University of Hertfordshire
Faculty of Engineering
and Information Sciences

TECHNICAL REPORT

Department of Computer Science

MODELLING PROTEIN INTERACTION IN CCS

J Baillie

May 1999
Report number : 331

Modelling Protein Interaction in CCS

Jean Baillie

1 Introduction

This is an account of the progress made to date in attempting to model proteins and their interaction as CCS agents under parallel composition. Some earlier work was done using CSP but that model ran into problems over catalysis—basically, the way CSP forces multiway synchronization rendered the model useless for this purpose.

1.1 Rationale

The essential properties of interacting proteins were identified (by Hamid Bolouri) and modelled in CCS. These properties are

- the *type* of the protein, that is, the combination of amino acids which characterizes it (from the 20 or so amino acids there are thousands of proteins which may be formed)
- the concentration of each protein, that is, the number of molecules of that protein in the cell
- catalysis—certain proteins, usually enzymes, acts as catalysts in certain contexts.
- complementarity or affinity—the natural tendency for certain proteins to interact with certain others
- symmetry—the reaction between proteins A and B is the same as that between B and A
- 'orders of interactions', by which we mean that reactions may be 1-M, M-1 or M-M, referring to the number of proteins involved in a reaction, compared with the number resulting from that reaction. Note that 1-M implies that a single protein may 'spontaneously' cleave into many (smaller) proteins

The model described in this report obeys the expansion and restriction laws of CCS. Below is a summary of how each of the above characteristics is modelled in CCS.

- proteins are modelled as CCS agents, which are defined in terms of their behaviour—the behaviour of the agent defines the *type* of protein
- *concentration* of a given protein is modelled by a separate counter agent associated with each protein type
- a protein is 'used up' in a chemical reaction, so agents which have been composed have their counters decremented
- *catalysis* is modelled by processes which are not used up by the interaction, so could be defined recursively, with no counter, or else defined as ordinary proteins but without decrementing the counter
- interaction of proteins is modelled by concurrent composition of agents
- two or more proteins may be involved in an interaction (*order*), similarly for agents
- the result of CCS parallel composition is generally a choice of possible interleavings, only one of which will be the actual outcome—we need to model random or probabilistic selection of the resulting protein
- interaction is *symmetric*, since CCS composition is commutative
- *affinity* is implied by the composition laws of CCS
- *cleavage* cannot be done within the calculus (processes retain their integrity until they are composed with other processes), but can be managed outside it

In order to use concurrent composition in CCS as a model of interaction, we regard CCS as an 'engine' which performs the interaction. The input to this engine, and the output from it, are regarded as a kind of primordial soup out of which we may draw a (possibly random) selection of proteins for processing, and into which the resulting proteins return. Cleavage is achieved inside the soup, and not in the engine; the output from CCS may be returned to the soup intact, or we may decide to break it back up into its constituent parts—which may be those proteins whose interaction gave rise to it in the first place. This decision is taken in the wrapper (with apologies for multiple metaphors) program, which also chooses the proteins for interaction and the catalysts, if any, and keeps count of the numbers of molecules of each protein. But first, a quick introduction to CCS.

1.2 Brief summary of CCS syntax

- concurrent process are written with initial capitals and called 'agents'
- agents are modelled as sequence of 'actions'
- the set of actions admitted by an agent is called its 'sort'
- a . indicates sequence - say 'then'
- agents may be defined recursively, or not
- the distinguished agent '0' has no actions and models termination
- actions may be input or output, convention is that $\overline{overbar}$ denotes output
- complementary actions, eg, a and $\overline{a}$ communicate and 'resolve' into the distinguished action 'τ'
- τ is not observable but may cause nondeterminism in certain contexts
- communication may be *restricted*; if it is not then complementary actions may also communicate independently with their environment. All communication is restricted in this model.
- agents are composed, that is, allowed to run in parallel, modelling interaction. Composition allows agents to interleave and/or communicate, subject to sequencing of actions.

We may have to take liberties with certain aspects of CCS semantics.

