axioms, then we would also be able to apply these axioms to a differently designed example where the missing information ought not to be derived (because it is not valid) and we would have a serious problem.

Before concluding we should reiterate that in spite of all the debate, the differences between CT and PT-based systems do not have any real effect when dealing with most processes. Most processes (like the buffer processes in section 2) can be adequately dealt with using first order invariants on channel traces in a CT-system, or on projections of process traces on the individual channels in the PT-system. The specificiations using channel traces or such projections look essentially identical to each other and combining such invariants of component processes to obtain the invariant of the parallel composition is also an identical task in both systems. When processes become ugly, so to speak, the CT-based system will have to use temporal specifications, and the PT-based systems will have to use full process traces, not projections.

References

[BA] J.Brock, W.B.Ackerman, Scenarios: a model of non-deterministic computations, LNCS 107, 1981, pp. 252-259.

[JK] B.Jonsson, J.N.Kok, Comparing two fully abstract dataflow models, PARLE 1989, LNCS, pp. 217-234.

[NDGO] V.Nguyen, A.Demers, D.Gries, S.Owicki, A model and temporal proof system for networks of processes, Distributed Computing, 1, 1986.

[S] N.Soundararajan, Ax. semantics of CSP, ACM TOPLAS, 6, 1984.

[W] J.Widom, Trace based network proof systems, Ph.D. thesis, Cornell University, 1987.

[WGS] J.Widom, D.Gries, F.Schneider, Completeness and incompleteness of network proof systems, 14th Symp. on Principles of Prog. Languages, 1987, pp. 27-38.

[Z] J.Zwiers, Compositionality, concurrency, and partial correctness, LNCS 321, 1987.

by a single communication, but this sequence cannot arise during the execution of P_3.

In their proof that from precise invariants of component processes one can derive precise invariants of the parallel composition it is this stronger definition of preciseness that [WGS] use, not the weaker one as we implied earlier. But if we assume that we must start with specifications of component processes which are (both) precise, we have a serious problem. There is no way to specify, in first order logic, such invariants for P_3 and P_4 (as we saw r_3, r_4 are the strongest such invariants these processes satisfy). We would have to instead move to temporal assertions which means that much of the promise that the [WGS] approach held, that of retaining the simplicity of the first order CT-based system, will not be met. Indeed with such invariants, it is not clear that the axioms of [WGS] would seem to be needed only to deal with the type of incompleteness that Zwiers [Z] calls *adaptation incompleteness* rather than the kind of problem illustrated by the BA example or our examples.

Before concluding this section we briefly consider the models that [JK] prove are 'fully abstract'. As we said earlier, a model is fully abstract if it contains all (but no more) information necessary to be able to distinguish between processes that exhibit differences in their communications with the outside world, which differences can then be converted to differences that can be expressed in terms of channels by parallel composing, if necessary, these processes with other appropriately designed processes. [JK] establish the full abstractness of three models. The first is the process trace model. The remaining two models are the ones of interest to us. Both of these use channel traces but the semantics of a process is not just the set of values of its channel traces. Instead the semantics of a process is a set of functions whose values are essentially all the legal sequences (i.e. sequences that can arise during execution of the process) of states, where by a state we mean, as in the last two paras, the set of values of all the channel traces). But given such a legal sequence, we can directly map it to the corresponding process trace and this is exactly why these models are fully abstract. (The main difference between the two models that [JK] consider are in the details of how the functions are specfied.) And indeed this is also the reason why the CT-based system, or rather and more precisely, the system in which processes are characterized by sequences of possible states is complete. The problem, of course, is that the simplicity which was the original advantage of the CT-based system over the PT-based system is lost since specifying these legal sequences, or when combining the sets of legal sequences of component processes to obtain the set of legal sequences of the parallel composition, is no easier than the corresponding task in the PT-based system (may be even harder depending on how much temporal logic is used).

