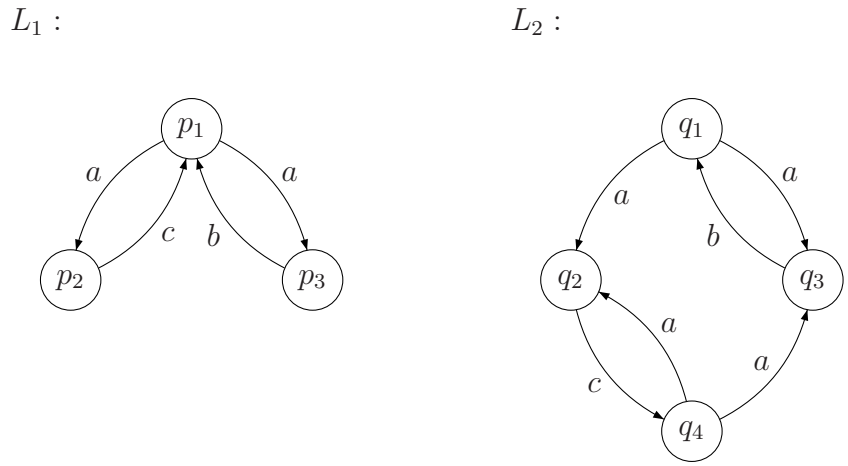


Figure 2.1: Example 2.2.4: Transition Systems

To prove this, we have to check that for all transitions in L_1, L_2 the conditions of Definition 2.2.3 are fulfilled. This is demonstrated in Figure 2.2. There, read $p_i \cdots\cdots q_j$ as $p_i R q_j$.

■ END EXAMPLE

End of Lecture #2

2.2.5 Lemma (Lifting bisimulation to $\xrightarrow{\sigma}$) *R is a bisimulation iff, for $\sigma \in Act_\tau^*$:*

- if pRq and $p \xrightarrow{\sigma} p'$, then $\exists q' \in \mathbb{P} : q \xrightarrow{\sigma} q'$ and $p'Rq'$.
- if pRq and $q \xrightarrow{\sigma} q'$, then $\exists p' \in \mathbb{P} : p \xrightarrow{\sigma} p'$ and $p'Rq'$.

Proof: The "$\Leftarrow$" direction is straightforward: since the two conditions hold not only for all $\sigma \in Act_\tau^*$, but especially for those with length 1, R must be a bisimulation.
We show the "$\Rightarrow$" direction with induction over the length of σ.
We want to show that

$$\text{if } p \xrightarrow{\sigma} p' \text{ then } \exists q' : q \xrightarrow{\sigma} q' \text{ and } p'Rq'. \quad (*)$$

Let R be a bisimulation and pRq.
The induction start, *i.e.,* the case $\sigma = \varepsilon$, is trivial. Naturally, also the case $|\sigma| = 1$ is fulfilled by definition.
For the induction step we assume that we have shown for all $\sigma \in Act_\tau^*$ with $|\sigma| \leq n$ that (*) holds. We assume a word $\sigma \cdot a$ with $|\sigma \cdot a| = n + 1$, and that $p \xrightarrow{\sigma \cdot a} p'$ Thus, by hypotheses, there is a p'' such that $p \xrightarrow{\sigma} p''$ and $p'' \xrightarrow{a} p'$, and a q'' such that $q \xrightarrow{\sigma} q''$ and $p''Rq''$. Since $p'' \xrightarrow{a} p'$ and $p''Rq''$, then there must be a q' such that $q'' \xrightarrow{a} q'$ and $p'Rq'$. Thus, for $p \xrightarrow{\sigma \cdot a} p'$, there is a q' with $q \xrightarrow{\sigma \cdot a} q'$ and $p'Rq'$.

Note: All material on the slides used in the lecture is in the script.