2 Some examples

First, the notational convention we propose to use throughout:

- an agent/protein whose action sequence is 'a then b then c' is defined as $P_{abc} \stackrel{def}{=} a.b.c.0$;
- a catalyst, which is basically an agent that admits a single action a, say, is defined recursively (*pro tem*, see above) as $C_a \stackrel{def}{=} a.C_a$; this is not entirely satisfactory as it implies a qualitative difference from 'ordinary' proteins, which is not the case *in vivo*. Further work is needed here.
- a new binary interaction operator is introduced between agents P and Q, say, and is denoted by I and defined as follows:

$$P\, I\, Q \stackrel{def}{=} P \mid Q \setminus S$$

where

$$s \in S \iff (s \in P \wedge \overline{s} \in Q) \vee (\overline{s} \in P \wedge s \in Q)$$

Example 1

Interaction of proteins P_{ab} and P_c:

$$\begin{aligned} P_{ab}\ I\ P_c &\stackrel{def}{=} a.b.0 \mid c.0 \\ &= a.(b.0 \mid c.0) + c.(a.b.0 \mid 0) \\ &= a.b.c.0 + a.c.b.0 + c.a.b.0 \end{aligned}$$

Notes

1. The restriction set S is empty since the RHS of the condition simplifies to false.
2. There are three possible interleavings so we need some mechanism for choosing one—assigning probabilities to each outcome based on the length of the chain, for example, or maybe use a random number indicator[1].
3. If, for example, the selected outcome is $a.b.c.0$ then the counter for $a.b.c.0$ is incremented, while the counters for $a.b.0$ and $c.0$ are both decremented.

Example 2

Interaction of proteins P_{ab} and $P_{c\overline{a}}$:

$$\begin{aligned} P_{ab}\ I\ P_{c\overline{a}} &\stackrel{def}{=} a.b.0 \mid c.\overline{a}.0 \setminus a \\ &= c.(a.b.0 \mid \overline{a}.0 \setminus a \\ &= c.\tau.(b.0 \mid 0) \\ &= c.\tau.b.0 \\ &= c.b.0 \end{aligned}$$

Notes

1. The restriction set is $\{a\}$ (set notation is not necessary in the case of a singleton set)
2. Action c must occur first since a and $\overline{a}$ must synchronize
3. a and $\overline{a}$ resolve into τ
4. Non-leading τ's are absorbed, that is, they can be ignored since they model internal and so unobservable and inevitable state changes mid-sequence.

[1] We could use the CCS model-checking tool, the Concurrency Workbench (CWB) to check our models. The way that the CWB works in practice when offered a concurrent expansion is to give one of the possible outcomes—pragmatically, we could just take this as *the* one

Example 3

Interaction of proteins P_{ab} and $P_{\overline{a}c}$

$$\begin{aligned} P_{ab}\ I\ P_{\overline{a}c} &\stackrel{def}{=} (a.b.0 \mid \overline{a}.c.0) \setminus a \\ &= \tau.(b.0 \mid c.0) \\ &= \tau(b.0 + c.0) \end{aligned}$$

Notes

1. Within the calculus, leading τ's, although not observable, cannot be dropped. The result above is weakly equivalent to $b.0+c.0$, that is to say, as 'stand alone' agents their behaviour is indistinguishable. Full equality, however, requires that they be interchangeable in any context, and this is not the case in the context of choice. For any process R, say,

   $$R + \tau(b.0 + c.0) \neq R + (b.0 + c.0)$$

   The apparent choice presented on the left hand side may be unobservably pre-empted by the internal state transition τ, thus removing the possibility of R. This is not the case on the right hand side where, in fact, the brackets are redundant and there is a choice between three initial events (unless of course R itself contains a leading τ, which makes matters even worse).

2. This is where we might be seen by purists to be taking a liberty with CCS semantics. The choice in the result of this example will be regarded as modelling two possible resulting proteins, only one of which will materialise. Thus this agent as it stands will never appear in the context of choice; we will, by some mechanism yet to be decided, arbitrate between the two options and either $b.0$ will be selected or else $c.0$. An agent containing the choice operator cannot model a single protein; the choice between the alternatives must be made before the resulting (purely sequential) protein may interact with others. The result is therefore treated as if it were $b.0 + c.0$, disregarding the leading τ.

