv E$ ***and*** $\tau \models q$**.**

Partial Correctness

Interference freedom

Standard proof outlines $\{p_i\}\ S_i^*\ \{q_i\}$, $i \in \{1, \ldots, n\}$, for partial correctness are called *interference free* if no *normal assignment* or *await statement* of a component program S_i interferes (in the sense of the previous chapter) with the proof outline of another component program S_j, $i \neq j$.

Rule 27: Please refer to the above notions of a standard proof outline and interference freedom.

RULE 27: PARALLELISM WITH SHARED VARIABLES

$$\frac{\text{The standard proof outlines } \{p_i\}\ S_i^*\ \{q_i\},\ i \in \{1, \ldots, n\},\ \text{are interference free}}{\{\bigwedge_{i=1}^{n}\ p_i\}\ [S_1 \| \ldots \| S_n]\ \{\bigwedge_{i=1}^{n}\ q_i\}}$$

Partial Correctness

Proof system PSY for partial correctness of parallel programs with synchronization:

PROOF SYSTEM PSY:

This system consists of the group of axioms and rules 1-6, 25, 27, 28 and A2-A6.

RULE 25: AUXILIARY VARIABLES

$$\frac{\{p\}\ S\ \{q\}}{\{p\}\ S_0\ \{q\}}$$

RULE 27: PARALLELISM WITH SHARED VARIABLES

$$\frac{\text{The standard proof outlines } \{p_i\}\ S_i^*\ \{q_i\},\ i \in \{1,\ldots,n\},\ \text{are interference free}}{\{\bigwedge_{i=1}^{n}\ p_i\}\ [S_1 \| \ldots \| S_n]\ \{\bigwedge_{i=1}^{n}\ q_i\}}$$

RULE 28: SYNCHRONIZATION

$$\frac{\{p \wedge B\}\ S\ \{q\}}{\{p\}\ \textbf{await}\ B\ \textbf{then}\ S\ \textbf{end}\ \{q\}}$$

Partial Correctness

Example 9.1. Prove the correctness formula in the proof system PSY.

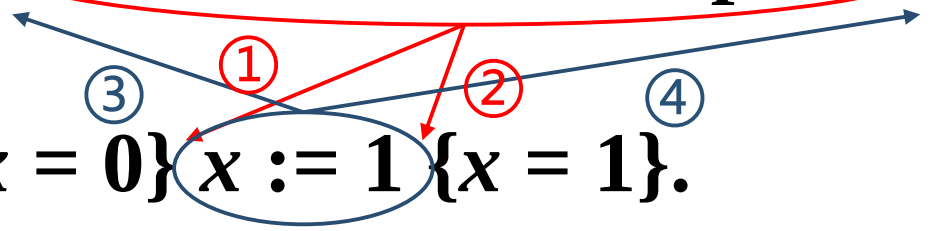
$$\{x = 0\}\ [\text{await } x = 1 \text{ then } \textit{skip} \text{ end} \parallel x := 1]\ \{x = 1\}$$

Proof. Proof outlines for the two components:

$$\{x = 0 \vee x = 1\}\ \underbrace{\text{await } x = 1 \text{ then } \textit{skip} \text{ end}}_{R}\ \{x = 1\}$$