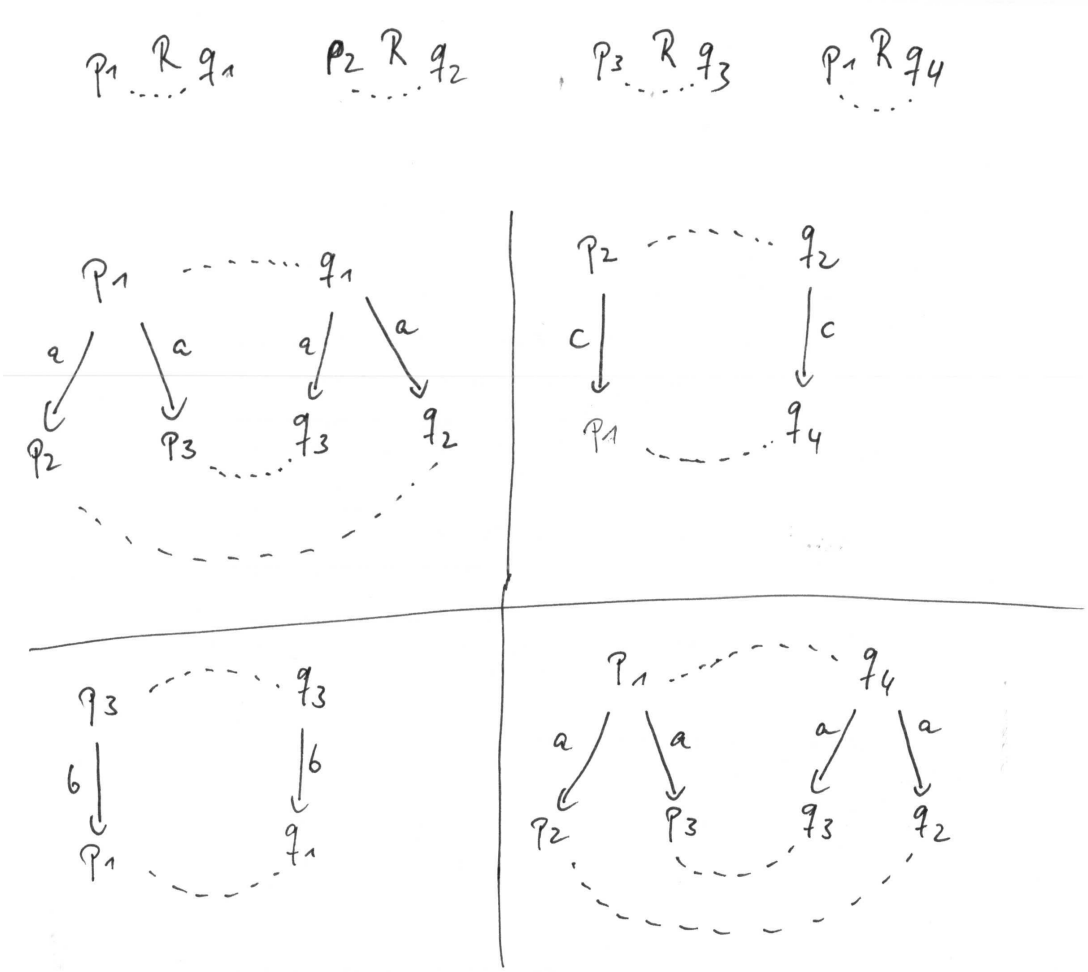


Figure 2.2: Example 2.2.4: R is a Bisimulation

Note: All material on the slides used in the lecture is in the script.

The second statement works the same. ■ END PROOF

End of Lecture #3

2.2.6 Example (Generalised bisimilarity)
Recall Example 2.2.4. q_1 **after** $aca = \{q_2, q_3\}$, and the (derived) transition $q_1 \stackrel{aca}{\Longrightarrow} q_2$ in L_2 corresponds to $p_1 \stackrel{aca}{\Longrightarrow} p_2$, and $p_2 R q_2$. Similarly, $q_1 \stackrel{aca}{\Longrightarrow} q_3$ corresponds to $p_1 \stackrel{aca}{\Longrightarrow} p_3$, and $p_3 R q_3$.

■ END EXAMPLE

A preorder is a set, thus we can relate different preorders with each other by set inclusion.

2.2.7 Definition (Finer and coarser) Let $\leq, \leq'$ be preorders and $\leq \subseteq \leq'$. We say that $\leq$ is *finer* than $\leq'$, and that $\leq'$ is *coarser* than $\leq$. Analogously for equivalences.