4. Dead End?

Is it likely that stronger axioms than those [WGS] can be found using which we can achieve completeness in a CT-based system while using only first order assertions for invariants of processes? The answer, as indicated by our $[P_3 \parallel P_4]$ and $[P_3' \parallel P_4']$, is almost certainly no. The problem is that even a 'small amount of missing information' about the ordering between communications on separate channels can be converted by composing the given process with an appropriately cleverly designed process to manifest itself as influencing the actual values that are output on specific channels. In fact, [JK] (in an appendix to their paper) show how to construct such a process quite simply. If we could recover this information using general

communications on c_3, c_4. We can see this by noting that the process

$P_3' :: [c_3!0 \rightarrow c_4!0 [] c_4!0 \rightarrow c_3!0];\ [e!0 \rightarrow \textbf{skip} [] e!1 \rightarrow \textbf{skip}\ ]$

also satisfies r_3 and for this process there is no relation between the value output on e and the order of communications on c_3, c_4. A similar process P_4' satisfies r_4 but there is no relation between the value output on f and the order or communications on c_3, c_4. Given these facts it is clear that the [WGS] axioms cannot be used to strengthen $r_3 \wedge r_4$ to allow us to establish the equality of the values output on e and f by $[P_3 \parallel P_4]$ since this equality does not hold for $[P_3' \parallel P_4']$.

The only other possibility would be to try to arrive at stronger invariants for P_3 and P_4 (P_3' and P_4' will presumably not satisfy these stronger invariants). But if we are using a CT-based approach, no stronger invariant than r_3 (r_4) can be valid for P_3 (P_4) since every one of the combinations of values for the traces c_3, c_4, and e (f) allowed by r_3 (r_4) can actually arise during execution. So no strengthening of the invariants of P_3, P_4 is possible. If we cannot strengthen r_3, r_4 and we should not even try to strengthen $r_3 \wedge r_4$ (given the existence of P_3', P_4'), the conclusion has to be that there is no way to establish the relation between the values output on channels e and f by $[P_3 \parallel P_4]$ in a CT-based system.

[WGS] introduce the notion of 'precise invariants' in their paper. Informally, an invariant is precise if every combination of (channel) trace values that satisfies the invariant either can actually arise during the execution of the process or is such that 'somewhere along the way to reaching this set of values' the traces would have to go through one or more combinations of values that do not satisfy the invariant. (The idea being that in that case we can use their [WGS] axioms to strengthen the invariant to be satisfiable only by those combinations of trace values that can actually arise during execution.) They then proceed to show that given precise invariants of component processes, we can derive a precise invariant for the parallel composed process. This seems to be contradicted by our example. r_3, r_4 are precise invariants for P_3, P_4 since each combination of trace values allowed by r_3, r_4 can arise in practice, yet $r_3 \wedge r_4$ is not precise for the process $[P_3 \parallel P_4]$ as we saw.

The problem is that though the examples that they consider suggest the above definition of preciseness, [WGS]'s formal definition of preciseness is stronger. To see this definition let us introduce the notion of a 'state' of a process as the set of values of all the channel traces of the process at any given time. Consider now a sequence of states such that each state can arise from the previous one by performing one communication on one channel, in other words there are no violations of the [WGS] axioms in going from state to the next in the sequence. Suppose also that each state in the sequence satisfies the invariant r in question. Then we will say that r is 'path precise' (this is not a term that [WGS] use; we are using this term to distinguish it from the notion introduced in the last para) if this sequence of states can actually arise during the execution of the process. Note that it is not enough that each *state* can arise; the entire *sequence* must be capable of arising during execution. It is clear that path preciseness implies preciseness but the converse is not true. In particular r_3, r_4 are *not* path precise for P_3, P_4 since r_3 for instance is satisfied by each state in the following sequence:

$\langle\ (c_3 = \langle\rangle, c_4 = \langle\rangle, e = \langle\rangle), (c_3 = \langle 0\rangle, c_4 = \langle\rangle, e = \langle\rangle), (c_3 = \langle 0\rangle, c_4 = \langle 0\rangle, e = \langle\rangle), (c_3 = \langle 0\rangle, c_4 = \langle 0\rangle, e = \langle 0\rangle)\ \rangle$, and each state in this sequence can be obtained from the previous one

Now the problem cannot be ignored any longer since we can, in fact, write a stronger invariant for $[P_3 \parallel P_4]$ to express the equality of values output on e, f:

$$R_{34} : [\,[(e \preceq \langle 0 \rangle) \wedge (f \preceq \langle 0 \rangle)] \vee [(e \preceq \langle 1 \rangle) \wedge (f \preceq \langle 1 \rangle)]\,]$$

but there is no way to *derive* this from $r_3 \wedge r_4$ and as we will see in the next section this doesn't change even with the [WGS] axioms.

