

Operational Semantics

- Labelled transition systems (LTD)
- Bisimulation
- Firing rules for CSP
- Model-Checker FDR
- Examples

Semantic approaches to CSP

Operational semantics interprets CSP processes as *transition diagrams*, with visible and invisible actions for moving between various program states.

Denotational semantics maps CSP into some abstract models: *traces*, *failures* and *failures/divergences*.

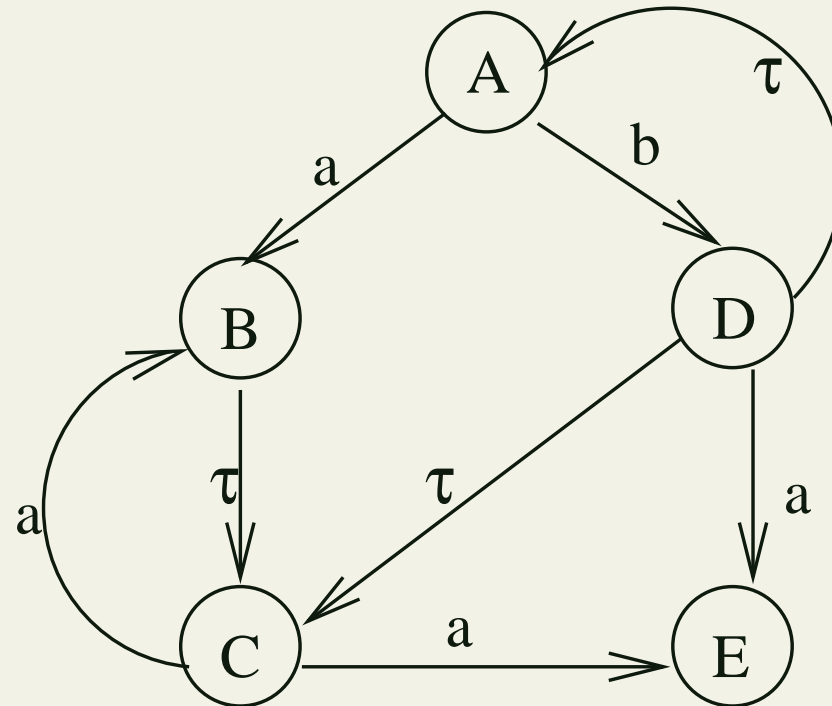
Algebraic semantics is defined by a set of *algebraic laws*, from which process equivalence between CSP processes can be derived.

Labelled transition systems (LTD)

- a set of nodes as process states
- a starting node n_0
- for each $a \in \Sigma^{\checkmark, \tau}$, a relation \xrightarrow{a} between nodes, where

$$\Sigma^{\checkmark, \tau} = \Sigma \cup \{\checkmark, \tau\}$$

Σ is the alphabet of all communications of a process



Beispiel 1 (A labelled transition system)

Bisimulation

Definition 1 (Strong Bisimulation)

IF S is an LTS, the relation R on the set of nodes S' of S is said to be a *(strong) bisimulation* if, and only if, both the following hold:

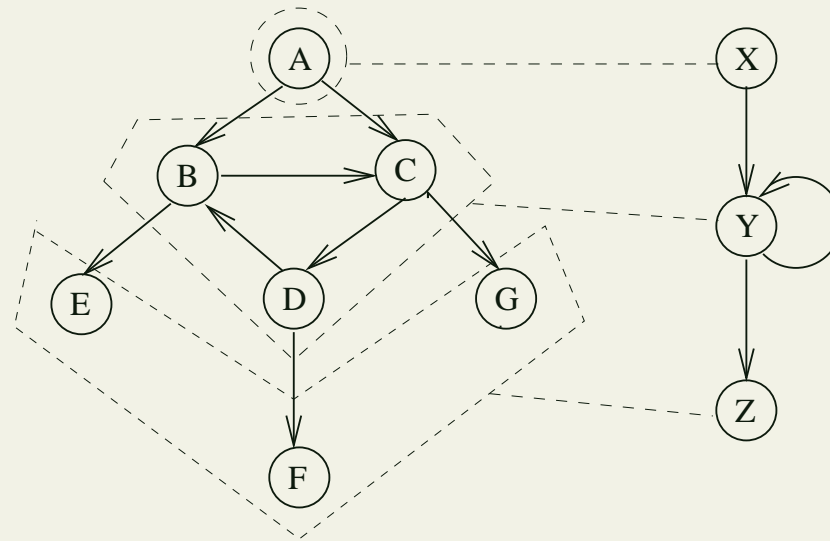
$$\forall n_1, n_2, m_1 \in S'. \forall x \in \Sigma^{\checkmark, \tau}.$$

