Divergence and Fair Testing*

V. Natarajan   Rance Cleaveland

Abstract

This paper develops a new testing-based semantic theory of processes that aims to circumvent difficulties that traditional testing/failures theories have in dealing with divergent behavior. Our framework incorporates a notion of *fairness* into the determination of when a process passes a test; we contrast this definition with existing approaches and give characterizations of the induced semantic preorders. An example highlights the utility of our results.

1 Introduction

Research into algebraic models of concurrency has focused on the use of semantic equivalences and preorders for establishing that systems meet their specifications. In such an approach to verification one formulates a specification as a system describing the required high-level behavior; a design/implementation then meets such a specification if its behavior is indistinguishable from the specification's (if one is using an equivalence) or if its behavior is in some sense better than the specification's (if one is using a preorder). A variety of different equivalences and preorders have been suggested for this purpose, with the testing/failures relations [4, 8] attracting particular attention because of their intuitive formulations in terms of the responses a system exhibits to tests. The theory underlying this framework has been extensively investigated, and several case studies attest to the utility of these relations in verification [1, 5].

One aspect of process behavior that the testing/failures framework is particularly sensitive to is *divergence*. A divergent process may engage in an infinite sequence of internal computation steps; in doing so, it ignores requests for interaction offered by its environment. As such behavior is in general undesirable, the testing/failures relations ensure that processes that may diverge are "not as good" as those that will not. However, in some cases processes deemed divergent by this framework may be viewed as unproblematic. As an example, consider a communications protocol that is designed to operate over a lossy medium. The system consisting of the protocol and this medium would exhibit a possible infinite internal computation in which the medium loses every message that the sender gives to it. In reality, however, such behavior would in all likelihood not occur; if a lossy medium is indeed capable of delivering messages, then it will certainly deliver some of the ones it is given, and the system just described ought not to be considered divergent. The traditional testing/failures framework does not admit such distinctions, however, and this fact complicates the use of the theory in reasoning about communications protocols in particular, and distributed fault-tolerant systems in general.

---

*To appear in *ICALP '95*. Research supported by NSF grant CCR-9120995, ONR Young Investigator Award N00014-92-J-1582, NSF Young Investigator Award CCR-9257963, and NSF grant CCR-9402807. Authors' address: Department of Computer Science, North Carolina State University, Raleigh, NC 27695-8206, USA, e-mail: {rance, raj}@science.csc.ncsu.edu.

$$
\begin{array}{rcll}
S_0 & \Leftarrow & send.S'_0 & S'_0 \Leftarrow \overline{s_0}.(r_{ack_0}.S_1 + r_{ack_1}.S'_0 + \tau.S'_0) \\
S_1 & \Leftarrow & send.S'_1 & S'_1 \Leftarrow \overline{s_1}.(r_{ack_1}.S_0 + r_{ack_0}.S'_1 + \tau.S'_1) \\
\\
R_0 & \Leftarrow & r_0.R'_0 + r_1.\overline{s_{ack_1}}.R_0 + \tau.\overline{s_{ack_1}}.R_0 & R'_0 \Leftarrow receive.\overline{s_{ack_0}}.R_1 \\
R_1 & \Leftarrow & r_1.R'_1 + r_0.\overline{s_{ack_0}}.R_1 + \tau.\overline{s_{ack_0}}.R_1 & R'_1 \Leftarrow receive.\overline{s_{ack_1}}.R_0 \\
\\
M & \Leftarrow & s_0.(\overline{r_0}.M + \tau.M) + s_1.(\overline{r_1}.M + \tau.M) & \\
 & & +s_{ack_0}.(\overline{r_{ack_0}}.M + \tau.M) + s_{ack_1}.(\overline{r_{ack_1}}.M + \tau.M) & \\
M' & \Leftarrow & s_0.\overline{r_0}.M' + s_0.M' + s_1.\overline{r_1}.M' + s_1.M' & \\
 & & +s_{ack_0}.\overline{r_{ack_0}}.M' + s_{ack_0}.M' + s_{ack_1}.\overline{r_{ack_1}}.M' + s_{ack_1}.M' & \\
\\
ABP & \Leftarrow & (S_0|M|R_0)\backslash\{r_0, r_1, s_0, s_1, r_{ack_0}, r_{ack_1}, s_{ack_0}, s_{ack_1}\} & \\
\\
ABP' & \Leftarrow & (S_0|M'|R_0)\backslash\{r_0, r_1, s_0, s_1, r_{ack_0}, r_{ack_1}, s_{ack_0}, s_{ack_1}\} &
\end{array}
$$

Subscripted s and r actions denote sends to and receives from the medium.

Figure 1: The alternating bit protocol.

In this paper we develop a theory of testing that incorporates a notion of *fairness* into process testing. Intuitively, in our framework a process is deemed to "pass" a test if it may come "arbitrarily close" to success; using this notion as a basis we characterize the resulting semantic preorders and show how, as a side-effect, a natural treatment of divergence emerges that permits reasoning about systems with faulty components. The rest of the paper develops along the following lines. In the next section we offer an example designed to illustrate in a concrete manner the shortcomings of the traditional view of divergence. The section following then presents the formal definitions of traditional testing and gives our new framework, while Section 4 examines the relationship between the old and the new testing approaches. The next section offers an alternative characterization of the testing relations that do not require analyzing the behavior of processes in response to tests and re-examines the example of Section 2 in light of our new relations. The last section contains our conclusions and directions for future work as well as comparisons and contrasts with existing research on this topic.

2 Motivating Example

Traditional testing/failures semantics relates processes on the basis of their capabilities for deadlock and divergence; if two processes are equivalent then their behavior in these regards should be identical. These theories typically view divergence as "catastrophic" in the sense that once a process exhibits a capacity for infinite internal behavior, its subsequent deadlocking behavior becomes irrelevant. This insensitivity to deadlock in the face of divergence can lead to counterintuitive determinations regarding process equivalence, as the example presented in this section illustrates.