4. Hot On The Trail (of the missing information)

Consider the first example $[P_1 \parallel P_2]$ from section 3. The conjunction of r_1, r_2, the invariants of P_1, P_2 is:

$$r_1 \wedge r_2 : [(c_1 = \langle\rangle \wedge c_2 = \langle\rangle \wedge d = \langle\rangle) \vee (c_1 = \langle 0 \rangle \wedge c_2 = \langle 0 \rangle \wedge d = \langle 0 \rangle)\,]$$

This assertion does, using the special kind of reasoning from section 2, let us conclude that no value will be output on d since for that to happen a value must have also been output on c_1 and on c_2. But if we consider the situation when only one or two of these three values have been output, the corresponding values of the channel traces c_1, c_2, d will not satisfy $r_1 \wedge r_2$ and hence, since there is no way for the process to *simultaneously* communicate on all three chanels, no values will be output on any of the channels. In particular no value will be output on d.

But this type of arguement does not apply once we abstract away the internal channels c_1, c_2 and obtain the 'proper' invariant (i.e., the invariant that refers only to $[P_1 \parallel P_2]$'s external channels:

$$r_{12} : [(d = \langle\rangle) \vee (d = \langle 0 \rangle)]$$

To solve this problem, [WGS] introduce two new axioms. Their *prefix* axiom says that the value of a channel trace at any time will be a proper prefix of its value at the 'next' moment. Essentially this says that communications cannot be 'taken back' once they have occurred. Their second axiom, the *ordering axiom*, essentially says that if (an invariant asserts that) two or more communications must take place simultaneously (or not at all) then they will not take place at all. They also go on to show that there is no way to express these axioms without using temporal operators and Widom [W] analyses in depth exactly 'how much' of temporal notations are needed to express these axioms (the point being that if we simply allow all the power of temporal logic (as do [NDGO]), we would completely lose the simplicity of the CT-based approach and such a system would be even more complex than the PT-based system). For our purposes though, it is not necessary to look at the precise formal expressions of these axioms. All we need is to note that using these axioms we can formalise the type of informal arguments we have considered. This essentially takes care of our first example $[P_1 \parallel P_2]$ since we can use the ordering axiom to $r_1 \wedge r_2$ and derive the stronger invariant:

$$[(c_1 = \langle\rangle \wedge c_2\langle\rangle \wedge d = \langle\rangle)]$$

We can then omit the references to the internal channels and derive the required invariant $[d = \langle\rangle]$.

Consider now our second example. By looking at r_3 (and r_4) there is, as we saw, no way to tell what the relation between the value output on e (and f) and the the order of prior

corresponding channel trace).

$$r_1 : [(c_1 = \langle\rangle \wedge c_2 = \langle\rangle) \vee (c_1 = \langle 0\rangle \wedge c_2 = \langle\rangle) \vee (c_1 = \langle 0\rangle \wedge c_2 = \langle 0\rangle)\ ]$$

$$r_2 : [(c_1 = \langle\rangle \wedge c_2 = \langle\rangle \wedge d = \langle\rangle) \vee (c_1 = \langle\rangle \wedge \#c_2 = 1 \wedge d = \langle\rangle)$$
$$\vee (\#c_1 = 1 \wedge \#c_2 = 1 \wedge d = \langle\rangle) \vee (\#c_1 = 1 \wedge \#c_2 = 1 \wedge d = \langle 0\rangle)\ ]$$