$$n_1 R n_2 \wedge n_1 \xrightarrow{x} m_1 \Rightarrow \exists m_2 \in S'. n_2 \xrightarrow{x} m_2 \wedge m_1 R m_2$$

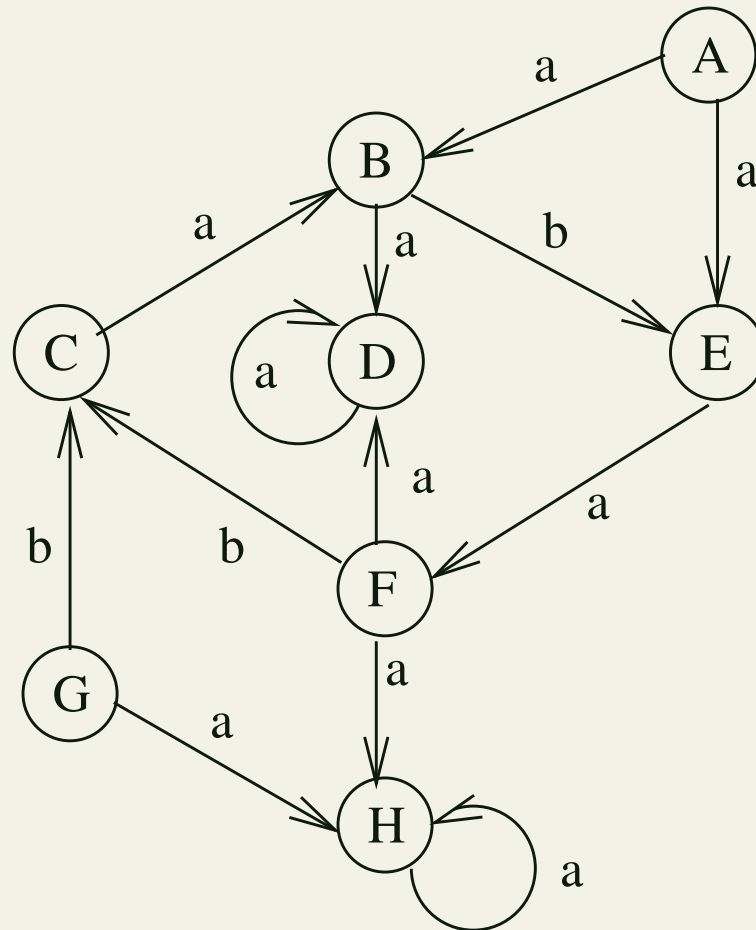
$$\forall n_1, n_2, m_2 \in S'. \forall x \in \Sigma^{\checkmark, \tau}.$$

$$n_1 R n_2 \wedge n_2 \xrightarrow{x} m_2 \Rightarrow \exists m_1 \in S'. n_1 \xrightarrow{x} m_1 \wedge m_1 R m_2$$

Two nodes in S' are said to be *bisimilar* if there is any bisimulation which relates them.



Beispiel 2 (Bisimulation equivalence)



Beispiel 3 (Which nodes are bisimilar?)