The system we consider is the Alternating Bit Protocol (ABP), a communications protocol designed to ensure reliable communication over lossy communication lines. Figure 1 contains a rendering of the ABP in CCS [13], some familiarity with which we assume on the part of the reader. The ABP consists of three entities: a sender, a receiver, and a lossy medium over which the sender and receive communicate. In order to recover from the possibility of message losses, the sender and receiver exchange messages and acknowledgements that are tagged with "sequence bits" in the following manner. When the sender is given a message to deliver, it appends its current sequence bit value i (initially $i = 0$) to the message and gives the resulting packet to the medium (by performing the action $\overline{s_i}$). It then awaits an acknowledgement from the receiver that is also tagged with i (by waiting to perform action r_{ack_i}). If it receives such an acknowledgement the sender increments its sequence bit modulo 2 and awaits for the next message to deliver. If the sender receives an acknowledgement with the wrong bit, or times out (by performing its τ action), then it resends its message. The receiver behaves in a dual fashion; it awaits a message from the medium whose sequence bit matches its current sequence bit i (initially $i = 0$), and if it receives one it delivers it (by executing its $\overline{receive}$ action), sends an acknowledgement tagged with i (by performing $\overline{s_{ack_i}}$), and increments its sequence bit modulo 2. If it receives a message with the wrong sequence bit, or if it times out, then the receiver sends an acknowledgement with sequence bit $1-i$. In the figure, S_i and R_i represent the "start states" of the sender and receiver, respectively, when their sequence bits are i.

In this example we consider two slightly different faulty media. Medium M may lose messages given to it (by performing τ) at any point until the message is received. M', on the other hand, may only drop a message at the time it is given to it; once it decides to deliver a message, it cannot lose it. ABP represents the ABP built using medium M, while ABP' represents the ABP built using M'.

From the operational semantics of CCS, it is straightforward to establish that both ABP and ABP' are initially capable of an infinite sequence of internal τ-steps. This results from the fact that the receiver may engage in an unbounded number of time-outs, which prompt it to send acknowledgements that the medium then loses. Accordingly both are divergent, and one may establish that, with respect to testing/failures equivalence, the two systems are indistinguishable. However, ABP is incapable of deadlocking; at any state it reaches during its execution it always has the potential of engaging in future $send$ and $\overline{receive}$ actions. On the other hand, ABP' does have reachable deadlocked states; for example, after executing a $send$ followed by several τ-actions, the system may wind up in the state

$$(S'_0 | \overline{r_{ack_1}}.M' | \overline{r_{ack_1}}.R_0) \backslash \{r_0, r_1, s_0, s_1, r_{ack_0}, r_{ack_1}, s_{ack_0}, s_{ack_1}\}.$$

In this state S'_0 and $\overline{r_{ack_1}}.R_0$ can only advance by sending a message/acknowledgement to the medium, which in turn is waiting to deliver an acknowledgement to the sender.

Note that the fact that these systems are divergent relies on the potential inability of the medium to deliver messages given to it. Under realistic assumptions about medium behavior, however, this would not happen; some messages will eventually be delivered. Therefore, one could argue that when such "fairness" considerations are taken into account, neither ABP nor ABP' should be considered divergent. Were one able to make this determination, then one might in fact ascertain that ABP is "strictly better", with respect to the failures/testing preorders, than ABP' because of its better deadlock behavior. Unfortunately, the traditional testing/failure framework requires that ABP and ABP' be considered divergent, and hence the semantic relations are insensitive to these systems' differing potential for deadlock.

3 Processes

This section formally defines the model of processes used in this paper, reviews relevant concepts from traditional testing theory, and introduces a new approach to testing.

3.1 Processes and Tests

We model processes using *labeled transition systems*, which are defined as follows.

Definition 3.1 *A* labeled transition system *is a tuple* $\langle P, A, \rightarrow, p_0\rangle$ *where* P *is a set of* states, *A is a set of* actions *containing a distinguished* internal *action* τ, $\rightarrow \subseteq P \times A \times P$ *is the* transition relation, *and* $p_0 \in P$ *is the* start state.

Intuitively, a labeled transition system encodes the operational behavior of a process. P consists of the set of states that the process may enter, and A contains the set of actions that it may perform. The relation $\rightarrow$ describes the actions available in a particular state and the states that may result when the action is executed. In what follows we write $p \xrightarrow{a} p'$ in lieu of $\langle p, a, p'\rangle \in\rightarrow$, and we also use $p \xrightarrow{a}$ to indicate that there is a p' such that $p \xrightarrow{a} p'$ and $p \not\xrightarrow{a}$ if no such p' exists. Also, $\rightarrow$ may be extended to sequences $s \in A^*$ in the obvious fashion. To simplify the presentation, we restrict our attention to *finite-sort* transition systems, namely, systems $\langle P, A, \rightarrow, p_0\rangle$ such that A is finite. We do not, however, require that processes be finite-branching.

3.2 Traditional Testing

The traditional testing framework of [7, 8, 11] defines preorders and equivalences that relate processes on the basis of their responses to tests. In their setting a test is any process whose action set contains a distinguished *success* action w; to determine a process' response(s) to a test one must examine the computations that result when the test is "applied" to the process. These intuitions may be formalized as follows.

Definition 3.2 *Let process* $\mathcal{P} = \langle P, A, \rightarrow, p_0\rangle$ *and test* $\mathcal{T} = \langle T, A' \cup \{w\}, \rightarrow, t_0\rangle$.

1. *A* configuration *has form* $p||t$, *where* $p \in P$ *and* $t \in T$. $p||t$ *is* successful *if* $t \xrightarrow{w}$.
2. *The* configuration transition relation, $p||t \xmapsto{a} p'||t'$, *is defined as follows.*
   - $p \xrightarrow{a} p', t \xrightarrow{a} t', t \not\xrightarrow{w} \;\Rightarrow p||t \xmapsto{a} p'||t'$.
   - $p \xrightarrow{\tau} p', t \not\xrightarrow{w} \;\Rightarrow p||t \xmapsto{\tau} p'||t$.
   - $t \xrightarrow{\tau} t', t \not\xrightarrow{w} \;\Rightarrow p||t \xmapsto{\tau} p||t'$.

   We sometimes write $p||t \longmapsto p'||t'$ *if* $p||t \xmapsto{a} p'||t'$ *for some* $a \in A \cap A'$.
3. *A* computation *of* $p||t$ *is a maximal (potentially infinite) sequence* $p||t \equiv p_1||t_1 \xmapsto{a_1} p_2||t_2 \xmapsto{a_2} \cdots$; *it is* strongly successful *if some* $p_i||t_i$ *is successful.*
4. p *may*$_{DH}$ t *if some computation of* $p||t$ *is strongly successful.* $\mathcal{P}$ *may*$_{DH}$ $\mathcal{T}$ *if* p_0 *may*$_{DH}$ t_0. p *must*$_{DH}$ t *if every computation of* $p||t$ *is strongly successful.* $\mathcal{P}$ *must*$_{DH}$ $\mathcal{T}$ *if* p_0 *must*$_{DH}$ t_0.