Example 4

Interaction of the proteins P_{ab} and $P_{\overline{a}\overline{b}}$.

$$\begin{aligned} P_{ab}\ I\ P_{\overline{a}\overline{b}} &\stackrel{def}{=} (a.b.0 \mid \overline{a}.\overline{b}.0)\backslash\{a, b\} \\ &= \tau.\tau.0 \\ &= 0 \end{aligned}$$

Notes

1. Leading τ's have been disregarded as described in note 2 of Example 3 above.
2. The result of this interaction is the anihilation of the original proteins.

Example 5

Interaction of the proteins P_{ab} and $P_{\overline{b}c\overline{a}}$. This interaction will deadlock since a and b are both restricted actions and so neither agent can proceed. A catalyst is needed, either $C_{\overline{a}}$ or C_b, to break the deadlock. Choose C_b.

$$\begin{aligned} P_{ab}\ I\ P_{\overline{b}c\overline{a}}\ I\ C_b &\stackrel{def}{=} (a.b.0 \mid \overline{b}.c.\overline{a}.0 \mid b.C_b)\backslash\{a,b\} \\ &= \tau.(a.b.0 \mid c.\overline{a}.0 \mid C_b)\backslash\{a,b\} \\ &= \tau.c.(a.b.0 \mid \overline{a}.0 \mid C_b)\backslash\{a,b\} \\ &= \tau.c.\tau.(b.0 \mid 0 \mid C_b)\backslash\{a,b\} \end{aligned}$$

Notes

1. Even after simplifying this using our (slightly sneaky) version of the τ laws of CCS, we still have deadlock; b is a restricted action and so cannot proceed. We need(ed) another catalyst, namely $C_{\overline{b}}$, to break this particular deadlock. (Note that the corresponding problem would have arisen had we made the initial choice of catalyst to be $C_{\overline{a}}$.)

2. This in turn gives rise to another question, namely the possible—indeed, probable—interaction of the catalysts themselves. We can 'get round' this by saying that this interaction only produces a potentially endless sequence of τ's, which have no effect on the nature of the protein that results from the intended interaction (flaky, I know). So now we revisit (without rewriting) the above expansion on the understanding that the second catalyst was there from the beginning but took no part in the reaction until its services were required, namely, at the point where we left it.

$$\begin{aligned} P_{ab}\ I\ P_{\overline{b}c\overline{a}}\ I\ C_b\ I\ C_{\overline{b}} &= \tau.c.\tau.(b.0 \mid 0 \mid C_b \mid \overline{b}.C_{\overline{b}})\backslash\{a,b\} \\ &= \tau.c.\tau.\tau.(0 \mid 0 \mid C_b \mid C_{\overline{b}}) \\ &= c.(C_b \mid C_{\overline{b}}) \end{aligned}$$

So the resulting protein is P_c, whose counter is then incremented. The two catalysts 'float off' back into the primordial soup (bit iffy).

3 Implementation

Example 5 raises the question of the 'choice' of catalysts: do we have a choice? When we come to implement this it may be that we have to make do with whichever catalysts are in the vicinity, and these might not be of any use at all, in which case the deadlock persists or, perhaps, is only partially broken. This may be a true reflection of the actual chemistry (or it may not).

Also, it won't have escaped the notice of anyone with more than a smattering of CCS that the definition of I in section 2 as a strictly *binary* operator means that Example 5 above won't actually work at all. The definition needs to be extended to allow muliple composition. This is possible but messy.

4 Conclusion

It would seem from this work that it is possible to model, at least in a rudimentary way, the interaction of proteins as the composition of CCS agents. To complete the model we would also need a wrapper program to generate the processes and catalysts for interaction, and for keeping track of the molecule count. Cleavage is not possible to model in CCS but would be possible inside the wrapper.

However, this model will never surprise us—at least, not in the way that a biological system might. We will never find a process produced having properties we could not have predicted: this itself is predictable. We cannot model anything other than the primary structure of proteins; this is a serious drawback since the chemical properties of proteins depend on their secondary and tertiary structures. Nevertheless, this is a novel use of process algebras and illustrates the possibility of their having uses well beyond their first intentions.