Firing rules for CSP

- Fundamental operators

$$\frac{}{Skip \overset{\checkmark}{\longrightarrow} \Omega} \quad \Omega \text{ denotes any terminated process}$$

$$\frac{}{e \rightarrow P \overset{a}{\longrightarrow} \mathit{subs}(a, e, P)} \quad (a \in \mathit{comms}(e))$$

where,

$\mathit{comms}(e)$ is the set of communication described by e
 $\mathit{subs}(a, e, P)$ is the result of substituting the part of a
 for each identifier in P bound by e

$$\overline{\mu p.P \xrightarrow{\tau} P[\mu p.P/p]}$$

$$\overline{P \sqcap Q \xrightarrow{\tau} P}$$

$$\overline{P \sqcap Q \xrightarrow{\tau} Q}$$

$$\frac{P \xrightarrow{\tau} P'}{P \sqcap Q \xrightarrow{\tau} P' \sqcap Q}$$

$$\frac{Q \xrightarrow{\tau} Q'}{P \sqcap Q \xrightarrow{\tau} P \sqcap Q'}$$

$$\frac{P \xrightarrow{a} P'}{P \sqcap Q \xrightarrow{a} P'} \quad (a \neq \tau)$$

$$\frac{Q \xrightarrow{a} Q'}{P \sqcap Q \xrightarrow{a} Q'} \quad (a \neq \tau)$$

Beispiel 4 ($a \rightarrow P \sqcap b \rightarrow Q$ and $a \rightarrow P \sqcap b \rightarrow Q$)

- Parallel operators

$$\frac{P \xrightarrow{\tau} P'}{P \parallel_X Q \xrightarrow{\tau} P' \parallel_X Q}$$

$$\frac{Q \xrightarrow{\tau} Q'}{P \parallel_X Q \xrightarrow{\tau} P \parallel_X Q'}$$

$$\frac{P \xrightarrow{a} P'}{P \parallel_X Q \xrightarrow{a} P' \parallel_X Q} \quad (a \in \Sigma \setminus X)$$

$$\frac{Q \xrightarrow{a} Q'}{P \parallel_X Q \xrightarrow{a} P \parallel_X Q'} \quad (a \in \Sigma \setminus X)$$

$$\frac{P \xrightarrow{a} P' \quad Q \xrightarrow{a} Q'}{P \parallel_X Q \xrightarrow{a} P' \parallel_X Q'} \quad (a \in X)$$

$$\frac{P \xrightarrow{\surd} P'}{P \parallel_X Q \xrightarrow{\tau} \Omega \parallel_X Q}$$

$$\frac{Q \xrightarrow{\surd} Q'}{P \parallel_X Q \xrightarrow{\tau} P \parallel_X \Omega}$$

$$\frac{}{\Omega \parallel_X \Omega \xrightarrow{\surd} \Omega}$$

Exercise: Derive the transitions of the process

$$\text{SVAR}(2) \quad \parallel \quad \text{USER} \\ \{read, write\}$$

$$\text{SVAR}(x) = (read.x \rightarrow \text{SVAR}(x))$$

$$\square (write?y \rightarrow \text{SVAR}(y))$$

$$\text{USER} = read?x \rightarrow write.x * x \rightarrow \text{USER}$$

- Hiding and renaming

$$\frac{P \xrightarrow{x} P'}{P \setminus B \xrightarrow{x} P' \setminus B} \quad (x \notin B \cup \{\checkmark\})$$

$$\frac{P \xrightarrow{\checkmark} P'}{P \setminus B \xrightarrow{\checkmark} \Omega}$$

$$\frac{P \xrightarrow{a} P'}{P \setminus B \xrightarrow{\tau} P' \setminus B} \quad (a \in B)$$

$$\frac{P \xrightarrow{\tau} P'}{P \parallel R \parallel \xrightarrow{\tau} P' \parallel R \parallel}$$

$$\frac{P \xrightarrow{\checkmark} P'}{P \parallel R \parallel \xrightarrow{\checkmark} \Omega}$$

$$\frac{P \xrightarrow{a} P'}{P \parallel R \parallel \xrightarrow{b} P' \parallel R \parallel} \quad (a R b)$$