As processes and tests are in general nondeterministic, one may distinguish between the possibility of success ($\mathcal{P}$ *may*$_{DH}$ $\mathcal{T}$) and the inevitability of success ($\mathcal{P}$ *must*$_{DH}$ $\mathcal{T}$). Also, the definition of $\longmapsto$ departs slightly from the one given in [8] in a couple of respects. Firstly, our definition labels $\longmapsto$ with actions. More importantly, it also adds the premise "$t \not\xrightarrow{w}$" to all the rules used in defining the relation. The latter alteration ensures that successful configurations have no outgoing transitions; this property turns out to be technically convenient and yet does not affect the resulting semantic theory. The following preorders may now be defined.

Definition 3.3 *Let* $\mathcal{P}$ *and* $\mathcal{P}'$ *be processes.*

1. $\mathcal{P} \sqsubseteq_{may_{DH}} \mathcal{P}'$ *if for every test* $\mathcal{T}$ *such that* $\mathcal{P}$ *may*$_{DH}$ $\mathcal{T}$, $\mathcal{P}'$ *may*$_{DH}$ $\mathcal{T}$.

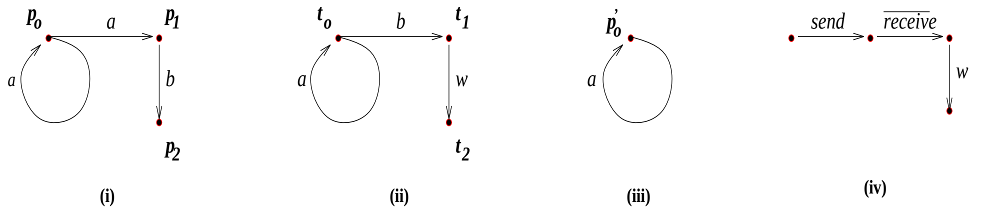

Figure 2: Sample processes and tests

2. $\mathcal{P} \sqsubseteq_{must_{DH}} \mathcal{P}'$ *if for every test* $\mathcal{T}$ *such that* $\mathcal{P}$ $must_{DH}$ $\mathcal{T}$, $\mathcal{P}'$ $must_{DH}$ $\mathcal{T}$.

A number of alternative characterizations have been developed to ease the task of reasoning using these preorders. Here we present one due to De Nicola and Hennessy [8].

Definition 3.4 *Let* $\mathcal{P} = \langle P, A, \rightarrow, p_0 \rangle$ *be a process, with* $p, p' \in P$. *Let* $s \in (A - \{\tau\})^*$ *be a sequence of visible actions.*

1. $p \stackrel{s}{\Rightarrow} p'$ *is defined inductively as follows:* $p \stackrel{\epsilon}{\Rightarrow} p'$ *if* $p(\stackrel{\tau}{\rightarrow})^* p'$ *and* $p \stackrel{as'}{\Rightarrow} p'$ *if* $p \stackrel{\epsilon}{\Rightarrow} \stackrel{a}{\rightarrow} \stackrel{s'}{\Rightarrow} p'$.

2. *The* language, $L(p)$, *of* p *is defined by:* $L(p) = \{\, s \mid \exists p'.p \stackrel{s}{\Rightarrow} p' \,\}$. *We sometimes refer to the elements of* $L(p)$ *as* traces *of* p *and take* $L(\mathcal{P}) = L(p_0)$.

3. $\Downarrow \subseteq P$ *is defined as the least predicate such that whenever* $p' \Downarrow$ *for all* p' *such that* $p \stackrel{\tau}{\rightarrow} p'$, *then* $p \Downarrow$.

4. $p \Downarrow s$ *is defined inductively as follows:* $p \Downarrow \epsilon$ *if* $p \Downarrow$, *and* $p \Downarrow as'$ *if* $p \Downarrow$ *and for all* p' *such that* $p \stackrel{a}{\Rightarrow} p', p' \Downarrow s'$.

5. $S(p) = \{\, a \in (A - \{\tau\}) \mid p \stackrel{a}{\Rightarrow} \}$.

Intuitively, $L(p)$ contains the sequences of visible actions that p may engage in, while $p \Downarrow$ holds if p is incapable of an infinite sequence of internal computation steps. $p \Downarrow s$ indicates that during any "execution" of s from p, no infinite internal computation sequences are possible. We now have the following [11].

Theorem 3.5 *Let* $\mathcal{P} = \langle P, A, \rightarrow, p_0 \rangle$ *and* $\mathcal{P}' = \langle P', A', \rightarrow, p_0' \rangle$ *be processes.*

1. $\mathcal{P} \sqsubseteq_{may_{DH}} \mathcal{P}'$ *iff* $L(p_0) \subseteq L(p_0')$.

2. $\mathcal{P} \sqsubseteq_{must_{DH}} \mathcal{P}'$ *iff for every* $s \in (A' - \{\tau\})^*$ *with* $p_0 \Downarrow s$, $p_0' \Downarrow s$ *and for any* p_1' *such that* $p_0' \stackrel{s}{\Rightarrow} p_1'$, *there is a* p_1 *such that* $p_0 \stackrel{s}{\Rightarrow} p_1$ *and* $S(p_1) \subseteq S(p_1')$.

3.3 Fair Testing

The definitions given above have a certain intuitive appeal: a computation is strongly successful if one observes a successful configuration at some point along the computation. However, consider the process $\mathcal{P}$ in Figure 2(i) and test $\mathcal{T}$ in Figure 2(ii). $\mathcal{P}$ $\not{m}ust_{DH}$ $\mathcal{T}$, since the computation $p_0 || t_0 \stackrel{a}{\longmapsto} p_0 || t_0 \stackrel{a}{\longmapsto} \cdots$ is not strongly successful. However, this computation may be viewed as "pathological" in that it "discriminates" against success: although unsuccessful, each configuration along this computation is also a starting point for the strongly successful computation $p_0 || t_0 \stackrel{a}{\longmapsto} p_1 || t_0 \stackrel{b}{\longmapsto} p_2 || t_1$. We now offer an alternative definition of successful computation that aims at a different treatment of such "unfair" computations.

