An Invitation to Friendly Testing*

David de Frutos-Escrig, Luis Llana-Díaz and Manuel Núñez

Dept. de Sistemas Informáticos y Programación
Universidad Complutense de Madrid, E-28040 Madrid, Spain.
E-mail: {defrutos,llana,manuelnu}@eucmax.sim.ucm.es

Received June 2, 1998.

Abstract

We present a new testing semantics, called *friendly testing*, whose main property is that the induced preorder between processes $\sqsubseteq_{\text{fr}}$ is consistent with the conformance relation, and so we have, for instance, $a \oplus b \sqsubseteq_{\text{fr}} a \sqsubseteq_{\text{fr}} a + b$. The new theory is strongly based on De Nicola & Hennessy's work on testing. Friendly tests are defined exactly as in their work, except that internal actions are not allowed. However, in order to obtain the desired notion of friendly testing this restriction is not enough and we also have to relax the conditions to pass a test. Thus we obtain a new testing semantics and a new preorder between processes which is strictly weaker than the relation $\sqsubseteq_{\text{must}}$. Moreover, we present an alternative characterization of our new testing semantics by defining a modification of acceptance sets.

Keywords: Process algebra, semantics of concurrency, testing, conformance.

1 Introduction

Since its presentation in [1,2] *Testing Semantics* has been widely studied and used as a natural way to define an observational behavior with a reasonable power to distinguish semantically different processes. Testing Semantics is defined by observing the operational semantics of processes by means of *tests*. Tests are just processes which may execute a new action ω reporting *success* of the test application. To define the application of a test to a process, we consider the different computations of the experimental system which is obtained by composing in parallel the test and the tested process. We say that a computation is *successful* if there exists a step in the computation such that the associated test can execute the action ω. Since it is possible that some, but not all, of the computations may succeed, we can distinguish two different semantics which are called *may* and *must* semantics. A process P *may* pass a test T (in short P *may* T) if the composition of P and T has at least one successful computation, while P *must* pass T (in short P *must* T) if every computation of the composition of P and T is successful. By combining these two semantics we can obtain a third one: the *may-must* semantics.

We say that two processes are (*may*, *must*) equivalent iff they pass (in the corresponding sense) the same tests. In addition to these equivalences, denoted by $\approx_{\text{may}}$ and $\approx_{\text{must}}$ respectively, we obtain the corresponding preorders between processes: Q is *better* than P if any test passed by P is also passed by Q. These preorders are used to define the notion of approximation needed to build the semantics of recursive processes by applying the technique of minimal fixpoints. Another application corresponds to a formalization of the notions of refinement and conformance (see e.g. [3]). It is in this framework where the term *better* finds its justification, because $P \sqsubseteq Q$ holds when Q can always be (correctly) used instead of P.

It is well-known that $P \sqsubseteq_{\text{must}} P + Q$ is not fulfilled at all, and so the testing preorder does not capture the notion of conformance. For instance, we have that $a \not\sqsubseteq_{\text{must}} a + b$ since $a \not\sqsubseteq_{\text{may}} a + b$, and more in general we cannot improve a process by extending its behavior by means of new capabilities, as it is sometimes done in the implementation task.

Research supported in part by CICYT project TIC 97-0669-C03-01.

* Or how to test processes without punishing them when they are able to execute actions.

Another proposal for formalizing the implementation notion is the so-called *conformance relation*, defined in [3] by:

$$P_1 \text{ conf } P_2 \iff \text{Failures}(P_1) \cap \Big(\text{Traces}(P_2) \times \mathcal{P}(\mathbf{Act})\Big) \subseteq \text{Failures}(P_2)$$

Using the usual (in the testing community) ordering notation we will write, reversing the arguments, $P_2 \sqsubseteq_{\text{conf}} P_1$. From this definition and the characterization by refusals of the testing preorder it is immediate that the conformance relation is a weakening of the testing preorder. Moreover we have $a \sqsubseteq_{\text{conf}} a+b$, and more in general $P \sqsubseteq_{\text{conf}} P+Q$ whenever P is extended by adding new capabilities offered by Q. Unfortunately, the conformance relation is not a preorder, since

$$a \oplus (b\,;c) \sqsubseteq_{\text{conf}} a \sqsubseteq_{\text{conf}} a+b, \text{but } a \oplus (b\,;c) \not\sqsubseteq_{\text{conf}} a+b$$

This is because under the definition of the conformance relation all the traces of P_2 have to be independently explored.

Based on these somehow negative facts about the two most extended conformance relations, that is $\sqsubseteq_{\text{must}}$ and $\sqsubseteq_{\text{conf}}$, we have looked for a compromise between them which could inherit the good properties of both, while avoiding their problems. We think that it is desirable to maintain a testing interpretation for a relation that will be the basis for the testing framework. So, we have tried to find a new notion of test, and of the passing of tests mechanism, such that the desired preorder could be defined in the usual testing way: an implementation is better than (or adequate with respect to) a specification, relatively to our desired conformance relation, if it passes more tests than this last one. It is clear that any relation defined in this way is a preorder. As an immediate consequence we have that the conformance relation cannot be characterized in this way.

We have investigated reasons why *must* testing is too powerful and ways to define a new notion of testing equivalence correcting this situation. We have found that De Nicola & Hennessy's tests [1] have the power to *punish* processes being able to execute actions. So, we have $a \not\sqsubseteq_{\text{must}} a+b$, since the former process passes the test $(a\,;\omega)+(b\,;\texttt{STOP})$, while the latter does not. We are interested in a testing framework where tests cannot punish processes when they are able to execute some action. This is why we call *friendly* testing to our new testing scenario, and we denote by $\sqsubseteq_{\text{fr}}$ the corresponding preorder. Intuitively, friendly tests can *reward* with success when the desired traces are executed, but cannot *punish* with failure when some other traces are executable by the tested process. A possible interpretation of this fact leads to the conclusion that the problem comes because we allow both successful (ω) and unsuccessful (STOP) terminations in tests, but this is not the case. In fact, if we restrict the set of tests to *always successful* tests, i.e. tests whose *leaves* are always labeled by ω, nothing is gained, since we could always assume the existence of a new action *reject*, to be read as failure, such that any STOP (failure) termination in the original tests could be simulated by a *reject* $;\omega$ termination.

This means that we have to look more thoroughly for the real cause of the power of tests to punish processes. In order to avoid that tests could be controlled by tested processes in the same way that they control the execution of these, internal actions that can be autonomously executed by the test are added to its alphabet, thus introducing an asymmetry in the definition of the experimental systems $(P \parallel T)$. We are not interested at all in this increasing of the power of the testing mechanism, and thus when we define our friendly tests we will not allow internal actions in tests.

But only by reducing the family of tests we cannot get the desired notion of friendly passing of tests. We cannot remove all tests containing the external choice operator $+$, because it is very easy to prove that this would lead us to a too restrictive semantics of "secure" traces (traces that *must* be executed). Instead, we change the definition of how tests are passed in such a way that external choices in tests will not behave as internal ones. For instance, we have that the test $(a;\omega)+(b;f;\omega)$, is not *must* passed by the process $a+b$ because of the computation executing b, after which we cannot get the passing mark ω. We want this test to be friendly passed by the considered process, since we interpret that if we have an external choice both in the test and in the tested process, it is enough that the computations executing some of the common offered actions will succeed.

In this paper we present the definition of friendly testing and compare its properties with those of classical testing. Since friendly tests are "weaker" than classical tests, the induced preorder is also weaker than must testing preorder and fulfills those properties that we expect from a conformance relation. As we have already commented, friendly tests by themselves are more natural, since we do not need the inclusion of an internal action in the alphabet of tests in order to get the desired strength. Unfortunately we need a more elaborated definition of test passing. We will see that even if we need such a more involved machinery to give the adequate definition of friendly testing, the formal properties of the induced friendly preorder are very similar to those of must preorder. Thus, our main result is a characterization of the friendly preorder, obtained by a slight change in the definition of acceptance sets, characterizing the must preorder.

Currently we are working on a smooth extension of our theory to cope with infinite processes. Finally we will mention [4], where we relate our friendly preorder with the conformance relation, proving that it is exactly the transitive closure of the conformance relation, and therefore a nice alternative to formalize conformance. Thus we conclude that, as we will try to justify in the following, friendly testing is indeed a nice compromise between must testing and the conformance relation.

The rest of the paper is structured as follows. In Section 2 we recall the definitions of classical testing and some of its properties. In Section 3 we present the definition of our new testing semantics. In order to focus on the main ideas, first we just consider normal form processes. Later, we extend the definition to cover arbitrary (finite) processes. In Section 4 we present an alternative characterization of friendly testing semantics by means of a modification of the notion of acceptance sets, called *friendly acceptance sets*. Finally, in Section 5 we sketch some lines for further work.