The definition of *finer* and coarser reflects what was already mentioned in the introduction of this section. A finer implementation relation is more "picky" in comparing the behaviour of processes: fewer processes are accepted as implementations. A coarser implementation relation, on the other hand, is more "generous": more processes are accepted as implementations. If then $\leq$ finer than $\leq'$, then this implies that $p \leq q \implies p \leq' q$. Since $\subseteq$ is a partial order, there are incomparable preorders.
The next proposition compares bisimilarity and trace equivalence.

2.2.8 Proposition (Bisimulation and trace equivalence) $\sim_{tr}$ is coarser than $\sim_B$, but not finer.

Proof:

Coarser means that we want to show that $\sim_B \subseteq \sim_{tr}$, and therefore, if for $p, q \in \mathbb{P}$, if $p \sim_B q$, then $p \sim_{tr} q$.

Proof overview:

1. We assume $p \sim_B q$
2. We show by induction over the length of σ that, whenever $\sigma \in traces(p)$, then $\sigma \in traces(q)$, *i.e.,* $traces(p) \subseteq traces(q)$;
3. It is the easy to see that the proof can be reused to show that whenever $\sigma \in traces(q)$, then $\sigma \in traces(p)$, *i.e.,* $traces(q) \subseteq traces(p)$;
4. The conclusion is then of course that $traces(q) = traces(p)$, *i.e.,* $p \sim_{tr} q$.

ad 1.) So, given $p \sim_B q$, we know there is a bisimulation relation R such that pRq.

ad 2.) Let $\sigma \in traces(p)$. Induction start: $|\sigma| = 0$, *i.e.,* $\sigma = \varepsilon$. $\varepsilon \in traces(q)$ is trivial.

The induction hypothesis is now that we have shown for all $\sigma \in traces(p)$ with $|\sigma| = n$ that $\sigma \in traces(q)$. Let us now assume that $\sigma \cdot a \in traces(p)$ for $a \in Act$,

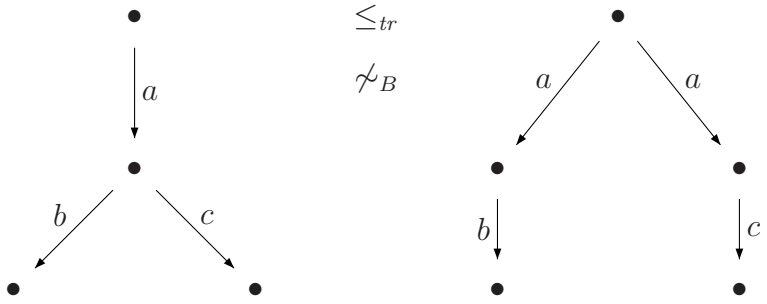
Note: All material on the slides used in the lecture is in the script.

i.e., $|\sigma \cdot a| = n{+}1$. So there is a $p' \in \mathbb{P}$ with $p \xrightarrow{\sigma} p'$ and $p' \xrightarrow{a}$. Since $p \sim_B q$ and $p \xrightarrow{\sigma} p'$, there is a $q' \in \mathbb{P}$ such that $q \xrightarrow{\sigma} q'$ and $p' R q'$. Since R is a bisimulation, $p' R q'$ implies that if $p' \xrightarrow{a}$, then $q' \xrightarrow{a}$ as well. Therefore, $\sigma \cdot a \in \mathit{traces}(q)$.

ad 3.) Since $\sim_B$ is an equivalence relation, also $q \sim_B p$ holds, and we can repeat the proof in steps 1.)/2.) for the other direction.

ad 4.) as said above: $\mathit{traces}(q) = \mathit{traces}(p)$, *i.e.*, $p \sim_{tr} q$.

Not finer can be shown with the following counterexample: it proves that $p \sim_{tr} q$ does not imply $p \sim_B q$.

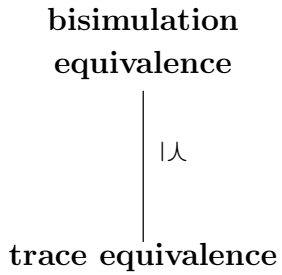


■ END PROOF

Note that in step 2.) in the previous proof, if there is no $\sigma \cdot a \in \mathit{traces}(p)$ with $|\sigma \cdot a| = n{+}1$, then the statement is vacuously true.