and

$$\{x = 0\}\ \underbrace{x := 1}_{R}\ \{x = 1\}.$$

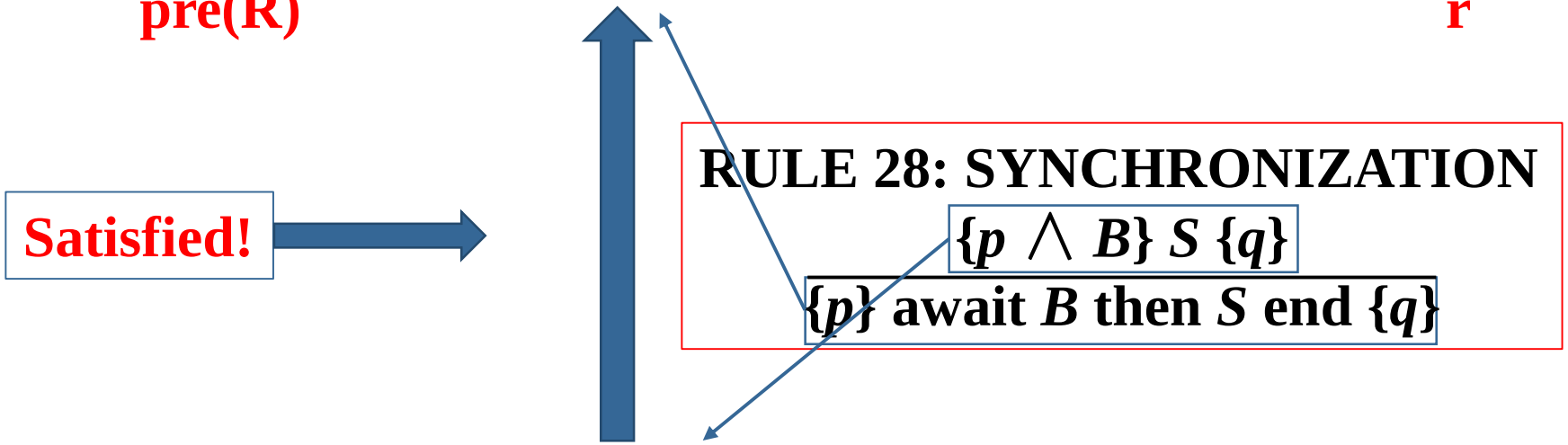


Partial Correctness

Case 1

$\{x = 0 \wedge (x = 0 \vee x = 1)\}$ await $x = 1$ then *skip* end $\{x = 0\}$

r — pre(R) — r



$\{x = 0 \wedge (x = 0 \vee x = 1) \wedge x = 1\}$ *skip* $\{x = 0\}$

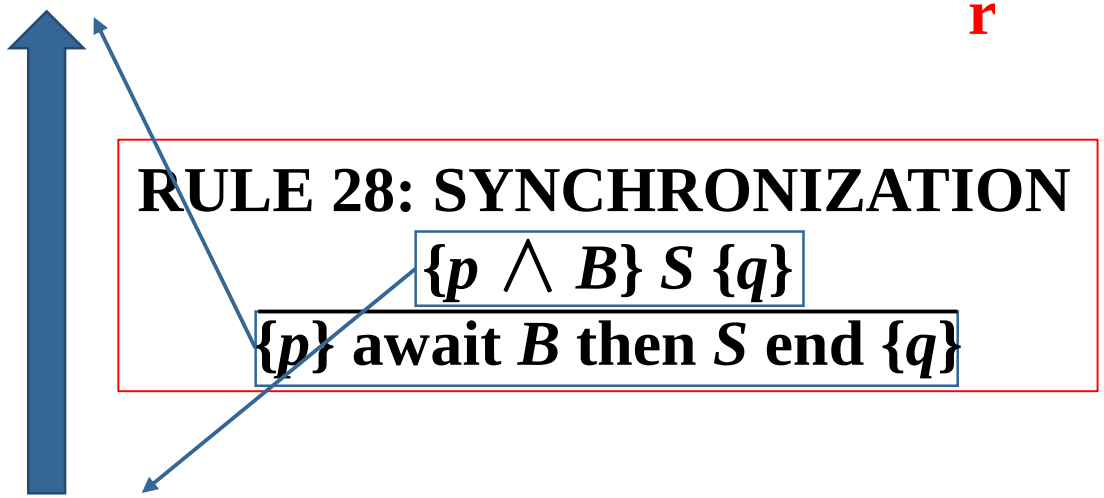
$\{false\}$

Partial Correctness

Case 2

$\{x = 1 \wedge (x = 0 \vee x = 1)\}$ await $x = 1$ then *skip* end $\{x = 1\}$

r — pre(R) — r



$\{x = 1 \wedge (x = 0 \vee x = 1) \wedge x = 1\}$ *skip* $\{x = 1\}$

$\{x = 1\}$

Satisfied! Thus rule 27 is applicable and yields the desired result.

Weak Total Correctness

—— **combines partial correctness with divergence freedom.**

Definition:

A correctness formula $\{p\}\ S\ \{q\}$ is true in the sense of weak total correctness if

$$M_{\text{wtot}}\,[[S]]([[p]]) \subseteq [[q]]$$

holds.

—— **Since $\perp \notin [[q]]$, every execution of S starting in a state satisfying p is finite and thus either terminates in a state satisfying q or gets blocked.**

Note 1: Proving weak total correctness of component programs is simple. We use the proof rules of the system TW for while programs and the synchronization rule 28 when dealing with await statements.