2 Classical Testing: Definitions and Properties

Testing semantics was introduced in [1,2]. In this section we will present the main definitions borrowed, with some minor changes, from [2], which is the main reference for the formal study of the testing methodology.

Definition 2.1. *The set of finite processes, denoted by Proc, is defined as the set of expressions given by the following BNF-grammar:*

$$P ::= \mathtt{STOP} \mid a\,;P \mid P + P \mid P \oplus P$$

where $a \in \mathbf{Act}$ (the set of actions). For the sake of clarity we will omit trailing occurrences of STOP.

The operational semantics of the language is defined by:

$$\frac{}{a;P \xrightarrow{a} P} \qquad \frac{}{P \oplus Q \succ\!\!\rightarrow P} \qquad \frac{}{P \oplus Q \succ\!\!\rightarrow Q}$$

$$\frac{P \xrightarrow{a} P'}{P+Q \xrightarrow{a} P'} \qquad \frac{Q \xrightarrow{a} Q'}{P+Q \xrightarrow{a} Q'} \qquad \frac{P \succ\!\!\rightarrow P'}{P+Q \succ\!\!\rightarrow P'+Q} \qquad \frac{Q \succ\!\!\rightarrow Q'}{P+Q \succ\!\!\rightarrow P+Q'}$$

The following conventions will be used:

$P \xrightarrow{a}$ stands for $\exists P' : P \xrightarrow{a} P'$, $P \not\xrightarrow{a}$ stands for $\nexists P' : P \xrightarrow{a} P'$,
$P \not\rightarrow$ stands for $\nexists P', a : P \xrightarrow{a} P'$,
$P \succ\!\!\rightarrow$ stands for $\exists P' : P \succ\!\!\rightarrow P'$, $P \succ\!\!\not\rightarrow$ stands for $\nexists P' : P \succ\!\!\rightarrow P'$, and
$\succ\!\!\rightarrow^*$ stands for the transitive and reflexive closure of $\succ\!\!\rightarrow$.

Moreover, for $s = a_1, \ldots, a_n$ we write $P \stackrel{s}{\Longrightarrow} P'$ if there exist $P_1, \ldots, P_n, P'_1, \ldots, P'_n$ such that $P \succ\!\!\rightarrow^* P_1 \xrightarrow{a_1} P'_1 \succ\!\!\rightarrow^* P_2 \cdots P_n \xrightarrow{a_n} P'_n \succ\!\!\rightarrow^* P'$, and we call *computations* of P to these sequences. We say that a computation $P \stackrel{s}{\Longrightarrow} P'$ is *complete* if we have both $P' \not\rightarrow$ and $P' \succ\!\!\not\rightarrow$.

Tests are just finite processes over the alphabet $\mathbf{Act} \cup \{1, \omega\}$, and the previous operational semantics is also valid for tests. We define the operational semantics of *experimental systems*, $P \,\|\, T$,

as follows:

$$\frac{P \xrightarrow{a} P' \ \wedge \ T \xrightarrow{a} T'}{P \parallel T \xrightarrow{a} P' \parallel T'} \qquad \frac{P \rightarrowtail P'}{P \parallel T \rightarrowtail P' \parallel T} \qquad \frac{T \rightarrowtail T'}{P \parallel T \rightarrowtail P \parallel T'} \qquad \frac{T \xrightarrow{1} T'}{P \parallel T \xrightarrow{1} P \parallel T'}$$

Let us remark that, in contrast with the classical definition, we do not hide the actions that experimental systems execute.

Definition 2.2. *Let P be a process and T be a test.*

1. *We say P may T if there exists a computation $P \parallel T \stackrel{s}{\Longrightarrow} P' \parallel T'$ such that $T' \xrightarrow{\omega}$.*
2. *We say P must T if for any complete computation $P \parallel T \stackrel{s}{\Longrightarrow} P' \parallel T'$ there exist $s', s'' \in (\mathbf{Act} \cup \{1, \omega\})^*$ such that $s = s's''$ with $P \parallel T \stackrel{s'}{\Longrightarrow} P'' \parallel T'' \stackrel{s''}{\Longrightarrow} P' \parallel T'$ and $T'' \xrightarrow{\omega}$.*

Definition 2.3. *Let P, P' be processes.*

1. *We write $P \sqsubseteq_{\text{may}} P'$ if for any test T we have P may T implies P' may T, and $P \approx_{\text{may}} P'$ if $P \sqsubseteq_{\text{may}} P'$ and $P' \sqsubseteq_{\text{may}} P$.*
2. *We write $P \sqsubseteq_{\text{must}} P'$ if for any test T we have P must T implies P' must T, and $P \approx_{\text{must}} P'$ if $P \sqsubseteq_{\text{must}} P'$ and $P' \sqsubseteq_{\text{must}} P$.*
3. *We write $P \sqsubseteq_{\text{may-must}} P'$ if $P \sqsubseteq_{\text{may}} P'$ and $P \sqsubseteq_{\text{must}} P'$, and $P \approx_{\text{may-must}} P'$ if $P \approx_{\text{may}} P'$ and $P \approx_{\text{must}} P'$.*

Proposition 2.4. *Let P, P' be processes. The following properties hold:*

- $P \sqsubseteq_{\text{must}} P' \implies P' \sqsubseteq_{\text{may}} P$.
- $P \approx_{\text{must}} P' \implies P \approx_{\text{may}} P'$.
- $P \approx_{\text{may-must}} P' \iff P \approx_{\text{must}} P'$.

In the following we will concentrate on the *must* preorder $\sqsubseteq_{\text{must}}$, and we will usually write $\sqsubseteq$ instead of $\sqsubseteq_{\text{must}}$. It is immediate to see that the family of tests can be restricted to those that always execute ω as the last action of all their computations, since this would not change the distinguishing power of the full family of tests. Besides we could also assume that tests do not include the internal choice operator since we have

$$P \ must \ (T \oplus T') \iff P \ must \ T \ \wedge \ P \ must \ T'$$

Of course, the same is not true for the external choice operator, since, for instance, we have $a \oplus b$ *must* $(a\,;\omega + b\,;\omega)$ but neither $a \oplus b$ *must* $a\,;\omega$ nor $a \oplus b$ *must* $b\,;\omega$. In fact, if we remove both choice operators from the signature of tests then only *trace tests* $t;\omega$ remain. By means of them we can check if the execution of a trace could be rejected by the tested process. Therefore these tests characterize the *secure traces* semantics that associate to each process the set of traces that it cannot refuse. Thus, under this semantics $a \oplus b$ is equivalent to STOP. We need indeed a test like $a;\omega + b;\omega$ to distinguish these processes.

The testing preorder between processes can be characterized in an explicit way by means of the so-called *acceptance sets*.

Definition 2.5. *Let P, P' be processes.*

- *We define the set of initial actions of P, denoted by $S(P)$, as $S(P) = \{a \mid P \stackrel{a}{\Longrightarrow}\}$.*
- *Let s be a sequence of actions. We define the acceptance sets of P after s as $\mathcal{A}(P, s) = \{S(P') \mid P \stackrel{s}{\Longrightarrow} P'\}$.*
- *We write $P \ll_{\text{must}} P'$ iff for any $s \in \mathbf{Act}^*$ we have that for any $A' \in \mathcal{A}(P', s)$ there exists $A \in \mathcal{A}(P, s)$ such that $A \subseteq A'$.*

Theorem 2.6. *Let P, P' be processes. Then we have $P \sqsubseteq_{\text{must}} P'$ iff $P \ll_{\text{must}} P'$.*

It is immediate to check that acceptance sets give us exactly the same information as refusals[5] and thus the following equivalent characterization of the testing preorder is obtained as a corollary.

Corollary 2.7. *Let P, P' be processes. Then, $P \sqsubseteq_{\text{must}} P'$ iff* Failures$(P') \subseteq$ Failures(P).

Finally, we introduce Hennessy normal form processes, which can be used to fully characterize the must testing semantics. First we define *normal forms*, which give us the notation to define Hennessy normal forms, and also will be used in the following section.

Definition 2.8. *Normal form processes are those defined by the foolowing grammar:*

$$P ::= \bigoplus_{A \in \mathcal{A}} \sum_{a \in A} (a; P_a)$$

where $\mathcal{A} \subseteq \mathcal{P}_f(\mathbf{Act})$, $\mathcal{A}$ is non-empty, and $\bigoplus$ and $\sum$ are th obvious generalizations of $\oplus$ and $+$ to an arbitrary (but finite) number of arguments.