Definition 3.6 *Let* $\mathcal{P} = \langle P, A, \rightarrow, p_0\rangle$ *be a process and* $\mathcal{T} = \langle T, A' \cup \{w\}, \rightarrow, t_0\rangle$ *a test. A configuration* $c \equiv p||t$ *is* potentially successful *if there is a successful configuration* c' *such that* $c \longmapsto^* c'$. *A computation is* potentially successful *if every configuration in it is potentially successful.*

If we replace the term "strongly successful" by "potentially successful" in Definition 3.2(4) then we arrive at definitions for p *may* t, p *must* t, $\mathcal{P}$ *may* $\mathcal{T}$ and $\mathcal{P}$ *must* $\mathcal{T}$. That is, p *may* t if $p||t$ has a potentially successful computation, etc. These definitions induce the preorders $\sqsubseteq_{may}$ and $\sqsubseteq_{must}$ on processes through the obvious alterations to Definition 3.3.

Note that it is possible for a computation to be potentially successful even though it is not strongly so: the computation $p_0||t_0 \overset{a}{\longmapsto} p_0||t_0 \overset{a}{\longmapsto} \cdots$ in the previous example has this property. Indeed, we may show that $\mathcal{P}$ *must* $\mathcal{T}$. The following lemma shows the relationship between *may* and *must* and their traditional counterparts.

Lemma 3.7 *Let* $\mathcal{P}$ *be a process and* $\mathcal{T}$ *be a test. Then* $\mathcal{P}$ *may* $\mathcal{T}$ *iff* $\mathcal{P}$ may_{DH} $\mathcal{T}$*, while* $\mathcal{P}$ $must_{DH}$ $\mathcal{T}$ *implies* $\mathcal{P}$ *must* $\mathcal{T}$.

The previous example shows that in general, $\mathcal{P}$ *must* $\mathcal{T}$ does not imply $\mathcal{P}$ $must_{DH}$ $\mathcal{T}$.

4 A Topological Comparison of $must_{DH}$ and $must$

In the remainder of this paper we give alternative characterizations of the preorders induced by *may* and *must*. Before doing so, however, we first wish to characterize precisely the distinction between the traditional notion of $must_{DH}$ and our new notion of *must*. In general, for a given process and test the set of potentially successful computations properly contains the set of strongly successful computations. Define a computation to be *weakly successful* if it is potentially successful but not strongly successful. Our aim in this section is two-fold: to show that the set of potentially successful computations is a "reasonable" extension of the set of strongly successful ones and that the set of weakly successful computations is "small".

To formalize these notions we first define a metric space over the computations of a process and test in which weakly successful computations may be viewed as limits of sequences of strongly successful computations. We then show that the set of weakly successful computations is *nowhere dense* in the set of computations. This amounts to saying that the "volume" of the weakly successful computations is "infinitesimal".

Definition 4.1 *A* metric space *is a pair* $\langle X, d\rangle$*, where* X *is a non-empty set and* d*, the* metric, *is a mapping from* $X \times X$ *to the non-negative reals satisfying the following properties, for all* $x, y, z \in X$.

1. $d(x, y) = d(y, x)$.
2. $d(x, y) = 0$ *iff* $x = y$.
3. $d(x, z) \leq d(x, y) + d(y, z)$.

If $x_0 \in X$ *and* $\epsilon > 0$ *is real then* $B_\epsilon(x_0)$*, the* open ball *of radius* ϵ *centered at* x_0*, is the set* $\{\, x \in X \mid d(x_0, x) < \epsilon\}$. *Set* $U \subseteq X$ *is an* open set *if for any* $x \in U$*, there exists* $\epsilon > 0$ *such that* $B_\epsilon(x_0) \subseteq U$. *Set* $F \subseteq X$ *is* closed *if* F^c*, its complement in* X*, is open. The* interior, U^0*, of a set* $U \subseteq X$ *is the largest open set contained in* U. *The* closure *of a set* $F \subseteq X$ *is the smallest closed set containing* F*; it will be denoted by* $\overline{F}$. *A subset* $A \subseteq X$ *is* nowhere dense *if* $(\overline{A})^0 = \emptyset$. *Note that a set* $U \subseteq X$ *has nonempty interior iff there is some* $x \in U$ *and a positive real* ϵ *such that* $B_\epsilon(x) \subseteq U$. *A sequence of elements* $x_1, x_2, \ldots$ *from* X converges *to* $x \in X$ *if for every* $\epsilon > 0$, $B_\epsilon(x)$ *contains all but a finite number of the* x_i*; in this case,* x *is said to be the* limit *of the sequence.*

The following defines two metric spaces over the computations of a process and test.

Definition 4.2 *Let $\mathcal{P}$ be a process with start state p_0 and $\mathcal{T}$ a test with start state t_0, and let $\mathcal{C}(\mathcal{P},\mathcal{T})$ be the set of computations of $p_0||t_0$. If $x \in \mathcal{C}(\mathcal{P},\mathcal{T})$ and n is a natural number then $x\langle n\rangle$ is the prefix of x of the form $p_0||t_0 \overset{a_0}{\longmapsto} \cdots \overset{a_{n-2}}{\longmapsto} p_{n-1}||t_{n-1}$; if x contains fewer than n configurations then $x\langle n\rangle = x$. Let $x, y \in \mathcal{C}(\mathcal{P},\mathcal{T})$; then $d_1(x,y)$ and $d_2(x,y)$ are defined as follows. $d_1(x,y)$ is 1 if $x \neq y$ and is 0 otherwise. $d_2(x,y) = \inf\{\, 2^{-n} \mid x\langle n\rangle = y\langle n\rangle \,\}$.*

It is easy to see that $\langle \mathcal{C}(\mathcal{P},\mathcal{T}), d_1\rangle$ and $\langle \mathcal{C}(\mathcal{P},\mathcal{T}), d_2\rangle$ are metric spaces for any $\mathcal{P}$ and $\mathcal{T}$. The next lemma establishes that weakly successful computations may be seen as *limits* of sequences of strongly successful ones. Therefore, although a weakly successful computation is not strongly successful, it comes "arbitrarily close" to being so.

Lemma 4.3 *Let $\mathcal{P}$ be a process and $\mathcal{T}$ be a test, and let $x \in \mathcal{C}(\mathcal{P},\mathcal{T})$ be weakly successful. Then there exists a sequence $x_1, x_2, \ldots$ in $\langle \mathcal{C}(\mathcal{P},\mathcal{T}), d_2\rangle$ converging to x such that each x_i is strongly successful.*

We now are able to establish the following metric-based characterizations for $must_{DH}$ and $must$. Each says in essence says that a process $must_{DH}/must$ a test exactly when "almost all" the computations (in the appropriate metric space) are successful.