RULE 28: SYNCHRONIZATION

$$\frac{\{p \wedge B\}\ S\ \{q\}}{\{p\}\ \text{await}\ B\ \text{then}\ S\ \text{end}\ \{q\}}$$

Note 2: The synchronization rule is sound for weak total correctness but not for total correctness. (Reason: The execution of *await B then S end* does not terminate when started in a state satisfying $\neg B$.)

Review: Proof Outline: Total Correctness (Sect. 8.5, pp. 285)

Definition 8.3. (Proof Outline: Total Correctness) Proof outlines and standard proof outlines for the total correctness of component programs are generated by the same formation axioms and rules as those used for defining (standard) proof outlines for the partial correctness of component programs. The only exception is the formation rule (v) dealing with *while* loops which is replaced by the following formation rule.

(xi)

(1) $\{p \wedge B\}\ S^*\ \{p\}$ **is standard,**

(2) $\{pre(R) \wedge t = z\}\ R\ \{t \leq z\}$ **for every normal assignment and atomic region** R **within** S**,**

(3) for each path $\pi \in path(S)$ **there exists a normal assignment or atomic region** R **in** π **such that** $\{pre(R) \wedge t = z\}\ R\ \{t < z\}$**,**

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

$\{\text{inv} : p\}\{\text{bd} : t\}$ **while** B **do** $\{p \wedge B\}\ S^*\ \{p\}$ **od** $\{p \wedge \neg B\}$

where t **is an integer expression and** z **is an integer variable not occurring in** p, t, B **or** S^***, and where** $pre(R)$ **stands for the assertion preceding** R **in the standard proof outline** $\{p \wedge B\}\ S^*\ \{p\}$ **for total correctness.**

Weak Total Correctness

Proof outlines for weak total correctness:

First we must ensure that await statements decrease or leave unchanged the bound functions of while loops. To this end, we adapt
P285 **Definition 8.2 of the set *path*(*S*) for a component program *S* by replacing the clause *path*(< *S* >) = {< *S* >} with**

- ***path*(*await B then S end*) = {*await B then S end*}.**

With this change, (standard) proof outlines for weak total correctness of component programs are defined by the same rules as those used for (standard) proof outlines for total correctness in Definition 8.3 together with rule (xii) dealing with await statements. P311

P285 (xii)

$$\frac{\{p \wedge B\}\ S^*\ \{q\}}{\{p\}\ \text{await}\ B\ \text{then}\ \{p \wedge B\}\ S^*\ \{q\}\ \text{end}\ \{q\}}$$

Weak Total Correctness

Interference freedom

Standard proof outlines $\{p_i\}\ S_i^*\ \{q_i\}$, $i \in \{1, \ldots, n\}$, for weak total correctness are called *interference free* if no normal assignment or await statement of a component program S_i interferes with the proof outline of another component program S_j , $i \neq j$.

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

Total Correctness

- Proving **total correctness** is more complicated than in Chapter 8.

  Reason 1: In the presence of await statements program termination **not only requires divergence freedom**.

  Reason 2: Deadlock freedom is a *global* property that can be proved only by examining all components of a parallel program together.

- To prove **total correctness** of a parallel program, we first prove **weak total correctness** of its components, and then **establish deadlock freedom**.

- To prove **deadlock freedom** of a parallel program, we examine interference free standard proof outlines for weak total correctness of its component programs and use the following method:

  1. Enumerate all ***potential* deadlock situations**.
  2. Show that **none of them can actually occur**.

- Every deadlock is also a potential deadlock. (page 315)

Total Correctness

Potential deadlock

Definition 9.2. Consider a parallel program $S \equiv [S_1 \parallel \ldots \parallel S_n]$.

(i) A tuple $(R_1, \ldots, R_n)$ of statements is called a *potential deadlock* of S if the following two conditions hold:

- For every $i \in \{1, \ldots, n\}$, R_i is either an await statement in the component S_i or the symbol E which stands for the empty statement and represents termination of S_i,
- for some $i \in \{1, \ldots, n\}$, R_i is an await statement in S_i.

(ii) Given interference free standard proof outlines $\{p_i\}\ S_i^*\ \{q_i\}$ for weak total correctness, $i \in \{1, \ldots, n\}$, we associate with every potential deadlock of S a corresponding tuple $(r_1, \ldots, r_n)$ of assertions by putting for $i \in \{1, \ldots, n\}$:

- $r_i \equiv pre(R_i) \wedge \neg B$ if $R_i \equiv$ await B then S end,
- $r_i \equiv q_i$ if $R_i \equiv E$.