By convention $\bigoplus_{A\in\{\phi\}}$ represents the process STOP. We call *deterministic processes* those in which all the occurrences of $\bigoplus$ are trivial, which means that the indexes $\mathcal{A}$ verify $|\mathcal{A}| = 1$. In order to simplify the notation, we will usually omit trivial occurrences of the operator $\bigoplus$. Note that all the continuations after different occurrences of the same action a in several states $\mathcal{A} \in \mathcal{A}$ must be the same.

Definition 2.9. *Let $\mathcal{A} \subseteq \mathcal{P}(\mathbf{Act})$ be a set of sets of actions. We say that $\mathcal{A}$ is* saturated *whenever it is closed with respect to union and convex closure, that is:*

- *$A_1, A_2 \in \mathcal{A}$ implies $A_1 \cup A_2 \in \mathcal{A}$.*
- *$A_1, A_2 \in \mathcal{A}$ and $A_1 \subseteq A' \subseteq A_2$ imply $A' \in \mathcal{A}$.*

Definition 2.10 (Hennessy Normal Form).

- STOP *(or equivalently $\bigoplus_{A\in\{\phi\}} \sum_{a\in A} P_a$) is a Hennessy normal form.*
- *If $\mathcal{A}$ is a saturated set and for every $a \in \bigcup_{A\in\mathcal{A}} A$ there is a Hennessy normal form P_a then $\bigoplus_{A\in\mathcal{A}} P_A$ is a Hennessy normal form, where P_A represents $\sum_{a\in A} a\,;P_a$.*

Proposition 2.11. *Let P be a process:*

- *There exists a unique normal form process $nf(P)$ such that $P \approx_{\text{must}} nf(P)$.*
- *If $nf(P) = \bigoplus_{A\in\mathcal{A}} \sum_{a\in A} a\,;n(a)$ and $A \in \mathcal{A}$, then there exists a process P' such that $P \rightarrow\!\!\!\!\!\rightarrow^* P' \not\rightarrow$ and $\{a \mid P' \xrightarrow{a}\} \subseteq A$.*

3 Friendly Testing: Definitions and Justification

Before introducing the notion of friendly testing, we discuss the reasons why the must testing preorder is too strong, as already commented in our introduction. First we study the role of internal actions in tests. When defining the classic notion of test, and the interaction between processes and tests, Hennessy [2,p.67] writes:

> *... An obvious possibility is to let $\rightarrow$ be the least relation which, for an arbitrary $a \in$ **Act**, satisfies*
>
> $$t \xrightarrow{a} t',\ p \xrightarrow{a} p' \ \text{ implies } \ t \parallel p \rightarrow t' \parallel p'$$
>
> *so that they interact by performing the same action. This gives the experimenter complete control over the process ... Unfortunately this control works both ways since the process also has the same control over the experimenter. To break this symmetry, and allow the experimenter a natural independence from the process it is investigating we allow it to use a special action 1, such that we have*
>
> $$t \xrightarrow{1} t' \ \text{ implies } \ t \parallel p \rightarrow t' \parallel p$$

We could rephrase this sentence by saying that tests may be nondeterministic in order to avoid to be always controlled by the tested process. Internal actions, labeled by 1, are a way to introduce such a nondeterministic behavior, and at the moment they appear in [2] they are fully justified, since only the external choice operator had been considered. However, the internal choice operator is later introduced [2,p.89], and once it appears we could think that internal actions are not needed any more in order to have nondeterministic choices in tests. But, even in the presence of internal choices, the possibility of having internal actions increases the discriminatory power of tests. For instance, we have STOP $\not\sqsubseteq a$ since for $T = 1\,;\omega + a\,;f\,;\omega$ we have STOP *must* T but not a *must* T. Nevertheless the role of the internal action in the test T can be simulated by combining the internal choice operator and the accepting action in a tricky way.[1] We can see that for the test $T' = (\omega \oplus \omega) + a;f;\omega$ we have P *must* T iff P *must* T'. Moreover, we can substitute any subtest $1;\omega$ by $\omega \oplus \omega$ to obtain an equivalent test. And since all the internal actions appearing in the set of *essential* tests, which are proved to have the same discriminatory power as the full family of tests (see the proof of Theorem

[1] We thank the anonymous referee of a previous version of this paper for this remark.

2.6 in [2]), appear in the context of a subtest 1; ω, we can substitute these essential tests by an equivalent family of tests without internal actions.

We think that this is a rather simple but very important result about the power of tests, that probably is already known for many people familiar with testing theory, but nevertheless we have not found any explicit statement of this fact in the literature on the topic.

As a conclusion we will say that the allowance of internal actions in tests, once we have an internal choice operator generating internal transitions, is not justified since they do not add any power to tests. Thus we will preclude this kind of actions when we define our notion of friendly testing. But as long as we allow the internal choice operator in tests we still have the power of internal actions to punish a process when it is able to execute some action. Therefore, we will also preclude the use of the internal choice operator when defining friendly testing. If we consider plain must testing when the class of tests is restricted to those obtained by combining the prefix and external choice operator, and we denote by $\sqsubseteq_{\mathrm{obs}}$ the induced preorder, it is clear that this relation is closer to the conformance relation than the original must preorder. For instance, for any process P we have $\mathtt{STOP} \sqsubseteq_{\mathrm{obs}} P$, which is also the case for the conformance relation. We also have $a\,;\mathtt{STOP} \sqsubseteq_{\mathrm{obs}} a\,;P$ for any P, and more in general, the relation $\sqsubseteq_{\mathrm{obs}}$ would reflect the conformance relation if no external choices appear in the compared processes. Nevertheless, if we consider the processes a and $a+b$, which are related by $\sqsubseteq_{\mathrm{conf}}$, we have $a \not\sqsubseteq_{\mathrm{obs}} a+b$ since the test $(a;\omega)+(b;f;\omega)$ is passed by the former process but not by the latter. More in general, whenever we apply a test like $T = a\,;T_a + b\,;T_b$, offering several actions that could be executed by the tested process, it does not matter whether the involved choices in this process are either internal or external. As a matter of fact, and even if that would have no effect in the definition of passing tests, in [2] the transitions of experimental systems are not labeled. Such decision is justified by the assumption of testing being the only way to observe the behavior of the tested process. But in this case we cannot conclude that the allowance of tests including external choices, or more exactly the simultaneous offering of several actions, is the reason for the too strong power of the testing mechanism, and thus that we should also forbid this kind of tests. We cannot remove those tests, since we need them to conclude, for instance, $\mathtt{STOP} \sqsubseteq_{\mathrm{obs}} a \oplus b$ but $a \oplus b \not\sqsubseteq_{\mathrm{obs}} \mathtt{STOP}$, reflecting the fact that the process $a \oplus b$ is strictly better than STOP since it is always ready to execute either a or b.

In conclusion, in order to get that a is worse than $a+b$ we need to decrease the discriminatory power of tests. This can be done by using the information about the observable actions executed by experimental systems. This is why we have not removed that information when defining the computations of experimental systems, expressing the fact that passing of tests is not the only way to observe the behavior of processes.

For instance, if we consider two processes $P_1 = a;P_a+b;P_b$ and $P_2 = a;P_a \oplus b;P_b$, and apply a test $T = a;T_a+b;T_b$, we obtain, in both cases, two families of computations: Those executing either a or b as their first observable action. But in the first case we have $P_1 \| T \xrightarrow{a} P_a \| T_a$ and $P_1 \| T \xrightarrow{b} P_b \| T_b$, while in the second the first step is internal, that is, $P_2 \parallel T \rightarrowtail (a\,;P_a) \parallel T \xrightarrow{a} P_a \parallel T_a$ and $P_2 \parallel T \rightarrowtail (b\,;P_b) \parallel T \xrightarrow{b} P_b \parallel T_b$. It is clear that, as already remarked before, under the classical notion of testing, based on the analysis of the *set* of computations of experimental systems, to keep the information about the executed observable actions has no use. This justifies that they were indeed removed in the original definition in order to simplify the definitions.

If we do not base our definition of test passing in the analysis of individual computations but in a more global analysis of the full tree of computations, we will be able to notice the difference between two processes such as P_1 and P_2, when tested with a test like T. As we will detail in the following sections, the idea will be that whenever we have an external choice between several different observable actions in an experimental system, as it is the case when testing P_1 with T, in order to pass a test will be enough that the computations corresponding with the selection of *some* of the offered actions (in our example, either a or b) succeed.