2.2.9 Definition For two preorders (or equivalences) pre_1, pre_2 with $pre_1 \subseteq pre_2$, we write $pre_1 \preceq pre_2$

Since $\preceq$ is a directly related to set inclusion, $\preceq$ is a partial order.
Up til now:



Note: All material on the slides used in the lecture is in the script.

The whole picture: The linear time – branching time spectrum
Van Glabbeek [vG01] has found a more complete picture. See Figure 2.3. Read "semantics" as "equivalence" and arrow $\longrightarrow\!\!\blacktriangleright$ as $\preceq$.
This means that in-between trace- and bisimulation equivalence there are 10 other meaningful equivalences with different degrees of coarseness.

2.3 A little surprise

Let $\mathbb{P}_{\text{det}}$ be the set of all deterministic processes.

2.3.1 Proposition $\sim_B$ and $\sim_{tr}$, both restricted on $\mathbb{P}_{\text{det}}$, coincide.

Restricted means: $\sim_B |_{\mathbb{P}_{\text{det}}} = \{(p,q) \mid p \sim_B q \text{ and } p,q \in \mathbb{P}_{\text{det}}\}$, and thus, $\sim_B |_{\mathbb{P}_{\text{det}}} = \leq_{tr} |_{\mathbb{P}_{\text{det}}}$.
This proposition has the following implications:

1. The linear time – branching time spectrum for deterministic processes has therefore only one element.
2. It also means that the different equivalences differ in essence in the way how they treat nondeterminism in processes, and how much of it.
3. The position in the diagram gives a hint on how much nondeterminism is taken into account.
4. trace equivalence apparently ignores non-determinism altogether (because nondeterministic and deterministic processes with the same set of traces are identified)
5. bisimulation on the other hand takes much of the information on nondeterminism into account to differentiate between processes.
6. trace equivalence works on traces, linear sequences of actions $\Longrightarrow$ "linear time"
7. bisimulation takes branching, esp. nondeterministic branching into account $\Longrightarrow$ "branching time"
8. Studying preorders and implementation relations on processes are thus studies in nondeterminism
9. Some of the preorders in detail and the connection to testing later

Proof of Proposition 2.3.1: We assume that processes $p, q \in \mathbb{P}_{\text{det}}$. We must show that, if $p \sim_{tr} q$, then $p \sim_B q$ (the other direction we know already to be true). The technique to prove this is very classical: we construct a relation R on the processes such that pRq, and show that R is a bisimulation. From the definition of bisimulation equivalence it follows then that $p \sim_B q$. First, note that for all $\sigma \in \mathit{traces}(p)$, p **after** σ is a singleton (due to the determinism of p), *i.e.,* a set $\{p'\}$ with $p' \in \mathbb{P}$. In the following we identify p' and $\{p'\}$.

Note: All material on the slides used in the lecture is in the script.