- Sequential composition

$$\frac{P \xrightarrow{x} P'}{P; Q \xrightarrow{x} P'; Q} \quad (x \neq \checkmark)$$

$$\frac{\exists P'. P \xrightarrow{\checkmark} P'}{P; Q \xrightarrow{\tau} Q}$$

The Model-Checker FDR

- The basic concept of FDR

FDR (Failures-Divergence-Refinement) is a model-checking tool for labelled transition systems with foundations in CSP based mainly on explicit model-checking techniques.

Explicit model-checking techniques: the check proceeds by a recursive induction which fully expands the reachable state-space of the two systems and visits each pair of supposedly corresponding state in turn.

- The basic operation of FDR
 - Calculate the operational semantics using firing rules.
 - Normalise the specification process into a LTS in which all states are semantically distinct based mainly on the strong bisimulation.
 - Check process properties in terms of refinement relations.

- A concurrent system is **deadlocked** if no component can make any progress, generally because each is waiting for communication with others.
- A concurrent system can **livelock**, when a network communicates infinitely internally without any component communicating externally.

Examples

Start FDR2

- Five dining philosophers (deadlock check)

phil.fdr2

- Producer-consumer system with hiding (livelock check)

pc_sys.fdr2

- Buffer (refinement check)

buff_spec.fdr2

Denotational Semantics

A Short Introduction

The traces model (\mathcal{T})

- Definitions

A **trace** of process P is a sequence of communications between the environment and P .

For any process P , **traces**(P) is defined to be the set of all its finite traces, i.e., members of Σ^* .

The set of all non-empty, prefix-closed subsets of Σ^* is called **traces model** (\mathcal{T}).

Refinements

- Definition

Process Q **refines** process P (written $P \sqsubseteq_T Q$) if $traces(Q) \subseteq traces(P)$.

- Beispiel 5 (A change-giving machine)

A change-give machine which takes in 1 Euro coins and gives changes in 10, 20, 50 cent coins. It should have the following events:

$$\{in, out.10, out.20, out.50\}$$

Some specifications

$$CHANGE1 = in \rightarrow out.50 \rightarrow out.50 \rightarrow CHANGE1$$
$$CHANGE2 = in \rightarrow out.50 \rightarrow out.20 \rightarrow \\ out.20 \rightarrow out.10 \rightarrow CHANGE2$$
$$CHANGE3 = in \rightarrow out.50 \rightarrow out.20 \rightarrow \\ out.10 \rightarrow out.10 \rightarrow \\ out.10 \rightarrow CHANGE3$$

...

A more general specification

$$CHANGE = in \rightarrow CH_OUT(100)$$

$$CH_OUT(x) = \mathbf{if} (x == 0) \mathbf{then} CHANGE \\ \mathbf{if} (x \geq 50) \mathbf{then} CH_OUT1(x) \\ \mathbf{if} (50 > x \geq 20) \mathbf{then} CH_OUT2(x) \\ \mathbf{else} out.10 \rightarrow CH_OUT(x - 10)$$

$$CH_OUT1(x) = out.10 \rightarrow CH_OUT(x - 10)$$

$$\sqcap out.20 \rightarrow CH_OUT(x - 20)$$

$$\sqcap out.50 \rightarrow CH_OUT(x - 50)$$

$$CH_OUT2(x) = out.10 \rightarrow CH_OUT(x - 10)$$

$$\sqcap out.20 \rightarrow CH_OUT(x - 20)$$

- Some advantages of refinement relations

- Mechanical verification with tools like FDR.
- Refinement has many properties that can be exploited, for example