Total Correctness

RULE 29: PARALLELISM WITH DEADLOCK FREEDOM

(1) The standard proof outlines $\{p_i\}\ S_i^*\ \{q_i\}$, $i \in \{1, \ldots, n\}$ for weak total correctness are interference free,

(2) For every potential deadlock $(R_1, \ldots, R_n)$ of $[S_1 \parallel \ldots \parallel S_n]$ the corresponding tuple of assertions $(r_1, \ldots, r_n)$ satisfies $\neg \bigwedge_{i=1}^{n} r_i$.

$$\{\bigwedge_{i=1}^{n} p_i\}\ [S_1 \parallel \ldots \parallel S_n]\ \{\bigwedge_{i=1}^{n} q_i\}$$

Note: If we can show $\neg \bigwedge_{i=1}^{n} r_i$ for every such tuple $(r_1, \ldots, r_n)$ of assertions, **none of the potential deadlocks can arise**.

Total Correctness

To prove total correctness of parallel programs with synchronization, we use the following proof system *TSY* :

PROOF SYSTEM TSY:

This system consists of the group of axioms and rules 1-5, 7, 25, 28, 29 and A2-A6.

RULE 25: AUXILIARY VARIABLES

$$\frac{\{p\}\ S\ \{q\}}{\{p\}\ S_0\ \{q\}}$$

RULE 28: SYNCHRONIZATION

$$\frac{\{p \wedge B\}\ S\ \{q\}}{\{p\}\ \textbf{await}\ B\ \textbf{then}\ S\ \textbf{end}\ \{q\}}$$

Total Correctness

Example 9.2. We now wish to prove the correctness formula

$$\{x = 0\}\ [\text{await } x = 1 \text{ then } skip \text{ end} \parallel x := 1]\ \{x = 1\} \qquad (9.1)$$

of Example 9.1 in the proof system *TSY*.

Proof.

Interference Freedom for Weak Total Correctness

$$\{x = 0 \vee x = 1\}\ \text{await } x = 1 \text{ then } skip \text{ end}\ \{x = 1\} \qquad (9.2)$$

and

$$\{x = 0\}\ x := 1\ \{x = 1\}.$$

RULE 28: SYNCHRONIZATION

$$\frac{\{p \wedge B\}\ S\ \{q\}}{\{p\}\ \text{await } B \text{ then } S \text{ end}\ \{q\}}$$

Total Correctness

Example 9.2. We now wish to prove the correctness formula

$$\{x = 0\}\ [\text{await } x = 1 \text{ then } \textit{skip} \text{ end} \parallel x := 1]\ \{x = 1\} \qquad (9.1)$$

of Example 9.1 in the proof system *TSY*.

Proof.

Deadlock Freedom

The only potential deadlock is

$$(\text{await } x = 1 \text{ then } \textit{skip} \text{ end}, E)$$

The corresponding pair of assertions is

$$((x = 0 \vee x = 1) \wedge x \neq 1, x = 1)$$

Applying Rule 29: Yielding (9.1) as desired.

Soundness for PSY

Lemma 9.2. **(Auxiliary Variables)** ***The auxiliary variables rule 25 is sound for partial (and total) correctness of parallel programs with synchronization.***

Corollary 9.1. **(Parallelism)** ***The parallelism with shared variables rule 27 is sound for partial correctness of parallel programs with synchronization.***

Corollary 9.2. **(Soundness of PSY)** ***The proof system PSY is sound for partial correctness of parallel programs with synchronization.***

Soundness for TSY

Lemma 9.4. (Divergence Freedom) *Let* $\{p_i\}\ S_i^*\ \{q_i\}$, $i \in \{1, \ldots, n\}$, *be interference free standard proof outlines for weak total correctness for component programs* S_i. *Then*