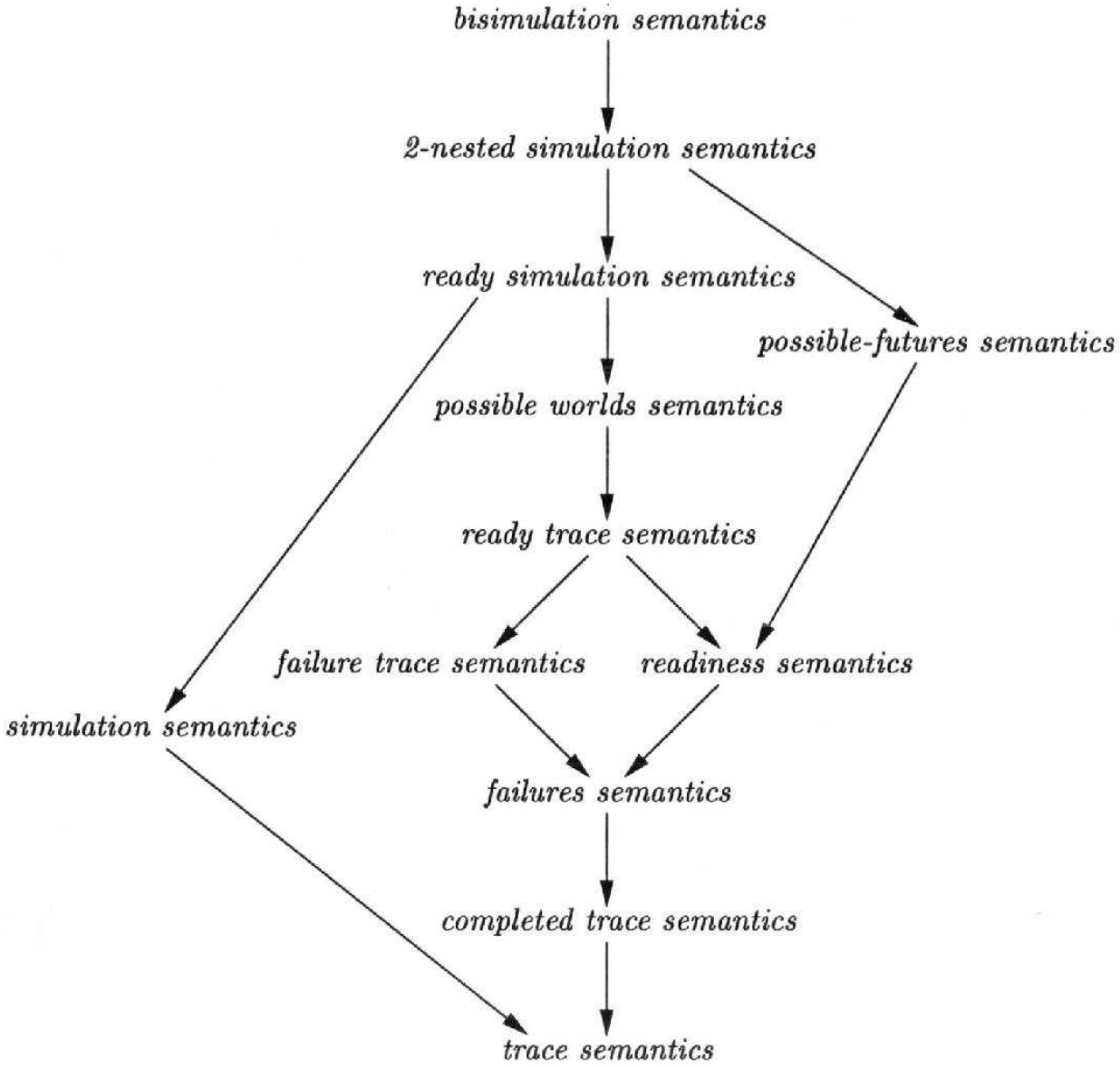


Figure 2.3: The linear time – branching time spectrum (slide #10) (copied from [vG01])

.

Note: All material on the slides used in the lecture is in the script.

Definition of R: Let $R \subseteq \mathbb{P} \times \mathbb{P}$ be the relation

$$R = \bigcup_{\sigma \in traces(p)} \left\{ (p', q') \mid p' = \underline{p \textbf{ after } \sigma}, q' = \underline{q \textbf{ after } \sigma} \right\}$$

R is a bisimulation: Let $(p', q') \in R$. There is thus a trace $\sigma \in traces(p) = traces(q)$ with $p' = \underline{p \textbf{ after } \sigma}$ and $q' = \underline{q \textbf{ after } \sigma}$. If now $p' \xrightarrow{a} p''$ for some $a \in Act$ and $p'' \in \mathbb{P}$, then $\sigma \cdot a \in traces(p)$, thus also $\sigma \cdot a \in traces(q)$. Therefore, also $q' \xrightarrow{a} q''$ for some $q'' \in \mathbb{P}$. Since p and q are deterministic, $p'' = \underline{p \textbf{ after } \sigma \cdot a}$ and $q'' = \underline{q \textbf{ after } \sigma \cdot a}$, and thus $(p'', q'') \in R$, due to the definition of R.

To show the "bi" in bisimulation, we must repeat the previous argument, with q, q', q'' changing the places with p, p', p'' in the appropriate places, respectively.

Thus, R is a bisimulation, and since $(p, q) \in R$, $p \sim_B q$ holds.

■ END PROOF

Note that Figure 2.3 depicts the linear time – branching time spectrum **I**, which only holds for the set of processes with no internal steps. Linear time – branching time spectrum II is more complicated. See Figure 2.4

Note: All material on the slides used in the lecture is in the script.