$$P \sqsubseteq Q \wedge Q \sqsubseteq R \Leftrightarrow P \sqsubseteq R$$

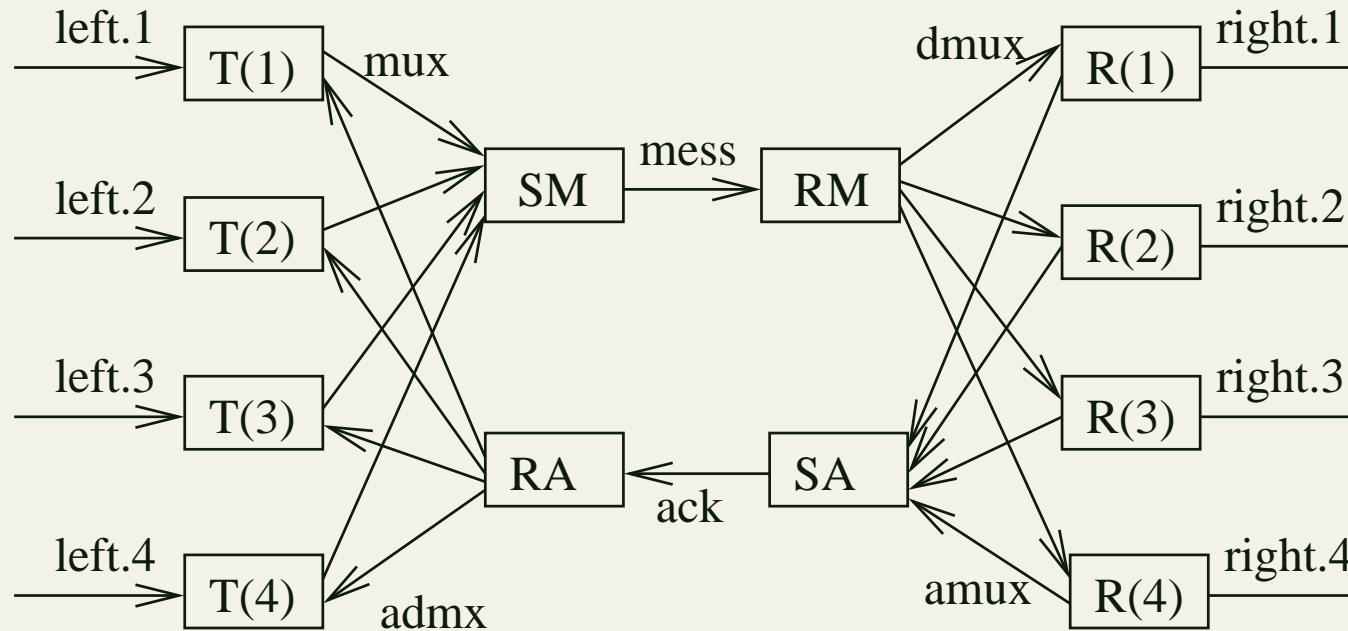
$$P \sqsubseteq Q \wedge Q \sqsubseteq P \Leftrightarrow P = Q$$

- **Compositionality**, that is

$$P \sqsubseteq Q \Rightarrow C[P] \sqsubseteq C[Q].$$

- **Stepwise refinement**

$$Spec \sqsubseteq P_1 \sqsubseteq \dots \sqsubseteq P_n \sqsubseteq Impl$$



Beispiel 6 (Multiplexed buffer)