Combining r_1, r_2, we get

$$r_1 \wedge r_2 : [(c_1 = \langle\rangle \wedge c_2 = \langle\rangle \wedge d = \langle\rangle) \vee (c_1 = \langle 0\rangle \wedge c_2 = \langle 0\rangle \wedge d = \langle 0\rangle)\ ]$$

Eliminating references to the internal channels c_1, c_2, we get, as the invariant for $[P_1 \parallel P_2]$:

$$r_{12} : [(d = \langle\rangle) \vee (d = \langle 0\rangle)]$$

From this there is no way (using the type of informal argument we saw in the last section) to see that $[P_1 \parallel P_2]$ will not output any values on d.

Our next example is more involved. It is the parallel composition of 2 processes $[P_3 \parallel P_4]$ where

$P_3 :: [c_3!0 \rightarrow c_4!0;\ e!0 [] c_4!0 \rightarrow c_3!0;\ e!1]$, and

$P_4 :: [c_3?x \rightarrow c_4?y;\ f!0 [] c_4?y \rightarrow c_3?x;\ f!1]$

The invariants r_3, r_4 for P_3, P_4 are:

$$r_3 : [(c_3 = \langle\rangle \wedge c_4 = \langle\rangle \wedge e = \langle\rangle) \vee (c_3 = \langle 0\rangle \wedge c_4 = \langle\rangle \wedge e = \langle\rangle)$$
$$\vee (c_3 = \langle 0\rangle \wedge c_4 = \langle 0\rangle \wedge e = \langle\rangle) \vee (c_3 = \langle 0\rangle \wedge c_4 = \langle 0\rangle \wedge e = \langle 0\rangle)$$
$$\vee (c_3 = \langle\rangle \wedge c_4 = \langle 0\rangle \wedge e = \langle\rangle) \vee (c_3 = \langle 0\rangle \wedge c_4 = \langle 0\rangle \wedge e = \langle 1\rangle)\ ]$$

The reader might already see the beginnings of a problem in r_3. There is no way to see the relation between the value output on e and the order of the previous communications on c_3, c_4. There is also no way to strengthen r_3 since each combination of values for c_3, c_4, e allowed by r_3 can actually arise during the execution of P_3. We may, for now, simply ignore this relation but when we consider $[P_3 \parallel P_4]$, we will see that without information about this relation (and a similar relation for P_4) we will be unable to establish a property of that process although the property is easily stated in terms of its channel traces.

$$r_4 : [(c_3 = \langle\rangle \wedge c_4 = \langle\rangle \wedge f = \langle\rangle) \vee (\#c_3 = 1 \wedge c_4 = \langle\rangle \wedge f = \langle\rangle)$$
$$\vee (\#c_3 = 1 \wedge \#c_4 = 1 \wedge f = \langle\rangle) \vee (\#c_3 = 1 \wedge \#c_4 = 1 \wedge f = \langle 0\rangle)$$
$$\vee (c_3 = \langle\rangle \wedge \#c_4 = 1 \wedge f = \langle\rangle) \vee (\#c_3 = 1 \wedge \#c_4 = 1 \wedge f = \langle 1\rangle)\ ]$$

Combining the two:

$$r_3 \wedge r_4 : [(c_3 = \langle\rangle \wedge c_4 = \langle\rangle \wedge e = \langle\rangle \wedge f = \langle\rangle) \vee (c_3 = \langle 0\rangle \wedge c_4 = \langle\rangle \wedge e = \langle\rangle \wedge f = \langle\rangle)$$
$$\vee (c_3 = \langle 0\rangle \wedge c_4 = \langle 0\rangle \wedge e = \langle\rangle \wedge f = \langle\rangle) \vee (c_3 = \langle 0\rangle \wedge c_4 = \langle 0\rangle \wedge e = \langle 0\rangle \wedge f = \langle\rangle)$$
$$\vee (c_3 = \langle 0\rangle \wedge c_4 = \langle 0\rangle \wedge e = \langle 1\rangle \wedge f = \langle\rangle) \vee (c_3 = \langle 0\rangle \wedge c_4 = \langle 0\rangle \wedge e = \langle\rangle \wedge f = \langle 0\rangle)$$
$$\vee (c_3 = \langle 0\rangle \wedge c_4 = \langle 0\rangle \wedge e = \langle\rangle \wedge f = \langle 1\rangle) \vee (c_3 = \langle 0\rangle \wedge c_4 = \langle 0\rangle \wedge e = \langle 0\rangle \wedge f = \langle 0\rangle)$$
$$\vee (c_3 = \langle 0\rangle \wedge c_4 = \langle 0\rangle \wedge e = \langle 1\rangle \wedge f = \langle 1\rangle) \vee (c_3 = \langle 0\rangle \wedge c_4 = \langle 0\rangle \wedge e = \langle 0\rangle \wedge f = \langle 1\rangle)$$
$$\vee (c_3 = \langle 0\rangle \wedge c_4 = \langle 0\rangle \wedge e = \langle 1\rangle \wedge f = \langle 0\rangle)\ ]$$

Simplifying and omitting references to the internal channels c_3, c_4, we get

$$r_{34} :\ [\ [(e \preceq \langle 0\rangle) \vee (e \preceq \langle 1\rangle)] \wedge [(f \preceq \langle 0\rangle) \vee (f \preceq \langle 1\rangle)]\ ]$$

would expect. Note that we have used the same name $h_{c'}$ for the trace of values output by B_1 on c' and the trace of values input by B_2 on c'. This is a fairly standard trick but, of course, it is ok only if i/o is synchronous (so that these traces are, in fact, always equal). If i/o were asynchronous, we would have to use distinct names, say, $h^{out}_{c'}, h^{in}_{c'}$ for these traces and add the condition $h^{in}_{c'} \preceq h^{in}_{c'}$ at parallel composition, i.e., when combining r_1, r_2 to obtain the invariant for $[B_1 \parallel B_2]$. For convenience we will assume synchronous communications in our discussion.

In a PT-based system, since the invariants r_1, r_2 would be expressed in terms of h_1, h_2, the process traces of B_1, B_2, things are more complex since to obtain the process trace h of $[B_1 \parallel B_2]$, we would have to 'merge' h_1, h_2, and then omit the elements corresponding to communications on the internal channel c'. (For details, see for example [S].) In an example like $[B_1 \parallel B_2]$, the situation in the PT-based system is actually not very different than in the CT-based system since r_1, r_2 can be expressed in terms of projections of h_1 on the channels c, c' and the projections of h_2 on c', d respectively. Once this is done, the invariants in the PT-based system look essentially like the invariants in the CT-based system and combining these to obtain the invariant for the parallel composition is also no harder than in the CT-based system since the 'merge' operartion in this case reduces to identifying, for instance, the projection of h_1 on c' with the projection of h_2 on c', the projection of the trace h of the parallel composition on the channel c with the projection of h_1 on c etc. It is only if the invariants of the individual processes are stated in terms of the respective process traces rather than their projections on the individual channels does the composition become complex. It is this complexity in the general case that is the main disadvantage of the PT-based system compared to the CT-based system.

Consider r_{12} again. The same type of reasoning we used earlier on r applies to r_{12} since r_{12} and r have essentially the same structure. It was examples of this kind that suggested that there was no need to complicate the formalism by using something like process traces. Instead this kind of reasoning applied (informally) whenever necessary would, it was hoped, always supply the information about the ordering of communications along separate channels that was not formally included in the invariants. The BA example as well as the simpler examples we consider in the next section show that this is not the case.

3. The Missing Information

We present two examples in this section. The first one is similar, as far as the problem it demonstrates is concerned, to the BA example. This problem, as we will see in the next section, is handled by the [WGS] axioms. Our second example will demonstrate a different type of problem with the CT-based systems.