Theorem 4.4 *Let $\mathcal{P}$ be a process and $\mathcal{T}$ a test, and let $\mathcal{S} \subseteq \mathcal{C}(\mathcal{P},\mathcal{T})$ be the set of strongly successful computations. Define $\mathcal{F} = \mathcal{C}(\mathcal{P},\mathcal{T}) - \mathcal{S}$. Then $\mathcal{P}$ $must_{DH}$ $\mathcal{T}$ iff $\mathcal{F}$ is nowhere dense in $\langle \mathcal{C}, d_1\rangle$, and $\mathcal{P}$ must $\mathcal{T}$ iff $\mathcal{F}$ is nowhere dense in $\langle \mathcal{C}, d_2\rangle$.*

5 Characterizing the Preorders

This section provides alternative characterizations of $\sqsubseteq_{may}$ and $\sqsubseteq_{must}$ that are analogous to those for the traditional preorders given in Section 3. Such characterizations simplify the task of determining when two processes are related, and they may be used as a basis for developing denotational models. In view of Theorem 3.5 and Lemma 3.7 we immediately have the following.

Theorem 5.1 *Let $\mathcal{P}$ and $\mathcal{P}'$ be processes. Then $\mathcal{P} \sqsubseteq_{may} \mathcal{P}'$ iff $L(\mathcal{P}) \subseteq L(\mathcal{P}')$*

The remainder of this section is devoted to a characterization of $\sqsubseteq_{must}$. We approach this in two stages; we first consider the preorder induced by *finite* tests, and we then generalize this to the scenario in which arbitrary tests are allowed. Note that, in contrast with the traditional framework, infinite tests do add expressive power.

5.1 Finite Tests

Finite transition systems have a bound on the longest execution sequence they can engage in. Formally, a transition system $\mathcal{P} = \langle P, A, \rightarrow, p_0\rangle$ is *finite* if there is a k such that for any $s \in A^*$ with $p_0 \overset{s}{\rightarrow}$, $|s| \leq k$. Let $\mathcal{F}$ be the class of finite tests, and define $\mathcal{P} \sqsubseteq^{\mathcal{F}}_{must} \mathcal{P}'$ to hold if for every $\mathcal{T} \in \mathcal{F}$, $\mathcal{P}$ *must* $\mathcal{T}$ implies $\mathcal{P}'$ *must* $\mathcal{T}$. Certainly, $\mathcal{P} \sqsubseteq_{must} \mathcal{P}'$ implies $\mathcal{P} \sqsubseteq^{\mathcal{F}}_{must} \mathcal{P}'$.

To give an alternative characterization of $\sqsubseteq^{\mathcal{F}}_{must}$, we first give the following lemma.

Lemma 5.2 *Let $\mathcal{P} \sqsubseteq^{\mathcal{F}}_{must} \mathcal{P}'$. Then $L(\mathcal{P}') \subseteq L(\mathcal{P})$.*

Note that this lemma stands in contrast with the analogous result for $\underset{\sim}{\sqsubset}_{must_{DH}}$, which states that if $\mathcal{P}\underset{\sim}{\sqsubset}_{must_{DH}}\mathcal{P}'$, $s \in L(\mathcal{P}')$, *and* the start state of $\mathcal{P}$ converges on s, then $s \in L(\mathcal{P}')$ [8]. Indeed, on the basis of this and Theorem 5.1 we have the following.

Corollary 5.3 *Let $\mathcal{P}\underset{\sim}{\sqsubset}_{must}\mathcal{P}'$. Then $\mathcal{P}'\underset{\sim}{\sqsubset}_{may}\mathcal{P}$.*

Using standard techniques from traditional testing, we now have the following.

Theorem 5.4 *Let $\mathcal{P} = \langle P, A, \rightarrow, p_0\rangle$, $\mathcal{P}' = \langle P', A', \rightarrow, p_0'\rangle$. Then $\mathcal{P}\underset{\sim}{\sqsubset}^{\mathcal{F}}_{must}\mathcal{P}'$ iff for each s and p_1' with $p_0' \stackrel{s}{\Rightarrow} p_1'$, there is a p_1 with $p_0 \stackrel{s}{\Rightarrow} p_1$ and $S(p_1) \subseteq S(p_1')$.*

Note the similarity between this characterization and the one for $\underset{\sim}{\sqsubset}_{must_{DH}}$ given in Theorem 3.5; the difference lies in the insensitivity of $\underset{\sim}{\sqsubset}^{\mathcal{F}}_{must}$ to divergence.

5.2 Arbitrary Tests

In traditional testing theory, the preorders induced by finite tests coincide exactly with those resulting from arbitrary tests; infinite tests do not add any distinguishing power. The next example shows that this fails to be the case in our framework.

Example 5.1 *Let $\mathcal{P}$, $\mathcal{P}'$ and $\mathcal{T}$ be the processes and tests in Figure 2(i), (iii) and (ii), respectively. Then $\mathcal{P}\underset{\sim}{\sqsubset}^{\mathcal{F}}_{must}\mathcal{P}'$ but $\mathcal{P} \not\underset{\sim}{\sqsubset}_{must}\mathcal{P}'$, since $\mathcal{P}$ must $\mathcal{T}$ but $\mathcal{P}'$ $\not\text{must}$ $\mathcal{T}$.*

Although initially surprising, this fact does have an intuitive basis. In traditional testing, computations are deemed "good" (i.e. strongly successful) if, after a finite amount of time, one observes a "good" configuration. In our setting, however, computations may be successful even though no such "good" configuration is ever seen; in this case, the configurations in the computation must exhibit an ongoing capability to "become" good. Thus the infinitary behavior of a process becomes important, and as we use tests to probe a process's behavior, it follows that infinite tests add power.

We now present a characterization of $\underset{\sim}{\sqsubset}_{must}$. We begin with some definitions.

Definition 5.5 *Let $\mathcal{P} = \langle P, A, \rightarrow, p_0\rangle$ be a process, with $p, p' \in P$, and let $s \in (A - \{\tau\})^*$. If S is a set then define S^ω to be the set of infinite sequences of elements of S.*

1. *$L_\omega(p) = \{\, \sigma \in (A - \{\tau\})^\omega \mid p \stackrel{\sigma}{\Rightarrow} \}$, where $p \stackrel{\sigma}{\Rightarrow}$ is defined in the obvious fashion.*
2. *$\Sigma(p) = \{\, \sigma \in L_\omega(p) \mid (s \text{ is a prefix of } \sigma \wedge p \stackrel{s}{\Rightarrow} p') \Rightarrow \exists a \in (A - \{\tau\}) :\ p' \stackrel{a}{\Rightarrow} \}$.*
3. *If $X \subseteq A^\omega \cup A^*$ then $s \prec X$ if $st \in X$ for some t.*
4. *$\mathcal{S}(p, s) = \bigcup\{\, S(q) \mid p \stackrel{s}{\Rightarrow} q \}$.*

So $L_\omega(p)$ contains the infinite traces p may perform, while $\Sigma(p)$ consists of elements of $L_\omega(p)$ with the property that while p is "executing" any one of them it cannot reach a state in which all visible actions are disabled. Note that $s \prec X$ holds if s is a prefix of *some* element of X. We now introduce a relation on processes that we will show coincides with $\underset{\sim}{\sqsubset}_{must}$.