So we formalize the idea that the observer maintains the control, even after a test is applied, as long as external choices remain, as it is the case for P_1 in the example above. Then it is possible that the computations leading to a failure will be avoided when friendly testing a process, and thus a test that is not passed in the classic way could be passed in the friendly way.

Moreover, whenever we find internal choices in the experimental systems we will keep the condition that all the computations must succeed in order to pass a test. This is why in the example above we can find a test like T such that P_1 friendly passes T while P_2 does not, which is not the case under must testing[2].

Although we will give a general definition of friendly testing (in Definition 3.3), which closely follows the classic operational pattern, in order to avoid obscure technicalities we start by studying a particular case. By now, we will just consider finite processes in *normal form*. Tests will be just processes over the alphabet $\mathbf{Act} \cup \{\omega\}$. Following the ideas that were discussed above, the internal action 1 is removed from the alphabet of tests. For the same reason as for processes, by now we will consider a restricted version of tests: those finite deterministic tests with acceptance actions at the end of any trace. Later, we will show that this family of deterministic tests is a basis for the full family of tests, in the sense that whenever two processes are not friendly equivalent then there exists a deterministic test distinguishing them. Thus, *deterministic tests* are defined by the grammar:

$$DT ::= \omega \mid \sum_{a \in A \subseteq \mathbf{Act}} a\,;DT$$

In Fig.1 we give a graphical representation of normal forms and deterministic tests.

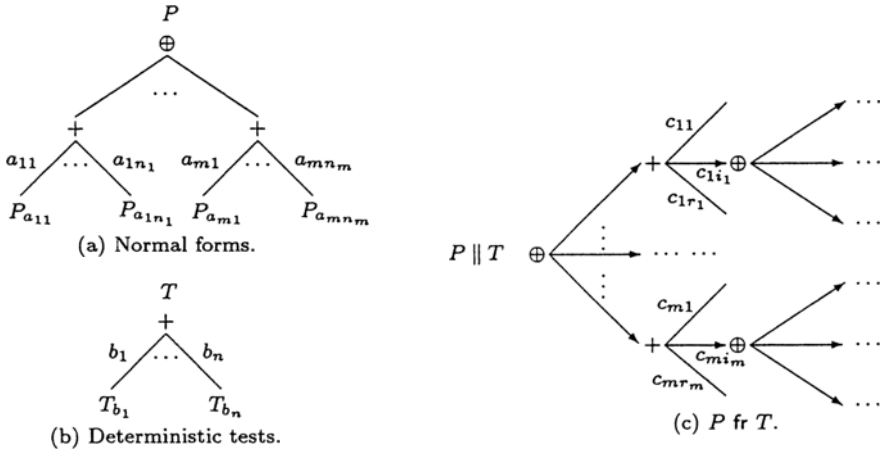

Fig.1

Definition 3.1. *Given a normal form process P and a deterministic test T, we say that P* friendly passes *T, denoted P fr T, iff one of the following conditions holds:*

1. $T = \omega$.
2. $P = \bigoplus_{A \in \mathcal{A}} P_A$, *and for each* $A \in \mathcal{A}$, P_A fr T.
3. $P = \sum_{a \in A}(a\,;P_a)$, $T = \sum_{b \in B}(b\,;T_b)$, *and there exist some* $a \in A \cap B$ *such that* P_a fr T_a.

Note that this recursive definition is well founded because we only consider finite tests.

Let us note that the first two cases of this definition are equivalent to those of classical must testing. The differences appear in the last case. If we are testing a generalized external choice, and the test offers several actions in that choice, then we do not impose that all possible computations must succeed; on the contrary, we only impose that the computations starting with one of the (common) offered actions succeed. In Fig.1(c) we illustrate this definition. In order to friendly pass the test, it is enough that all computations obtained by following the arrows succeed.

Although we are more interested in the (good) properties of the induced semantics than in an intuitive justification of our notion of testing, we will try to justify the definition by itself. The existential quantification in the third case of the definition represents the fact that the observer maintains the control to select the action that will be executed in order to pass the test, from the

[2]It is clear that P_1 and P_2 can be distinguished under plain must testing by a test like $a\,;\omega$. In fact, if this would not be the case, they could neither be distinguished under friendly testing. However, it is interesting to observe that P_1 and P_2 cannot be distinguished under must testing by a test like T that offers both a and b; on the contrary, under friendly testing we are able to distinguish P_1 and P_2 by such a test.

set of actions that are offered for both the tested process and the test. This is the same that is done in the definition of may testing. Nevertheless, under our notion of friendly testing it is not sufficient to have a single successful computation. We need a full set of successful computations corresponding to the adequate selection of an action among the common set of offered actions for both the tested process and the test, whenever we find an external choice during the application of the test. We could say that friendly testing is a compromise between *may* and *must* semantics: external choices are tested in a *may* way, while internal choices are tested in a *must* way.

The reader could think this new notion of passing tests is much more involved than the classic one, but we advocate that this is not the case, at least for normal form processes. As a matter of fact, a recursive definition of the classical notion of must test passing for normal form processes can be obtained from our definition of friendly test passing, just by changing the existential quantification in the third condition of Definition 3.1 by a universal quantification. Of course, it is true that the original definition is more natural and easily understandable, mainly because it is not a recursive definition. But, in order to check if a test is passed, we have to explore the full set of computations of the corresponding experimental system.

One could insist on the fact that to impose that all the computations have to be successful is simpler than to check our (apparently) more complicated condition, but this is not the case. In order to check any of these notions we must (in the worst case) explore the full tree of computations. More exactly, we have to evaluate a Boolean expression whose structure is analogous to that of the tested normal form process. In the must testing case we have $\wedge$ as connective in all the internal nodes of the tree expression, while in our friendly framework we have $\wedge$ for the nodes that correspond to internal choices, and $\vee$ for those corresponding to the external choice operator. So, even if we will not give any formal proof for it, we claim that the computational complexity of testing is the same in the must and friendly frameworks: sometimes to check must testing will be faster (when the test fails), and sometimes it is faster to check friendly testing (when the test is successfully passed).

3.1 Friendly Testing for Arbitrary Finite Processes and Tests

In this section we consider arbitrary finite processes and tests. First, we introduce some auxiliary concepts for the definition of friendly testing.

Definition 3.2. *Let P be a process.*

- *We say that P is stable iff $P \not\longrightarrow$. Moreover, given a test T we say that a configuration $P \,\|\, T$ is stable iff $P \,\|\, T \not\longrightarrow$.*
- *Given an action $a \in \mathbf{Act}$ such that $P \stackrel{a}{\Longrightarrow}$, we define the process P after the execution of the action a, denoted by P/a, as $P/a = \bigoplus\{P' \mid P \stackrel{a}{\Longrightarrow} P'\}$.*

Definition 3.3 (Friendly Test Passing). *Given a process P and a test T, we say that P* friendly passes *T, written P fr T, if the following conditions hold:*

- *If $P \,\|\, T$ is stable, then either $T \stackrel{\omega}{\longrightarrow}$, or there exists some $a \in \mathbf{Act}$ such that $P \,\|\, T \stackrel{a}{\longrightarrow}$ and (P/a) fr (T/a).*
- *If $P \,\|\, T$ is not stable, then for each P', T' such that $P \,\|\, T \longrightarrow^* P' \,\|\, T'$ and both P' and T' are stable we have that either $T' \stackrel{\omega}{\longrightarrow}$, or there exists some $a \in \mathbf{Act}$ such that $P' \,\|\, T' \stackrel{a}{\longrightarrow}$ and (P/a) fr (T/a).*

Let us discuss why the second condition in the previous definition is more complicated than it could be expected. Frist we were (erroneously) working with the following condition:

> *If $P \,\|\, T$ is not stable, then there exist some processes P', T', and $a \in \mathbf{Act}$ such that $P \,\|\, T \longrightarrow^* P' \,\|\, T' \stackrel{a}{\longrightarrow}$, and either $T \stackrel{\omega}{\longrightarrow}$ or P'/a fr T'/a.*

But if we have, for instance, $P_1 = (a\,;c) + b$, $P_2 = a + (b\,;c)$ and $T = (a\,;c) + (b\,;c)$, then under this *alternative* definition $P_1 \oplus P_2$ would pass the test T. But this should not be the case, since if we consider the Hennessy normal form P of $P_1 \oplus P_2$, we get $P = (a\,;(\mathtt{STOP} \oplus c)) + (b\,;(\mathtt{STOP} \oplus c))$ which verifies $P \approx P_1 \oplus P_2$, and so P should be friendly equivalent to $P_1 \oplus P_2$, since we expect our friendly testing preorder to be weaker than the existing must testing preorder. But that cannot be the case, since P does not pass T. The problem comes from the fact that the continuations after a and b in P_1 and P_2 are not the same. One can see that our present definition solves the problem,

since the possible continuations after the execution of some action are put together in the adequate way when we consider P/a (and T/a) instead of just P'/a (and T'/a).