The first example is the parallel composition of 2 processes $[P_1 \parallel P_2]$ where

$P_1 :: c_1!0; c_2!0$, and

$P_2 :: c_2?x; c_1?y; d!0$

The invariants r_1, r_2 for P_1, P_2 (in r_1, r_2 below as well as in the rest of the paper, we will use c rather than h_c to denote the trace of communications on the channel c; this should cause no confusion since the context will make it clear whether we are talking about the channel or the

on channel c and outputs them on d. $A[0..(n-1)]$ is the (internal) array in which B stores the values it has input but not yet output. in, out keep track of how many values have been input and output respectively:

$$in := 0; out := 0;$$
$$*[in < (out + n);\ c?A[in \bmod n] \rightarrow in := in + 1$$
$$\quad | \ out < in;\ d!A[out \bmod n] \rightarrow out := out + 1]$$

A CT-invariant r for B would be:

$$[h_d \preceq h_c \wedge \#h_c - n \leq \#h_d \leq \#h_c]$$

where $\preceq$ denotes 'prefix of', and $\#h$ is the number of elements in h. Thus r says that the sequence of values B has output on channel d is a prefix of the sequence of values it has input on c, with the length of the former being no more than n less than the length of the latter.

Note that this invariant does give us information on the order of communications along channels c and d. Consider, for instance, the value $\langle (d,0),\ (c,0),\ (c,1),\ (d,1),\ (c,2)\rangle$ for the process trace h of B. Although the correpsonding values for h_c and h_d ($\langle 0,1,2\rangle$ and $\langle 0,1\rangle$ respectively) do satisfy the invariant, we can still argue that this is not an acceptable value for h according to the given invariant r: Not only must r be satisfied at the *current* moment, but it must also have been satisfied at *all* moments in the *past* since r is an *invariant*. But this would not have been the case immediately following the communication recorded in the first element $(d,0)$ of the h in question since at this point the corresponding values of h_c, h_d would have been $\langle\rangle$ (the empty trace) and $\langle 0\rangle$ respectively and this pair of values would not have satisfied the invariant.

Using this type of reasoning, one can in general extract, from the CT-based invariant, considerable amount of information on the relative ordering of communications along different channels. Indeed this is precisely the type of intuition that Widom et al [WGS] formalise in their axioms. Of course, in the absence of these axioms, the argument in the last para is only informal; formally there is no way to show, given the r, that the the h we considered is disallowed without appealing to the [WGS] axioms.

Suppose we have two bounded buffers B_1, B_2; B_1 is exactly like B except it has a capacity n_1, inputs on channel c, and outputs on channel c'; B_2 is also like B except it has a capacity n_2, inputs on c', and outputs on d. The invariant for B_1 is then:

$$r_1 :: \ [h_{c'} \preceq h_c \wedge \#h_c - n_1 \leq \#h_{c'} \leq \#h_c]$$

and for B_2:

$$r_2 :: \ [h_d \preceq h_{c'} \wedge \#h_{c'} - n_2 \leq \#h_d \leq \#h_{c'}]$$

The invariant for $[B_1 \parallel B_2]$ can be obtained by just conjuncting r_1 and r_2 and eliminating references to communications on the internal channel c':

$$r_1 \wedge r_2 :: \ [h_d \preceq h_{c'} \preceq h_c \wedge \#h_c - n_1 \leq \#h_{c'} \leq \#h_c \wedge \#h_{c'} - n_2 \leq \#h_d \leq \#h_{c'}]$$

Simplifying and eliminating references to channel c', we get, as invariant for $[B_1 \parallel B_2]$:

$$r_{12} :: \ [h_d \preceq h_c \wedge \#h_c - (n_1 + n_2) \leq \#h_d \leq \#h_c]$$

This invariant shows that $[B_1 \parallel B_2]$ behaves like a buffer, copying values from c to d as we

2. Processes, Traces, and Specifications

We use a standard CSP-like notation with processes communicating along named channels. A process may either be a sequential process or the parallel composition $[P_1 \parallel \ldots \parallel P_n]$ of n other processes. A sequential process is made up of the standard statements: **skip**, assignment, sequential composition, selection, and repetition. Selection and repetition may have input as well as output guards.