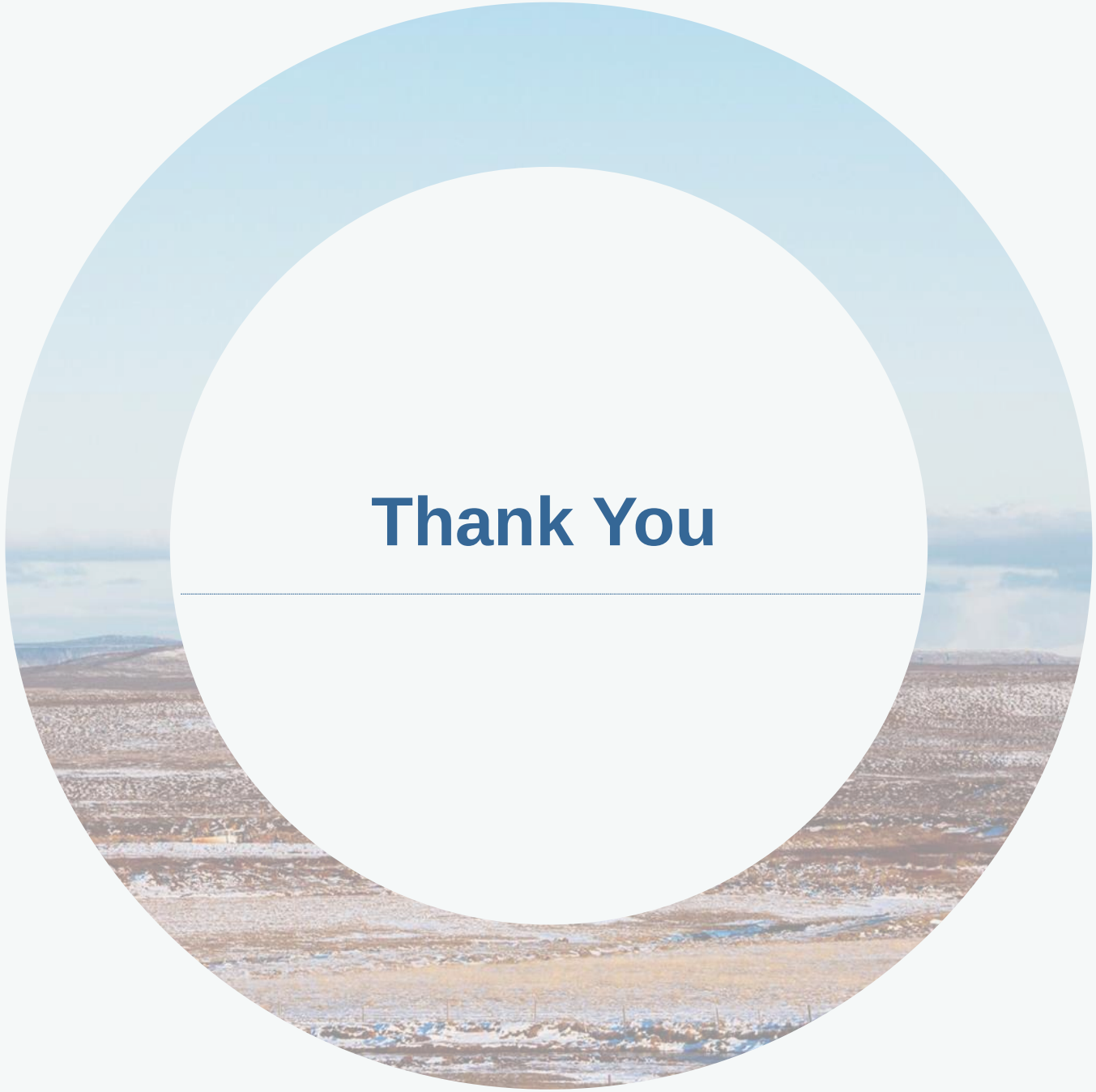
$$\bot \notin \mathit{Mtot}[[\,[S_1 \,\|\, \ldots \,\|\, S_n]\,]]([[\wedge_{i=1}^{n} p_i]])$$

Lemma 9.5. (Deadlock Freedom) *Let* $\{p_i\}\ S_i^*\ \{q_i\}$, $i \in \{1, \ldots, n\}$, *be interference free standard proof outlines for partial correctness for component programs* S_i. *Suppose that for every potential deadlock* $(R_1, \ldots, R_n)$ *of* $[S_1 \,\|\, \ldots \,\|\, S_n]$ *the corresponding tuple of assertions* $(r_1, \ldots, r_n)$ *satisfies* $\neg \wedge_{i=1}^{n} r_i$. *Then*

$$\Delta \notin \mathit{Mtot}[[\,[S_1 \,\|\, \ldots \,\|\, S_n]\,]]([[\wedge_{i=1}^{n} p_i]])$$

Corollary 9.3. (Parallelism) *Rule 29 is sound for total correctness of parallel programs with synchronization.*

Corollary 9.4. (Soundness of TSY) *The proof system TSY is sound for total correctness of parallel programs with synchronization.*



Thank You