It is easy to check that the general definition is an extension of the one for normal forms (Definition 3.1).

Proposition 3.4. *Let P be a normal form process and T a deterministic test. We have P fr T under Definition* 3.1 *iff P fr T under Definition* 3.3.

Proof. By structural induction over P:

• If we consider $P = \mathtt{STOP} = \bigoplus_{\phi\in\{\phi\}} P_\phi$, where $P_\phi = \sum_{a\in\phi} P_a$, it is obvious that under both definitions, for any deterministic test T we have P fr $T \iff T = \omega$.

• Let $P = \bigoplus_{A\in\mathcal{A}} P_A$. Under Definition 3.1 we have P fr T iff $T = \omega$ or for any $A \in \mathcal{A}$, P_A fr T; while under Definition 3.3, since P is not stable and T is, we have P fr T iff $T = \omega$ or $T = \sum_{b\in B} b;T_b$ and for each $A \in \mathcal{A}$ there exists some $a \in A\cap B$ such that P/a fr T/a. But $T/a = T_a$ and $P/a = P_a$, and since each P_A is an initial deterministic process, we have under Definition 3.1 P_A fr T iff there exists some $a \in A \cap B$ such that P_a fr T_a.

• $P = \sum_{a\in A} a\,;P_a$. Then under Definition 3.1 we have P fr T iff $T = \omega$ or $T = \sum_{b\in B} T_b$ and there exists $a \in A \cap B$ such that P_a fr T_a; while under Definition 3.3 we have P fr T iff $T = \omega$ or $T = \sum_{b\in B} b\,;T_b$ and there exists $a \in A \cap B$ such that P/a fr T/a. But obviously $P/a = P_a$ and $T/a = T_a$, and then the equivalence between both definitions follows. □

As usual, fr induces a preorder relation between processes:

Definition 3.5. *Let P, Q be processes. We write $P \sqsubseteq_{\mathrm{fr}} Q$ iff for any test T we have P fr T implies Q fr T. Moreover, we write $P \approx_{\mathrm{fr}} Q$ iff $P \sqsubseteq_{\mathrm{fr}} Q$ and $Q \sqsubseteq_{\mathrm{fr}} P$.*

Next we present some simple properties of the friendly testing preorder:

Proposition 3.6. *Let P, P_1, P_2 be processes, and T, T_1, T_2 be tests. Then, the following properties hold:*

1. *P fr ω.*
2. *If P_1, P_2 are stable, and $\{a \mid P_1 \xrightarrow{a}\} \cap \{b \mid P_2 \xrightarrow{b}\} = \phi$ then for any test T we have $P_1 + P_2$ fr T iff P_1 fr T or P_2 fr T.*

Proof.

1. Trivial.
2. Since both P_1 and P_2 are stable, for $i \in \{1,2\}$ we have

$$P_i \text{ fr } T \iff \forall T' \text{ stable } T \rightarrowtail^* T' : \begin{array}{l} T' \xrightarrow{\omega} \text{ or} \\ \exists a \in \mathbf{Act} : P_i \xrightarrow{a}, T' \xrightarrow{a} \text{ and } P_i/a \text{ fr } T/a, \end{array}$$

and the same is true for $P_1 + P_2$. But $(P_1 + P_2) \parallel T \xrightarrow{a}$ iff $P_1 \parallel T \xrightarrow{a}$ or $P_2 \parallel T \xrightarrow{a}$, and since the offerings of P_1 and P_2 are disjoint, we have that the disjunction in the characterization above is strict and if we take $i \in \{1,2\}$ such that $P_i \xrightarrow{a}$ we have $(P_1 + P_2)/a = P_i/a$. And from all these facts we conclude the desired equivalence. □

Proposition 3.7. *Let P and Q be processes. We have:*

1. *$P \oplus Q \sqsubseteq_{\mathrm{fr}} P$.*
2. *If $S(P) \cap S(Q) = \phi$ then $P \sqsubseteq_{\mathrm{fr}} P + Q$.*
3. *If $S(P) \cap S(Q) = \phi$ then $P \oplus (P + Q) \approx_{\mathrm{fr}} P$.*

Proof.

1. Let us consider a test T such that $P \oplus Q$ fr T. We will prove by induction on the depth[3] of T that $P \oplus Q$ fr T implies P fr T. If the depth of T is 1 the result is trivial since then $P \oplus Q$ fr T implies that $\forall T' : T \rightarrowtail^* T'$ we get $T' \xrightarrow{\omega}$. So let us suppose that its depth is greater than 1. Let us consider T' and P' stable such that $P \parallel T \rightarrowtail^* P' \parallel T'$, we must show that either $T' \xrightarrow{\omega}$ or there exists some action a such that $P' \parallel T' \xrightarrow{a}$ and P/a fr T/a. As $P \oplus Q \rightarrowtail^* P'$ and $P \oplus Q$ fr T, we have either $T' \xrightarrow{\omega}$ or there exists some action a such that $P' \parallel T' \xrightarrow{a}$ and $(P \oplus Q)/a$ fr T/a.

[3]The depth of a process P (or a test T) is the length of its longest trace.

If $a \notin S(Q)$ then $(P \oplus Q)/a = P/a$ and the result is immediate; otherwise $(P \oplus Q)/a = P/a \oplus Q/a$, and we get the result by applying the induction hypothesis because $P/a \oplus Q/a$ fr T/a.

2. Let us take a test T such that P fr T, and prove $P + Q$ fr T. Let T', P' and Q' stable such that $(P + Q) \parallel T \rightarrowtail^* (P' + Q') \parallel T'$. As $P \rightarrowtail^* P'$ and P fr T we have either $T' \xrightarrow{\omega}$ or there exists some action a such that $P' \parallel T' \xrightarrow{a}$ and P/a fr T/a. And applying $S(P) \cap S(Q) = \phi$ we obtain $(P + Q)/a = P/a$, and then the result is immediate.

3. From item 1 it is enough to prove $P \sqsubseteq_{\text{fr}} P \oplus (P + Q)$. Let T be a test such that P fr T, we must show that $P \oplus (P + Q)$ fr T. Let R and T' stable such that $P \oplus (P + Q) \parallel T \rightarrowtail^* R \parallel T'$. As either $P \oplus (P + Q) \rightarrowtail P$ or $P \oplus (P + Q) \rightarrowtail P + Q$, we have that either there exists P' stable such that $P \rightarrowtail^* P'$ and $R = P'$, or there exist P'' and Q' stable such that $P \rightarrowtail^* P''$, $Q \rightarrowtail^* Q'$ and $R = P'' + Q'$. Now we have P fr T, $P \parallel T \rightarrowtail^* P'' \parallel T'$, and $P \parallel T \rightarrowtail^* P' \parallel T'$, from which we conclude that we have either $T' \xrightarrow{\omega}$ or there exist some actions $b, b' \in \mathbf{Act}$ such that $P' \parallel T' \xrightarrow{b}$ with P/b fr T/b and $P'' \parallel T' \xrightarrow{b'}$ with P/b' fr T/b'. Then taking $a = b$ whenever $R = P'$ and $a = b'$ if $R = P'' + Q'$, we obtain $R \xrightarrow{a}$. Now we can conclude, since $a \notin S(Q)$ and therefore $(P \oplus (P + Q))/a = P/a$. □

Corollary 3.8. *Let $a, b \in \mathbf{Act}$, we have:*

1. $a \sqsubseteq_{\text{fr}} a + b$.
2. $a \oplus (a + b) \sqsubseteq_{\text{fr}} a + b$.
3. $a \oplus (b\,;c) \sqsubseteq_{\text{fr}} (a + b)$. *Nevertheless* $(a\,;c) \oplus (b\,;c) \not\sqsubseteq_{\text{fr}} (a + b)$.
4. $a \oplus (a + b) \approx_{\text{fr}} a$.