Consider a parallel composed process $[P_1 \parallel \ldots \parallel P_n]$. Let I_i, O_i be the sets of input and output channels of P_i. We will make the standard assumption that $P_1, \ldots, P_n$ are 'compatible', i.e., that no channel appears in more than one I_i or in more than one O_i. (We have borrowed some of our terminology and notation from [JK].) If c appears in $I_i \cap O_j$ then it is a channel along which P_j can send values to P_i; such a channel is an *internal* channel of $[P_1 \parallel \ldots \parallel P_n]$, communications along which are, of course, not visible to an external observer of $[P_1 \parallel \ldots \parallel P_n]$. If c appears in O_j but not in $\cup I_i$, it is an *external* channel of $[P_1 \parallel \ldots \parallel P_n]$ and is used by $[P_1 \parallel \ldots \parallel P_n]$ for outputting values (to the 'environment' – more precisely to the process at the other end of c). Similar remarks may be made if $c \in I_i \wedge c \notin \cup O_j$.

Let P be a process (sequential or parallel composed). An external observer of P will see all the communications that P engages in on its external channel but none of its internal activites including communications on any internal channels (in the case of parallel composed processes). The observer can record the entire sequence of P's communications (with external agents) and this record h is the process trace of P. Thus h is a seqence of elementseach representing a communciation between P and an external process, an element of $h$being of the form (c, d) where c is the channel and d the value communicated. Communications in which P receive a value and those in which it sends a value are both recorded in the same manner. Alternately, we may associate a separate channel trace h_c with each channel c of P and record all communications that P engages in along *this* channel in h_c. Each element of h_c is just a value there being no need to record the identity of the channel. Given the value of the process trace h at any time, we can obtain the corresponding channel traces of P, say h_c in particular, by simply projecting out those elements of h whose first component is c and then omitting this first component from each element. On the other hand, given the values of all the channel traces of P at some time we cannot, in general, obtain the corresponding h since we do not know the relative order of communications along the different channels.

Consider next the specification of a process. One possibility is to specify the set of possible values that h, in the case of the PT-based approach, or the various sets of possible values that the channel traces of P in the case of the CT-based approach, may have when P finishes. A better alternative, especially for non-terminating processes, would be to specify an *invariant* that specifies the possible values that h or the set of values that the channel traces of P may have at *any* time during execution. We will use invariants not only because of their ability to deal with non-terminating processes, but also because, as we will soon see, they add considerable power to the CT-based system even for terminating processes. We will not formally define a language of assertions for the invariants. Something along the lines of the STL (Simple Trace Logic) of Widom [W] would be perfectly adequate.

Let us consider a simple example. Suppose B is a bounded buffer (size n) which reads values

a model to be fully abstract according to the definition of [JK], the model is required not only not to contain too much information (such as about the internals of the process) but also not contain too little information; it is this latter aspect that we are concerned with in this paper.

Widom, Gries, and Schneider [WGS] take a rather novel approach to solving the BA problem. Rather than adding features to the CT-model in order to be able to specify additional information about the communications of the specific process(es) under consideration, they introduce two new axioms which express very general and quite natural properties of channel traces. Rather surprisingly Widom et al are able to show that the BA problem disappears once these axioms are available. Clearly this approach is much better than trying to add features to the CT-model since essentially all the simplicity of that model is retained with the [WGS] approach.

The main purpose of the current paper is to show that unfortunately the [WGS] approach is not as satisfactory in general as one may think. We present a new example to demonstrate the essential incompleteness of channel traces. In particular our example shows that even with the axioms of [WGS], the CT-model cannot adequately express the behaviour of certain processes unless we are willing to use complex temporal invariants. More generally the example seems to show that general axioms such as those in [WGS] alone are unlikely to solve the problem since such axioms can only express properties that are true in general, whereas the crucial missing information in examples such as ours depends upon the details of the processes we consider, and are not general properties that hold universally. The only solution would then seem to be to move to a model such as the fws or restricted-fws models of [JK] or, in view of the fairly high degree of complexity of such models, to the process trace based model.