The implementation

$$\text{LHS} = ((\parallel i : \{1, 2, 3, 4\} @ \mathbf{T}(i)) \parallel (\mathbf{SM} \parallel \mathbf{RA})) \setminus X$$

$$\text{RHS} = ((\parallel i : \{1, 2, 3, 4\} @ \mathbf{R}(i)) \parallel (\mathbf{RM} \parallel \mathbf{SA})) \setminus Y$$

$$\text{SYS} = (\text{LHS} \parallel \text{RHS}) \setminus Z$$

$$X = \{mux, admux\} \quad Y = \{amux, dmux\} \quad Z = \{mess, ack\}$$

$$\begin{aligned}T(i) &= \textit{left}.i?x \rightarrow \textit{mux}.i.x \rightarrow \textit{admx}.i \rightarrow T(i) \\R(i) &= \textit{dmux}.i?x \rightarrow \textit{right}.i.x \rightarrow \textit{amux}.i \rightarrow R(i) \\SM &= \textit{mux}?i.x \rightarrow \textit{mess}.i.x \rightarrow SM \\RM &= \textit{mess}?i.x \rightarrow \textit{dmux}.i.x \rightarrow RM \\SA &= \textit{amux}?i \rightarrow \textit{ack}.i \rightarrow SA \\RA &= \textit{ack}?i \rightarrow \textit{admx}.i \rightarrow RA\end{aligned}$$

The specification



$\text{COPY}(i) = \text{left.i?}x \rightarrow \text{right.i.x} \rightarrow \text{COPY}(i)$

$\text{SPEC} = ||| i : \{1, 2, 3, 4\} @ \text{COPY}(i)$

multiplex.fdr2

Start FDR2

The failures model (\mathcal{F})

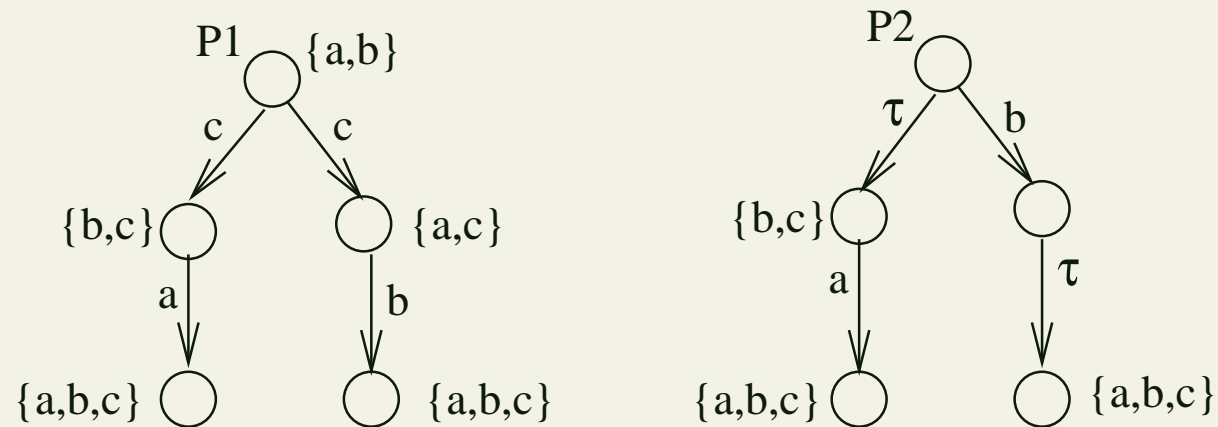
- Definitions

$$\mathit{refusals}(P) = \{a \mid \nexists s \in \Sigma^*. \langle a \rangle \hat{\ } s \in \mathit{traces}(P)\}$$

$\mathit{failures}(P)$ consists of all pairs of (s, X) where s is a trace of P and X a set of actions P can refuse in some state after s .

$$\mathit{failures}(P) = \{(s, X) \mid s \in \mathit{traces}(P) \wedge X \subseteq \mathit{refusals}(P/s)\}$$

- An example



$$P_1 = (c \rightarrow a \rightarrow Stop) \square (c \rightarrow b \rightarrow Stop)$$

$$P_2 = ((c \rightarrow a \rightarrow Stop) \square (b \rightarrow c \rightarrow Stop)) \setminus \{c\}$$

- Failures-Refinements

$$P \sqsubseteq_{\mathcal{F}} Q, \text{ iff } failures(Q) \subseteq failures(P)$$