Proof. Only the second part of item 3 is not immediate consequence of Proposition 3.7. To prove it we just apply the test $T = (a\,;c\,;\omega) + (b\,;c\,;\omega)$. It is easy to check that $(a\,;c) \oplus (b\,;c)$ fr T but we do not have $(a + b)$ fr T. □

Remark. It is important to note that we do not have (P fr T and Q fr T) $\Longrightarrow$ $P \oplus Q$ fr T. Let us consider $P = a\,;c + b$, $Q = a\,;d$ and $T = b\,;\omega + a\,;d\,;\omega$. Then we have

$$\begin{array}{rl}
P \text{ fr } T: & P \xrightarrow{b},\ T \xrightarrow{b} \\
& P/b = \mathtt{STOP},\ T/b = \omega,\ \text{and } \mathtt{STOP} \text{ fr } \omega \\
Q \text{ fr } T: & Q \xrightarrow{a},\ T \xrightarrow{a} \\
& Q/a = d,\ T/a = d\omega,\ \text{and } d \text{ fr } d\omega \\
\neg(P \oplus Q \text{ fr } T): & P \oplus Q \rightarrowtail P,\ P \xrightarrow{a} \\
& P/a = c \oplus d,\ T/a = d\,;\omega \\
& c \oplus d \rightarrowtail c,\ \text{but we have not } c \text{ fr } d\omega
\end{array}$$

This is a new statement of the reason why in the second clause of Definition 3.3 we had to take the "global" continuations P/a and T/a, instead of the "local" ones P'/a and T'/a.

Next we present the relation between friendly and must test passing:

Proposition 3.9. *Let P and Q be processes and T be a (finite)[4] test. We have:*

1. *If P must T then P fr T.*
2. $P \sqsubseteq_{\text{must}} Q \implies P \sqsubseteq_{\text{fr}} Q$.

Proof.

1. We need only prove that must test passing can be recursively characterized by means of the definition of friendly test passing, where the existential quantification in the second clause is substituted by an universal quantification. We observe that whenever $P \parallel T$ is stable its set of complete computations can be defined by

$$\{P \parallel T \stackrel{as}{\Longrightarrow} P'' \parallel T'' \mid \exists a : P \parallel T \xrightarrow{a} P/a \parallel T/a \stackrel{s}{\Longrightarrow} P'' \parallel T''\}$$

If $P \parallel T$ is not stable we have that the set of complete computations of $P \parallel T$ is equal to

$$\{P \parallel T \stackrel{s}{\Longrightarrow} P'' \parallel T'' \mid \exists\, P', T' \text{ stable}: \ P \parallel T \rightarrowtail^* P' \parallel T' \stackrel{s}{\Longrightarrow} P'' \parallel T''\}$$

[4] Although plain testing is defined for arbitrary tests, here we will only consider finite tests, since friendly testing has only been defined (in this paper) for finite tests. Moreover, as it is well known, in the must case finite tests have the same discriminatory power as the full family of tests (this will be needed when proving the second part of this proposition).

and since each $P' \parallel T'$ is stable we can apply the characterization above to obtain that the set of computations of $P \parallel T$ is

$$\{P \parallel T \stackrel{as}{\Longrightarrow} P'' \parallel T'' \mid \exists\, P', T' \text{ stable}: \\ P \parallel T \rightarrowtail^* P' \parallel T' \stackrel{a}{\longrightarrow} P/a \parallel T/a \stackrel{s}{\Longrightarrow} P'' \parallel T''\} \cup \\ \{P \parallel T \rightarrowtail P' \parallel T' \mid P', T' \text{ stable}, \forall a \in \mathbf{Act}: P' \parallel T' \stackrel{a}{\not\longrightarrow}\}$$

If we now compare this set with the explored by the recursive characterization, we observe that the only difference is that P/a and T/a are used instead of P'/a and T'/a. But $P/a = \bigoplus\{P'' \mid P \stackrel{a}{\Longrightarrow} P''\}$ and the same is true for T/a. Then we have that the set of computations of $P/a \parallel T/a$ is the union of those of the systems $P'' \parallel T''$, and we have

$$\{P'' \parallel T'' \mid P \stackrel{a}{\Longrightarrow} P'',\ T \stackrel{a}{\Longrightarrow} T''\} = \\ \{P'' \parallel T'' \mid \exists\, P', T':\ P \rightarrowtail^* P',\ T \rightarrowtail^* T',\ P',\ T' \text{ stable} \\ P' \stackrel{a}{\longrightarrow},\ T' \stackrel{a}{\longrightarrow}, P'/a \rightarrowtail^* P'',\ T'/a \rightarrowtail^* T''\}$$

and then by induction on the depth of P we get the desired result.

2. It will be proved by applying the alternative characterization to be developed at the following section. □

Concluding this section we present a result showing that deterministic tests constitute indeed a set of *essential* tests.

Lemma 3.10. *For any process P and test T we have P fr T iff P fr $\overline{T}$, where $\overline{T}$ is the normal form of T with respect to the must semantics.*

Proof. By induction on the depth of T.

• $depth(T) = 1$ we can distinguish two cases:

– $\exists\, T'$ stable : $T \rightarrowtail^* T'$ such that $T' \stackrel{\omega}{\not\longrightarrow}$. Then it is clear that P fr T for no process P. Besides we have either $\overline{T} = \mathtt{STOP}$ or $\overline{T} = \mathtt{STOP} \oplus T''$ for some process T'', and in both cases there is no process P such that $\overline{T} \rightarrowtail^* \mathtt{STOP}$, and thus P fr $\overline{T}$.

– $\forall\, T'$ stable : $T \rightarrowtail^* T' \implies T' \stackrel{\omega}{\longrightarrow}$. Then it is clear that P fr T for any process P. In this case $\overline{T} = \bigoplus_{A \in \mathcal{A}} \sum_{a \in A} a$ and for any $A \in \mathcal{A}$ we have $\omega \in A$, therefore for any process P we have P fr $\overline{T}$.

• Now let us suppose that $depth(T) > 1$ and P fr T. Let us take P' and T' stable such that $P \parallel \overline{T} \rightarrowtail^* P' \parallel T'$. Since $\overline{T} = \bigoplus_{A \in \mathcal{A}} \sum_{a \in A} a\,;T_a$ we have $T' = \sum_{a \in A} a\,;T_a$ for some $A \in \mathcal{A}$. By the properties of normal forms we have that there exists T'' stable such that $T \rightarrowtail^* T''$ and $S(T'') \subseteq A$. As P fr T we have that either $T'' \stackrel{\omega}{\longrightarrow}$ or there exists some action $a \in \mathbf{Act}$ such that $P' \parallel T'' \stackrel{a}{\longrightarrow}$ and P/a fr T/a. In the first case we immediately get the result, because $\omega \in A$; while in the second one, we get the result by applying the induction hypothesis, since $a \in A$ and $\overline{T/a} = \overline{T}/a$. □

Lemma 3.11. *For any process P and any normal form test $T = \bigoplus_{A \in \mathcal{A}} \sum_{a \in A} a\,;T_a$ we have P fr T iff for any $A \in \mathcal{A}$ we have P fr T_A where $T_A = \sum_{a \in A} a;T_a$.*

Proof. We have that P fr T iff for all P', T' stable such that $P \parallel T \rightarrowtail P' \parallel T'$ we have either $T' \stackrel{\omega}{\longrightarrow}$ or there exists a such that $P' \parallel T' \stackrel{a}{\longrightarrow}$ and P/a fr T/a. But $\{T' \mid T \rightarrowtail^* T', T' \text{ stable}\} = \{T_A \mid A \in \mathcal{A}\}$ from which we immediately conclude the desired equivalence. □

Corollary 3.12. *Let P, Q be processes. We have $P \sqsubseteq_{\mathrm{fr}} Q$ iff for any deterministic test T, P fr T implies Q fr T.*

Proof. We need only prove the right to left implication. We have to see that for any test T we have P fr T implies Q fr T. By applying Lemma 3.10 it is enough to prove it for normal form tests. We will prove that it is indeed the case by induction on the depth of T. If $depth(T) = 1$ then we have $T = \omega$ and the result is immediate. Let $T = \bigoplus_{A \in \mathcal{A}} \sum_{a \in A} a\,;T_a$. By applying Lemma 3.11 we have that for any $A \in \mathcal{A}$, taking $T_A = \sum_{a \in A} a;T_a$, we have P fr T_A and (Q fr $T \iff \forall\, A \in \mathcal{A}\ Q$ fr T_A). Thus we have to see that P fr T_A implies Q fr T_A. For it, we apply the definition of friendly test passing to obtain that for any P' stable such that $P \rightarrowtail^* P'$ there exist $a \in A$ and a process P'' with $P' \stackrel{a}{\longrightarrow} P''$, such that P/a fr T_a. Now if for any deterministic test T such that P/a fr T we consider the test $T^a = a\,;T$ we have that P fr T^a and thus Q fr T^a, from which we conclude Q/a fr T. Thus we can apply the induction hypothesis to the processes P/a and Q/a, and since $depth(T_a) < depth(T)$ and P/a fr T_a we obtain Q/a fr T_a, and therefore Q fr T. □

4 Alternative Characterization of Friendly Testing

In this section we provide an alternative characterization of the friendly testing preorder defined in Definition 3.5. This characterization is based on a modification of the notion of acceptance sets [2] called *friendly acceptance sets*. The last result of the previous section will be very helpful in order to prove that the preorder induced by the alternative characterization is equivalent to $\sqsubseteq_{\text{fr}}$.