Definition 5.6 *Let $\mathcal{P} = \langle P, A, \rightarrow, p_0\rangle$ and $\mathcal{P}' = \langle P', A', \rightarrow, p_0'\rangle$. Then $\mathcal{P} <<_{must} \mathcal{P}'$ if for every $s \in (A' - \{\tau\})^*$, p_1' such that $p_0' \stackrel{s}{\Rightarrow} p_1'$, and $X \subseteq \Sigma(p_1') \cup \{\epsilon\}$ then:*

$$\exists s' \prec X, p_1 :\ p_0 \stackrel{ss'}{\Rightarrow} p_1 \wedge (\forall t :\ s't \prec X \Rightarrow \mathcal{S}(p_1, t) \subseteq \mathcal{S}(p_1', s't)).$$

The definition is technically complex, but underlying intuition is not too difficult. Suppose $p_0' \stackrel{s}{\Rightarrow} p_1'$ and $\Sigma(p_1') = \{\sigma\}$. If $\mathcal{P} <<_{must} \mathcal{P}'$ then (assuming $s' = \epsilon$) this condition assures the existence of some s-derivative p_1 of p_0 such that the action capabilities of p_1 after any prefix t of σ is contained in those of p_1' after t. A consequence is that if $\mathcal{P}'$ fails a test $\mathcal{T}$ because of the derivation $p_0' \stackrel{s}{\Rightarrow} p_1' \stackrel{\sigma}{\Rightarrow}$, then surely $\mathcal{P}$ will also fail $\mathcal{T}$: the action capabilities of p_1 along σ are no larger than those of p_1' along σ, and thus if p_1' could not trigger a successful computation on $\mathcal{T}$ then neither could p_1. We now have the following; the proof may be found in the appendix.

Theorem 5.7 *Let $\mathcal{P}, \mathcal{P}'$ be processes. Then $\mathcal{P} \sqsubset\!\!\!\sim_{must} \mathcal{P}'$ iff $\mathcal{P} <<_{must} \mathcal{P}'$.*

A Sufficient Condition. Determining whether or not $\mathcal{P} <<_{must} \mathcal{P}'$ can be difficult. We now give a sufficient condition that it easier to check.

Proposition 5.8 *Let $\mathcal{P} = \langle P, A, \rightarrow, p_0 \rangle$ and $\mathcal{P}' = \langle P', A', \rightarrow, p_0' \rangle$. Then $\mathcal{P} \sqsubset\!\!\!\sim_{must} \mathcal{P}'$ if for every s and p_1' with $p_0' \stackrel{s}{\Rightarrow} p_1'$, there is a p_1 with $p_0 \stackrel{s}{\Rightarrow} p_1$ and $L(p_1) \subseteq L(p_1')$.*

The similarity between this condition and Theorem 5.4 is worth noting. From this proposition it also follows that observational equivalence [13] is finer than $\sqsubset\!\!\!\sim_{must}$.

5.3 The ABP Revisited

We now return to the sample implementations of the Alternating Bit Protocol in Section 2 in order to examine the relationship between ABP and ABP'. Using Proposition 5.8 it is not difficult to see that $ABP' \sqsubset\!\!\!\sim_{must} ABP$. On the other hand, $ABP \not\sqsubset\!\!\!\sim_{must} ABP'$; to see this, consider the test $\mathcal{T}$ given in Figure 2(iv). It is straightforward to show that ABP' $m\!\!\!/ust$ $\mathcal{T}$, since after executing a *send* this process may deadlock. However, it is the case that ABP *must* $\mathcal{T}$.

6 Conclusions

In this paper we have developed an alternative theory of process testing that incorporates a notion of "fairness" into the determination of when a process passes a test. In contrast with traditional testing theory, divergence no longer becomes catastrophic; rather, a process that is superficially capable of an infinite internal computation but is also always capable of interacting with its environment will eventually do so. We achieved this by redefining when a process must inevitably pass a test—in our setting, this holds if "almost all" of their computations are successful in a sense we made precise using simple notions from mathematical analysis. We also provided alternative characterizations of the preorders that obviate the need for reasoning about process' response to all tests and demonstrated the utility of the preorders on an example involving the Alternating Bit Protocol. In particular, we showed that two versions of the protocol that have intuitively different behavior but are equated under traditional testing theory are distinguished in our framework.

As future work, we would like to investigate denotational and algebraic issues involved in using our framework to give meaning to process algebras such as CCS. Our initial results indicate that our preorders are substitutive for typical constructs found in these languages; however, traditional fixpoint approaches to defining the meaning of recursive processes would no longer apply because of the infinitary nature of the "observables" required by our preorders. This problem also occurs in other semantic treatments of fairness, and we expect that results obtained there may be applicable. In addition, we would like to see if our alternative characterizations can be simplified, and we hope to develop algorithms for computing when two finite-state systems are related by our preorders.

Related Work. Fairness has been an active area of research for over a decade; a good overview of the subject may be found in [9]. A variety of different notions of fairness have been defined, and our definition of potentially successful may be seen as related to two of them in particular. First, define success to be *enabled* in a configuration c if there is a successful configuration reachable from c. A computation is then potentially successful if success is continuously enabled. This amounts to an assumption of *weak* fairness [9] with respect to success in that the definition of potential success implicitly assumes that if success is continuously enabled, then it is eventually "reached". For finite-state processes and tests connections may also be drawn with the notion of ∞-fairness introduced in [3]. According to this notion an execution is unfair if a transition which is enabled infinitely often by a sequence of unrestricted length occurs only finitely often. In our case, a weakly successful computation is in a certain sense ∞-unfair with respect to success transitions.