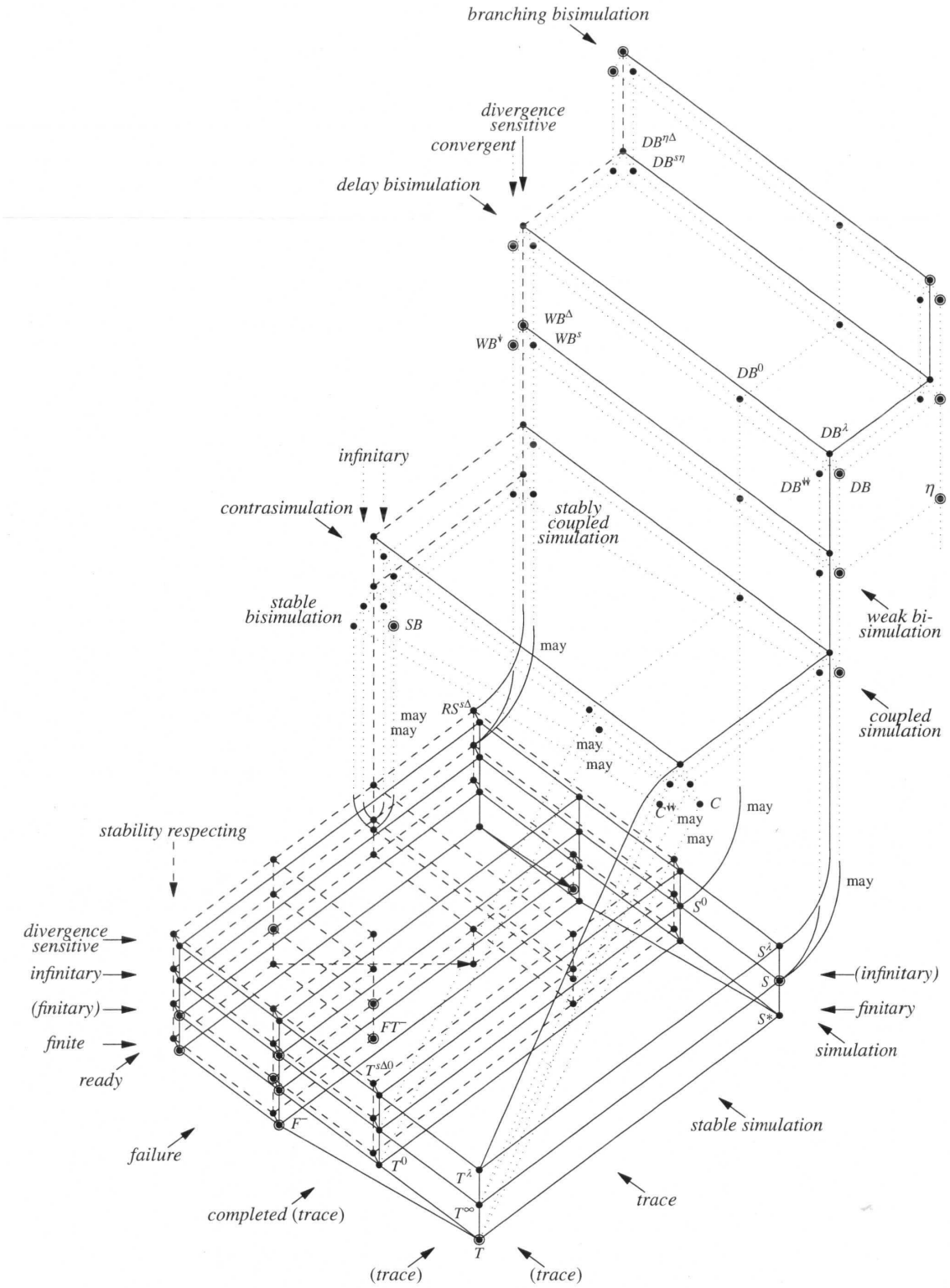


Figure 2.4: The linear time – branching time spectrum II (slide #11)

Note: All material on the slides used in the lecture is in the script.