Definition 4.1. *Let P be a process. We define the friendly acceptance sets of P as:*

$$\mathcal{F}(P) = \{A \in \mathcal{A}(P,\epsilon) \mid \nexists A' \in \mathcal{A}(P,\epsilon) : \ A' \subsetneq A\}$$

Note that we have defined friendly acceptance sets of a process only for the empty trace; friendly acceptance sets for each trace $s = a_1, \ldots, a_n$ could be defined as the friendly acceptance sets of the process $((P/a_1)/a_2)\cdots/a_n$. By comparing the friendly acceptance sets of processes we can obtain a new preorder. This preorder is defined by adapting the preorder for acceptance sets to the new setting.

Definition 4.2. *Let P, P' be processes. We write $P \ll_{\text{fr}} P'$ iff for all $A' \in \mathcal{F}(P')$ there exists $A \in \mathcal{F}(P)$ such that $A \subseteq A'$, and for all $a \in A$, $P/a \ll_{\text{fr}} P'/a$.*

Note that an equivalent definition could be obtained by substituting $\mathcal{F}(P)$ and $\mathcal{F}(P')$ by $\mathcal{A}(P,\epsilon)$ and $\mathcal{A}(P',\epsilon)$, but we prefer to present the new notation to stress the fact that only minimal (thus friendly) acceptance sets play a role in the characterization of our friendly testing semantics, while for must testing both minimal and maximal ones, or equivalently all the sets in the convex closure, have to be taken into account.

Now we present the main result of this paper, where we prove that the preorders $\sqsubseteq_{\text{fr}}$ and $\ll_{\text{fr}}$ coincide. We consider separately the two inclusions.

Theorem 4.3. *Let P, P' be processes. Then $P \sqsubseteq_{\text{fr}} P'$ implies $P \ll_{\text{fr}} P'$.*

Proof. The proof will be done by the contrapositive, and by structural induction over processes. So let us suppose $P \not\ll_{\text{fr}} P'$. Then, there exists $A' \in \mathcal{F}(P')$ such that some of the following conditions hold:[5]

- $\forall A \in \mathcal{F}(P) : \ A \not\subseteq A'$, or
- $\forall A \in \mathcal{F}(P) : \Big(A \subseteq A' \implies \exists a_A \in A : \ P/a_A \not\ll_{\text{fr}} P'/a_A\Big)$.

In the first case we construct a set S including for each $A \in \mathcal{F}(P)$ one action in $A - A'$. Then, if we consider the deterministic test $T = \sum_{a \in S} a\,;\omega$, we get P fr T but we do not have P' fr T.

In the second case, by applying the induction hypothesis we can assume that for each $A \subseteq A'$ there exists a test T_{a_A} such that P/a_A fr T_{a_A}, but P'/a_A does not. Besides, for each $A'' \in \mathcal{F}(P)$ such that $A'' \not\subseteq A'$ we take $a_{A''} \in A'' - A'$, and we consider the deterministic test

$$T = \sum_{\substack{A \subseteq A' \\ A \in \mathcal{F}(P)}} a_A\,;T_{a_A} + \sum_{\substack{A'' \not\subseteq A' \\ A'' \in \mathcal{F}(P)}} a_{A''}\,;\omega$$

It is easy to check that P fr T but we do not have P' fr T, since each P'/a_A does not *friendly pass* the test T_{a_A}. □

Theorem 4.4. *Let P, P' be processes. Then $P \ll_{\text{fr}} P'$ implies $P \sqsubseteq_{\text{fr}} P'$.*

Proof. Let T be a deterministic test such that P fr T. We prove by induction on the depth of T that P' fr T.

If $depth(T) = 1$ then $T = \omega$ and the result is trivial. Otherwise $T = \sum_{i \in I} a_i\,;T_i$. Then, in order to check that P' fr T we have to show that for each $A' \in \mathcal{A}(P',\epsilon)$ there exists some $a' \in A'$ with $a' = a_i$, for some $i \in I$, and such that P/a' fr T_i. Given that for any $A' \in \mathcal{A}(P',\epsilon)$ there exists $A'' \in \mathcal{F}(P')$ such that $A'' \subseteq A'$, it is enough to prove the previous property for the sets in $\mathcal{F}(P')$.

Since $P \ll_{\text{fr}} P'$, we have that for any $A' \in \mathcal{F}(P')$ there exists $A \in \mathcal{F}(P)$ with $A \subseteq A'$ such that for all $a \in A : \ P/a \ll_{\text{fr}} P'/a$. By applying the hypothesis of the theorem we have P fr T, and thus there exists $a \in A$, with $a = a_i$ for some $i \in I$, such that P/a fr T_i. Then we can take $a' = a = a_i$, and by applying the induction hypothesis we obtain P'/a' fr T_i, and thus we conclude P' fr T. □

[5] The first case is just a particular case of the second, but we think that by considering first that particular case we contribute to make the proof more easily understandable.

Note that in the previous two theorems we have used that deterministic tests have the same discriminatory power as the whole family of tests (Corollary 3.12). The combination of the previous results gives us the final result.

Corollary 4.5. *Let P, P' be processes. Then $P \ll_{\mathrm{fr}} P' \iff P \sqsubseteq_{\mathrm{fr}} P'$.*

It is interesting to note the close relationship between the alternative characterization of the must testing semantics (based on acceptance sets) and that of friendly testing semantics. We begin by presenting an alternative characterization of the former derived from that of friendly testing.

Definition 4.6. *We write $P \ll' P'$ iff the following conditions hold:*

- $\forall A' \in \mathcal{F}(P')\ \exists A \in \mathcal{F}(P) : A \subseteq A' \ \wedge\ \forall a \in A' :\ P/a \ll' P'/a$,
- $S(P) \supseteq S(P')$, *and*
- $\forall a \in S(P') :\ P/a \ll' P'/a$.

It is clear that the second conjunct of the first clause is redundant with the third clause, but we present the characterization in this way in order to obtain a formulation as close as possible to the characterization of our friendly semantics. Next we prove that $\ll'$ indeed characterizes classical must testing equivalence.

Theorem 4.7. *Let P, P' be processes. Then, $P \sqsubseteq P'$ iff $P \ll' P'$.*

Proof. It is enough to prove

$$P \ll' P' \quad \Longleftrightarrow \quad \forall A' \in \mathcal{A}(P', s)\ \exists A \in \mathcal{A}(P, s) :\ A \subseteq A'$$

The left to right implication is straightforward by induction over the length of s. For the right to left implication it is enough to note

$$A \in \mathcal{A}(P/a, s) \iff A \in \mathcal{A}(P, as) \qquad \square$$

This means that the difference between must and friendly testing, when comparing two processes P and P', comes from the more restrictive conditions in the characterization above. Thus, if $P \sqsubseteq_{\mathrm{fr}} P'$ but $P \not\sqsubseteq P'$ we have either:

1. We can satisfy the characterization of the friendly testing preorder, that means we have $\forall A'_n \in \mathcal{F}(P')\ \exists A_n \in \mathcal{F}(P) :\ A_n \subseteq A'_n \ \wedge\ \forall a \in A_n :\ P/a \ll_{\mathrm{fr}} P'/a$, but anyhow we take the sets A_n satisfying the previous condition, there exists some $a \in \left(\bigcup_{\mathcal{F}(P')} A'_n\right) - \left(\bigcup A_n\right)$ such that $P/a \not\ll' P'/a$.

   Example. $P = a \oplus (b\,;c)$, $P' = a + b$. $P/b = c$, $P'/b = \mathtt{STOP}$. So, we have $P/b \not\ll' P'/b$. Note that $P/b \not\ll_{\mathrm{fr}} P'/b$, but even so $P \sqsubseteq_{\mathrm{fr}} P'$.

2. $S(P) \not\supseteq S(P')$.

   Example. $P = a$, $P' = a + b$.

3. $\exists a \in (S(P') - \bigcup_{\mathcal{F}(P)} A') :\ P/a \not\ll' P'/a$.

   Example. $P = a \oplus (b\,;c)$, $P' = a \oplus (a + b)$. $P/b = c$, $P'/b = \mathtt{STOP}$.