Other researchers have also given semantic models of fairness in process-algebraic settings. Following [9], in this work one may distinguish between *structural* fairness, in which notions of fairness are incorporated into the semantics of process operators [6, 10], and *observational* fairness, in one focuses on the "allowable" execution sequences [2]. Our work falls in the latter category; in what follows we briefly compare our results with those of [10] and [2], which also develop testing/failures frameworks. In [10] Hennessy gives an algebraic theory for reasoning about asynchronous communicating processes built using a fair parallel composition operator, $\|$. Divergence is still considered to be catastrophic in his model; consequently, his semantics would equate the systems ABP and ABP' considered in our example, even when the the fair parallel combination operator is used for composing the protocol's entities. On the other hand, he does offer an algebraic theory and a fully abstract denotational model. The issue of *fair abstraction* of divergent processes is addressed in [2]. Like our theory, their approach differentiates between ABP and ABP'. However, it is possible to come up with another implementation ABP'' of the Alternating Bit Protocol using a different implementation M'' of the medium entity that could induce "divergent deadlocking" states (states which are divergent and incapable of ever performing visible actions); their theory would equate ABP and ABP'' while ours would not. There are also situations in which our semantics licenses the treatment of divergence as finite delay whereas theirs does not. Like [10], their work presents proof systems that permit the verification of processes using syntactic manipulations. Koomen's Fair Abstraction Rule [12] is not valid in their semantics, although a restricted version is; neither rule holds in our framework.

References

[1] J.C.M. Baeten, editor. *Applications of Process Algebra*, vol. 17 of *Cambridge Tracts in Theoretical Computer Science*. Cambridge Press, Cambridge, 1990.

[2] J.A. Bergstra, J.W. Klop, and E.-R. Olderog. Failures without chaos: A new process semantics for fair abstraction. In M. Wirsing, editor, *Formal Description Techniques—III*, pages 77–103. North-Holland, 1987.

[3] E. Best. Fairness and conspiracies. *Inf. Proc. Letters*, 18:215–220, 1984.

[4] S.D. Brookes, C.A.R. Hoare, and A.W. Roscoe. A theory of communicating sequential processes. *Journal of the ACM*, 31(3):560–599, July 1984.

[5] R. Cleaveland, J. Parrow, and B. Steffen. The Concurrency Workbench: A semantics-based tool for the verification of finite-state systems. *ACM Trans. on Prog. Lang. and Sys.*, 15(1):36–72, January 1993.

[6] G. Costa and C. Stirling. Weak and strong fairness in CCS. *Inf. and Comp.*, 87:207–244, 1987.

[7] R. De Nicola. Extensional equivalences for transition systems. *Acta Inf.*, 1987.

[8] R. De Nicola and M.C.B. Hennessy. Testing equivalences for processes. *Theoret. Comp. Sci.*, 34:83–133, 1983.

[9] N. Francez. *Fairness.* Springer-Verlag, 1986.

[10] M.C.B. Hennessy. An algebraic theory of fair asynchronous communicating processes. *Theoret. Comp. Sci.*, 49:121–143, 1987.

[11] M.C.B. Hennessy. *Algebraic Theory of Processes.* MIT Press, Boston, 1988.

[12] C.J. Koomen. Algebraic specification and verification of communications protocols. *Science of Comp. Prog.*, 5:1–36, 1985.

[13] R. Milner. *Communication and Concurrency.* Prentice-Hall, London, 1989.

A Appendix

We begin by stating a useful lemma.

Lemma A.1 *Let $\mathcal{P}, \mathcal{P}'$ be such that $\mathcal{P} <<_{must} \mathcal{P}'$. Then $\mathcal{P} \sqsubseteq^{\mathcal{F}}_{must} \mathcal{P}'$.*

The next two lemmas together prove Theorem 5.7. In what follows, if $p_0||e_0 \xrightarrow{a_1} \cdots \xrightarrow{a_k} p_k||e_k$ is a prefix of some computation of $p_0||e_0$ then we will refer to the sequence obtained by deleting the τ's from the string $a_1 \cdots a_k$ as its *visible content.* Note that if s is the visible content of a prefix of some computation of $p||e$ then $s \in L(p) \cap L(e)$.

Lemma A.2 *Let $\mathcal{P}$ and $\mathcal{P}'$ be processes. Then $\mathcal{P} <<_{must} \mathcal{P}'$ implies $\mathcal{P} \sqsubseteq_{must} \mathcal{P}'$.*

Proof. Let $\mathcal{P} = \langle P, A, \rightarrow, p_0 \rangle$ and $\mathcal{P}' = \langle P', A', \rightarrow, p_0' \rangle$, and let $\mathcal{E}$ be some test $\langle E, A'' \cup \{w\}, \rightarrow, e_0 \rangle$ such that $\mathcal{P}$ *must* $\mathcal{E}$. We must prove that $\mathcal{P}'$ *must* $\mathcal{E}$. This can be accomplished by showing any configuration $c = p_1'||e_1$ in an arbitrary computation of $p_0'||e_0$ is potentially successful. Let s be the visible content of this computation from $p_0'||e_0$ to c; then $p_0' \stackrel{s}{\Rightarrow} p_1'$ and $e_0 \stackrel{s}{\Rightarrow} e_1$.

The proof now splits into two cases. For the first, suppose there is some configuration $c' = p_2'||e_2$ such that $c \longmapsto^* c'$ and $S(p_2') \cap S(e_2) = \emptyset$. The fact that c is potentially successful follows from Lemma A.1 and Theorem 5.4. For the second case suppose that every $c' = p_2'||e_2$ with $c \longmapsto^* c'$ is such that $S(p_2') \cap S(e_2) \neq \emptyset$. This means the visible content of *every* computation of c will be an element of $\Sigma(p_1')$. Let X denote this induced nonempty subset of $\Sigma(p_1')$. From the definition of $<<_{must}$, we can infer the existence of s' and p_1 such that $s' \prec X$ and $p_0 \stackrel{ss'}{\Rightarrow} p_1$. Further, if $s't \prec X$ then $\mathcal{S}(p_1, t) \subseteq \mathcal{S}(p_1', s't)$. Since $s' \prec X$ $s' \in L(p_1') \cap L(e_1)$. As $e_0 \stackrel{s}{\Rightarrow} e_1$, it is clear that there is some e_2 such that $e_0 \stackrel{ss'}{\Rightarrow} e_2$. So a prefix of a computation of $\mathcal{P}||\mathcal{E}$ with the following structure can be constructed: $p_0||e_0 \longmapsto^* p_1||e_2$ and the visible content of this part of the computation is ss'. As $\mathcal{P}$ *must* $\mathcal{E}$, $p_1||e_2$ must be potentially successful.