The paper is organized as folows. In the next section we briefly describe the fairly standard notation that we use for processes, traces, and specifications. In the third section we present two examples and in the following section discuss how the CT-based approach deals (or fails to) with each. The first example exhibits behaviour like that of the Brock-Ackerman example and we show how the CT-based approach can, when strengthened with the axioms of [WGS], deal with it. (The original BA example and even the simplified version in [NDGO] is quite involved.) Our second example is the one that is new to this paper and we show why general axioms such as those of [WGS] are inadequate to deal with this problem. We also briefly discuss the notion of 'preciseness' introduced in [WGS] as well as that of 'full abstractness' of [JK]. In our final section we reiterate our conclusion that the search for a way to improve the channel trace based models without resorting to temporal invariants that are not as easy to work with as first order invariants and without going all the way to a process trace based approach is not likely to succeed.

1. Introduction

One of the most popular approaches to specifying and verifying systems of communicating processes is the trace based approach. Essentially each process P of such a system is characterised by the sequence of communciations it engages in with other processes. Two types of traces have been used in the literature. The first, which we will call channel traces, records the communications that P goes through on *one* of its channels, a separate channel trace being used to record P's communications along each of its channels. The second type of trace, which we call process trace, records *all* the communications that P engages in (along *all* of its channels) on a single trace variable.

The primary advantage of channel traces is ease of use. In particular when composing in parallel a set of processes $P_1, \ldots, P_n$, the channel traces of $[P_1 \parallel \ldots \parallel P_n]$ are obtained directly from the channel traces of the component processes. Thus if c is a channel along which $[P_1 \parallel \ldots \parallel P_n]$ communicates with an external process, c will also be a channel of (exactly) one of $P_1, \ldots, P_n$, say, P_i; hence the trace of $[P_1 \parallel \ldots \parallel P_n]$'s communications along c is identical to the trace of P_i's communications along c.

The primary disadvantage of channel traces is the lack of information about the relative ordering of communications along different channels of a process. One could try to argue that this ordering is of no particular interest to an external observer of the process and so there is no need to include it. As it turns out, however, it is possible, using parallel composition, to construct a process P whose channel trace behaviour, for instance the specific values P may communicate along one of its channels, depends upon precisely this type of ordering information concerning the communications of P's component processes.

The first example of this kind was presented by Brock and Ackerman [BA] in their now classic paper. The Brock-Ackerman (BA) anomaly (as it has come to be known) lead to a flurry of attempts (for a partial list see [JK]) to solve the problem of the missing information. In each case the goal was to try to beef up the channel trace based (or CT-based) model sufficiently to solve the BA (and similar) problems without going all the way to a process trace based (PT-based) model. Jonsson and Kok [JK] analyse the issue carefully. They introduce the formal notion of '*full abstractness*' to represent whether or not a particular model contains enough information to avoid BA type of problems; and they proceed to show that a number of proposals that had been previously made are not satisfactory by showing that these models are not fully abstract. They also consider two models (their *fws* and *restricted fws* models) and show that these *are* fully abstract. We will briefly consider these two models later in the paper. One problem with these models is that they are no simpler than the PT-based model; indeed they provide essentially the same information as that model does but the information is coded in a number of functions rather than being given as a sequence of communications so it is not clear that these models are in any sense preferable to the PT-based models.. It is also worth noting here that the term 'fully abstract' is perhaps a bit confusing. The problem with the channel trace models and some of the others analysed by [JK] is not that they are not abstract enough but that they are *too* abstract; the problem is not that they contain inappropriate information (such as, for instance, the details of the internal workings of a process) but that they don't contain enough information about the external behaviour of the process. In order for

Tracing The Missing Information

Neelam Soundararajan
Computer and Information Science
The Ohio State University
Columbus, OH 43210, USA.

e-mail: neelam@cis.ohio-state.edu

Abstract: We consider two types of trace based systems, channel-trace systems and process-trace systems. Specifications and proofs in the former type of system are usually considered much simpler than in the latter. We show that if we want to ensure that no essential information is missing, we are forced to add considerable complexity to the channel trace systems. We conclude that specifications and proofs of any program in the process trace systems are no more complex than in the channel trace systems.