It is easy to check that case 2 can be considered as a particular case of the other two cases, and several of these cases can be presented at the same time when comparing a pair of processes. Then, we can say that the stronger power of must testing comes from the possibility to check (and to punish for it) traces, as well as the continuations after those actions that are not informative for friendly testing.

As an immediate consequence we obtain the proof of Proposition 3.9 item 2.

Corollary 4.8. $P \sqsubseteq_{\mathrm{must}} Q \implies P \sqsubseteq_{\mathrm{fr}} Q$.

Proof. By a straightforward induction on the depth of P and Q we prove $P \ll' P'$ implies $P \ll_{\mathrm{fr}} Q$ because the first condition in Definition 4.2 is weaker than the first condition in Definition 4.6. Then we need only apply Theorem 4.7 and Corollary 4.5 to conclude. $\square$

5 Conclusions and Forthcoming Work

Currently we are developing a full theory of friendly testing similar to that for classical testing[2]. So we have extended our definition to cover recursive processes and provided both a denotational model and a complete axiomatization. In order to obtain both we have found an important technical

problem: Friendly testing equivalence is not substitutive for arbitrary contexts. More exactly, we have realized that $\sqsubseteq_{\text{fr}}$ is not a precongruence with respect to the external choice operator, as the following example shows:

$$\texttt{STOP} \oplus (b\,;P) \approx_{\text{fr}} \texttt{STOP}$$
$$(\texttt{STOP} \oplus b\,;P) + (b\,;Q) \approx_{\text{fr}} b\,;(P+Q) \not\approx_{\text{fr}} b\,;P$$

The problem disappears if there are no interferences between the offerings of the two involved processes.

Unfortunately, recently we have discovered that the situation is even worse: $\sqsubseteq_{\text{fr}}$ is not a precongruence with respect to the internal choice operator. We have long been working on the basis that the internal choice operator preserved the relation $\sqsubseteq_{\text{fr}}$, and in [6] we indeed based the denotational model and the complete axiomatization on this erroneous fact. The reason for this misunderstanding was that even if we noticed that we needed a more elaborate definition for the friendly passing of tests (see the discussion after Definition 3.3) we have been informally reasoning with the old definition, under which $\sqsubseteq_{\text{fr}}$ would be clearly a precongruence with respect to the internal choice operator. But under the revised definition this is no more the case, as the following example shows. On the one hand we have:

$$a\,;P \oplus (a\,;P + b\,;Q) \approx_{\text{fr}} a\,;P$$

We also have

$$(a\,;P \oplus (a\,;P+b\,;Q)) \oplus b\,;Q' \approx_{\text{fr}} (a\,;P \oplus (a\,;P+b\,;Q)) \oplus b\,;(Q \oplus Q')$$

We have that the last process is also equivalent to $a\,;P \oplus b\,;(Q \oplus Q')$, but this process is not equivalent to $a\,;P \oplus b\,;Q'$, so we finally get

$$(a\,;P \oplus (a\,;P+b\,;Q)) \oplus b\,;Q' \not\approx_{\text{fr}} a\,;P \oplus b\,;Q'$$

Therefore the situation is similar to that for the external choice operator, but there is a subtle difference: In order to obtain a counterexample we need to consider a process not offering the empty set, since for any process P we have $\texttt{STOP} \oplus P \approx_{\text{fr}} \texttt{STOP}$. So we have found that STOP is a singularity of our friendly testing theory.

Concerning the denotational semantics, it is obvious that we cannot obtain a fully abstract model, since friendly testing equivalence is not substitutive. At most we could obtain such a model for the weakest precongruence $\sqsubseteq_{\text{frext}}$ stronger than $\sqsubseteq_{\text{fr}}$. We have defined such a model, which is obtained from the model of acceptance trees characterizing must testing semantics, by closing in the adequate way the set of states labeling all the nodes of the tree but their roots.

To get the axiomatization for both $\sqsubseteq_{\text{frext}}$ and $\sqsubseteq_{\text{fr}}$ we have seen that it is not possible anymore to obtain an isolated axiomatization of these relations. Instead we have a two-levels axiomatization, including the axiomatization of the must equivalence $\approx_{\text{must}}$. Then, to get a sound and complete axiomatization of $\sqsubseteq_{\text{frext}}$ we need only to add the rules

$$\textbf{(STR)}\ \frac{P \leq Q}{P \leq_{\text{frext}} Q}\,,\quad \textbf{(FREXT)}\ \ a;\texttt{STOP} \leq_{\text{frext}} a;(\texttt{STOP} \oplus P)$$

In order to axiomatize $\sqsubseteq_{\text{fr}}$ we add instead the rules

$$\textbf{(STR)}\ \frac{P \leq Q}{P \leq_{\text{fr}} Q}\,,\quad \textbf{(FR)}\ \frac{\forall a \in \bigcup_{A \in \mathcal{A}} A : P_a \leq_{\text{fr}} Q_a}{\bigoplus_{A \in \mathcal{A}} \sum_{a \in A}(a;P_a) \leq_{\text{fr}} \bigoplus_{A \in \mathcal{A}} \sum_{a \in A \cup A'}(a;Q_a)}$$

Rule **(STR)** expresses the fact that the must testing preorder is stronger than the friendly one, while by applying Rule **(FR)** we obtain the friendly closure capturing the notion of friendly acceptance sets.

Thus we have seen that friendly testing is a *smooth* version of must testing. More exactly the induced testing preorder is weaker than the ordinary must preorder, and this is how the desired behavior as a nice conformance relation is obtained.

We have explored in detail the relationship between the conformance relation and our friendly testing in [4]. There we have proved that $\sqsubseteq_{\mathrm{fr}}$ is equal to $\sqsubseteq^{*}_{\mathrm{conf}}$, which is the transitive closure of $\sqsubseteq_{\mathrm{conf}}$[7], and thus the strongest order relation weaker than $\sqsubseteq_{\mathrm{conf}}$.

Acknowledgments We would like to thank Natalia López Barquilla and the anonymous referee for their careful reading, comments and suggestions that have contributed to improve the presentation of the paper.

References

[1] de Nicola R, Hennessy M C B. Testing equivalences for processes. *Theoretical Computer Science*, 1984, 34: 83–133.

[2] Hennessy M. *Algebraic Theory of Processes.* MIT Press, 1988.

[3] Brinksma E, Scollo G, Steenbergen C. LOTOS specifications, their implementations and their tests. In *Proceedings of Protocol Specification, Testing and Verification VI*, North-Holland, 1987, pp.349–360.

[4] de Frutos-Escrig D, Llana-Díaz L F, Núñez M. Friendly testing as a conformance relation. In *Proceedings of Formal Description Techniques for Distributed Systems (X) and Communication Protocols, and Protocol Specification, Testing, and Verification (XVII)*, Chapman & Hall, 1997, pp.283–298.

[5] Hoare C A R. *Communicating Sequential Processes.* Prentice Hall, 1985.

[6] de Frutos-Escrig D, Llana-Díaz L F, Núnez M. Introducing friendly testing. Technical Report DIA 53/97, Dept. Informática y Automática, Universidad Complutense de Madrid, 1997.

[7] Leduc G. A framework based on implementation relations for implementing LOTOS specifications. *Computer Networks and ISDN Systems*, 1992, 25(1): 23–41.

[8] Hennessy M. Acceptance trees. *Journal of the ACM*, 1985, 32(4): 896–928.

David de Frutos-Escrig graduated in pure mathematics and computer science in 1981 from the Universidad Complutense de Madrid (UCM) and achieved the Ph.D. degree in mathematics (computer science) in 1985 with a thesis devoted to "Denotational Semantics of Probabilistic Constructions (Probabilistic Powerdomains)". He has been a Professor since 1991 at the Computer Science Department of UCM, and heads the research group published papers on the subject.

Luis Llana-Díaz graduated in mathematics (computer science) in 1991 from the Universidad Complutense de Madrid (UCM) and achieved the Ph.D. degree in computer science with a thesis devoted to "Testing Semantics for Timed Processes Algebras" in 1996. He has been an Associate Professor since 1997 at the Computer Science Department of UCM. His main research area is formal models of concurrency and, more specifically, the semantics of timed processes.

Manuel Núñez graduated in mathematics (computer science) in 1992 from the Universidad Complutense de Madrid (UCM) and achieved the Ph.D. degree in computer science with a thesis devoted to "Testing Semantics for Probabilistic Process Algebras" in 1996. He has been an Associate Professor since 1997 at the Computer Science Department of UCM. His main research area is formal models of concurrency and, more specifically, the semantics of probabilistic processes.