Chapter 3

Distinguishing Processes by Observation

In the previous sections we have learned how to describe behaviour in terms of processes and labelled transition systems, and the elementary concept on how to distinguish behaviour of processes: implementation relations.
The next step is now to combine these concepts with *testing*.
As has been mentioned before, testing has much of an *experiment*: the test-object is manipulated in some way, and the reactions are observed. We will now adopt this view and define how we can compare the behaviour of *processes* by *manipulation* and *observation*. In order to do this we will define precisely the different ways that allow us to manipulate a process and the different types of observations that we can make. We will in general assume that we have no knowledge of the inner structure of the process that we observe, but can only see its different kind of interactions with the environment. The question that we want to answer is:

- how can we, by means of manipulating and observing two processes $p, q \in \mathbb{P}$, find out whether they behave the same or not?
- Can we, and, if yes, how can we characterise implementation relations in terms of the experiments, manipulations and observations?

In [vG01], v. Glabbeek describes the *generative machine*, an abstract, but intuitive framework to describe manipulations, observations of, and experiments on processes. His idea is the following: the process is put in a black box, and can only be accessed by a panel (see Figure 3.1) with different lights, displays, switches and buttons.
How can we characterise implementation relations in terms of *testing*?

- We consider a set of observers, $\mathcal{O}$.
- Observers can be processes, logical formulae, ...
- observers make *primitive observations*: *e.g., the system can execute an action a right now*, *can not execute an action a*, cannot make any action at all, etc.

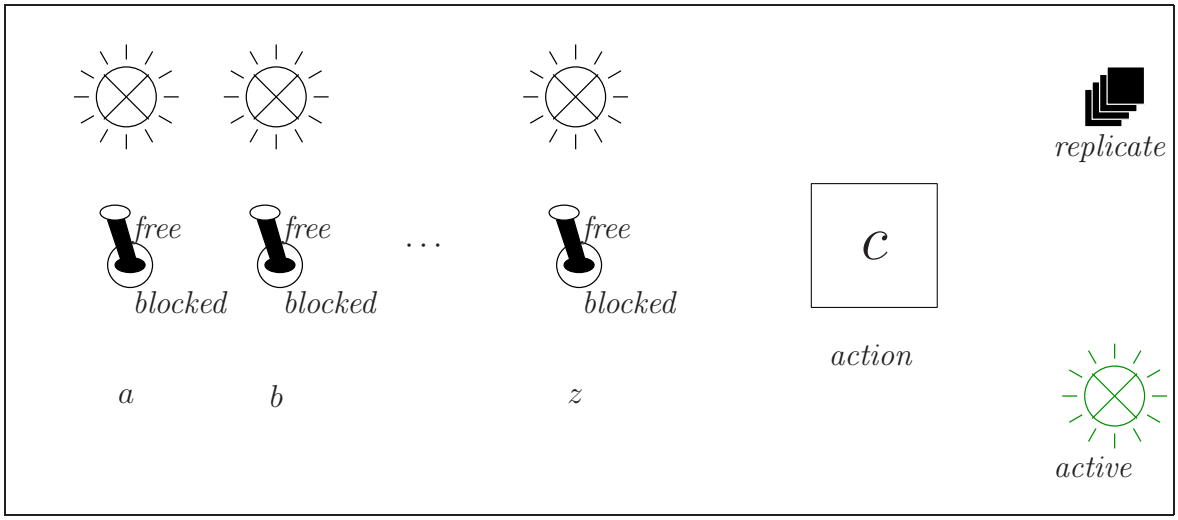


Figure 3.1: User panel of the generative machine

- Observers $o \in \mathcal{O}$ generate *observations* $\mathsf{obs}(o, p)$ from the primitive observations of the observed system p: *traces, verdicts, ...*
- In general: implementation relations are defined by relating observations to each other:

$$p \textbf{ imp } q :\Longleftrightarrow \forall o \in \mathcal{O} : \mathsf{obs}(o, p) * \mathsf{obs}(o, q)$$

where $*$ is some relation like *e.g.,* $\subseteq$.

In this chapter:

- Observers will be special processes
- observations syntactic ingredients of observers
- outcomes will be derived with a special parallel operator.

Again: no τ-steps!

End of Lecture #4

3.1 Trace inclusion

Our observers will be a special kind of processes, also called *test expressions*

Note: All material on the slides used in the lecture is in the script.