Now we claim that if $t \in L(p_1) \cap L(e_2)$ then $s't \in L(p_1') \cap L(e_1)$. The proof of this claim will be done by induction on the length of the string t. The basis case is $t = \epsilon$. Then $s' \in L(p_1') \cap L(e_1)$, since s' is the visible content of some prefix of a computation of $p_1'||e_1$. For the induction step assume $t = t'a$ for some visible action a and $t'a \in L(p_1) \cap L(e_2)$ and $s't' \in L(p_1') \cap L(e_1)$. Then $s't'$ is the visible content of some prefix of a computation of $p_1'||e_1$ and so by definition of X, $s't' \prec X$. Hence $\mathcal{S}(p_1, t')$ is a subset of $\mathcal{S}(p_1', s't')$. Because $t'a$ is an element of $L(p_1)$, a belongs to $\mathcal{S}(p_1, t')$ as well as $\mathcal{S}(p_1', s't')$. Consequently $s't'a$ belongs to the language of p_1'. Moreover $e_1 \stackrel{s'}{\Rightarrow} e_2$ and $t'a \in L(e_2)$ implies $s't'a$ is also an element of $L(e_1)$. As $s't'a$ belongs to the languages of both p_1' and e_1, the claim follows immediately.

Since $p_1||e_2$ is potentially successful there exists some $p_2||e_3$ such that $p_1||e_2 \longmapsto^* p_2||e_3$ and $e_3 \stackrel{w}{\rightarrow}$. Let t be the visible content of this part of the computation. Clearly $t \in L(p_1) \cap L(e_2)$, and because of the claim proved in the previous paragraph it is also the case that $s't \in L(p_1') \cap L(e_1)$. So there exists process p_3' such that $p_1' \stackrel{s't}{\Rightarrow} p_3'$. Since we know $e_1 \stackrel{s't}{\Rightarrow} e_3$ and $e_3 \stackrel{w}{\rightarrow}$, it follows that the configuration $c = p_1'||e_1$ is potentially successful.

$\square$

Lemma A.3 *Let $\mathcal{P}$ and $\mathcal{P}'$ be processes. Then $\mathcal{P} \sqsubseteq_{must} \mathcal{P}'$ implies $\mathcal{P} <<_{must} \mathcal{P}'$.*

Proof. Let $\mathcal{P} = \langle P, A, \rightarrow, p_0 \rangle$ and $\mathcal{P}' = \langle P', A', \rightarrow, p_0' \rangle$. Fix p_1' such that $p_0' \stackrel{s}{\Rightarrow} p_1'$ for some s in $(A' - \{\tau\})^*$, and let X be some subset of $\Sigma(p_1') \cup \{\epsilon\}$. We will construct a test $\mathcal{E}$ so that $\mathcal{P}'$ fails it. Since $\mathcal{P}'$ and $\mathcal{P}$ are related by $\sqsubseteq_{must}$, $\mathcal{P}$ must also fail $\mathcal{E}$. The test $\mathcal{E}$ will be constructed in such way that when $\mathcal{P}$ fails $\mathcal{E}$, it follows that $\mathcal{P}$ possesses the property given in Definition 5.6.

To define $\mathcal{E} = \langle E, A' \cup A, \rightarrow, e_0 \rangle$ let $E_1 = \{\ [t] \mid t \textit{ is a proper prefix of } s \}$ and $E_2 = \{\ [st] \mid t \prec X \}$ and $E_3 = \{[Succ]\}$ where $Succ$ does not belong to $(A \cup A')^*$. Take $E = E_1 \cup E_2 \cup E_3$. $\rightarrow$ is the smallest subset of $E \times (A \cup A') \times E$ satisfying the following conditions: (1) $[t] \stackrel{a}{\rightarrow} [ta]$; (2) $e \stackrel{\tau}{\rightarrow} [Succ]$ if $e \in E_1$; (3) $[Succ] \stackrel{w}{\rightarrow} [Succ]$; and (4) $[st] \stackrel{a}{\rightarrow} [Succ]$ if $[st] \in E_2$ and a is some visible action belonging to $(A \cup A') - \mathcal{S}(p_1', t)$. Also take $e_0 = [\epsilon]$. Note that if $[s'] \in E_1 \cup E_2$ and $e_0 \stackrel{t}{\Rightarrow} [s']$ then $t = s'$. Note also that $\{\ t \in (A' - \{\tau\})^* \mid [s] \stackrel{t}{\Rightarrow} [Succ] \} \cap L(p_1') = \emptyset$. Thus the configuration $p_1'|||[s]$ is not potentially successful. As $p_0' \stackrel{s}{\Rightarrow} p_1'$ and $e_0 \stackrel{s}{\Rightarrow} [s]$, it is possible to construct a computation such that $p_0'||e_0 \longmapsto^* p_1'|||[s]$. Consequently $\mathcal{P}'$ fails $\mathcal{E}$.

Since $\mathcal{P} \sqsubseteq_{must} \mathcal{P}'$, it must be the case that $\mathcal{P}$ fails $\mathcal{E}$ as well, i.e. there is a $c = p_1||e_1$ such that $p_0||e_0 \longmapsto^* c$ and c is not potentially successful. $e_1 \notin E_1$ since all the elements of E_1 have a τ-transition to $[Succ]$. So $e_1 \in E_2$ and hence has the form $[ss']$ for some $s' \prec X$. Since $e_0 \stackrel{t}{\Rightarrow} [ss']$ implies $t = ss'$, there exists p_1 such that $p \stackrel{ss'}{\Rightarrow} p_1$ and $p_1||e_1$ is *not* potentially successful. Now we must prove that for all t such that $s't \prec X, \mathcal{S}(p_1, t) \subseteq \mathcal{S}(p_1', s't)$. Assume instead the existence of t_0 such that $s't_0 \prec X$ but $\mathcal{S}(p_1, t_0)$ is not contained in $\mathcal{S}(p_1', s't_0)$. Let b be a element of $\mathcal{S}(p_1, t_0) - \mathcal{S}(p_1', s't_0)$. Obviously, $p_1 \stackrel{t_0 b}{\Rightarrow}$. From the construction of $\mathcal{E}$, $s't_0 \prec X$ implies $ss't_0$ is an element of E_2. Moreover $b \in (A \cup A') - \mathcal{S}(p_1', s't_0)$ implies $[ss't_0] \stackrel{b}{\rightarrow} [Succ]$ whence $[ss'] \stackrel{t_0 b}{\Rightarrow} [Succ]$. As $p_1 \stackrel{t_0 b}{\Rightarrow}$ and $e_1 \stackrel{t_0 b}{\Rightarrow} [Succ]$, we can conclude $p_1||e_1$ is potentially successful which is the desired contradiction.

$\square$