Generating Behavioural Traces in LCC

Dave Robertson

This document defines the syntax and operational semantics of the version of LCC used in Prolog-based LCC simulation experiments. It is intended to be used as documentation for the Prolog source code of the basic trace generator accompanying this document.

1 Syntax

Figure 1 defines the syntax of the Lightweight Coordination Calculus (LCC). An interaction model in LCC is a set of clauses, each of which defines how a role in the interaction must be performed. Roles are described by the type of role and an identifier for the individual peer undertaking that role. The definition of performance of a role is constructed using combinations of the sequence operator ('*then*') or choice operator ('*or*') to connect messages and changes of role. Messages are either outgoing to another peer in a given role ('$\Rightarrow$') or incoming from another peer in a given role ('$\Leftarrow$'). Message input/output or change of role can be governed by a constraint defined using the normal logical operators for conjunction, disjunction and negation. Notice that there is no commitment to the system of logic through which constraints are solved - so different peers might operate different constraint solvers (including human intervention).

$Model$	:=	$\{Clause, \ldots\}$
$Clause$	:=	$Role :: Def$
$Role$	:=	$a(Type, Id)$
Def	:=	$Role \mid Message \mid Def\ then\ Def \mid Def\ or\ Def$
$Message$	:=	$M \Rightarrow Role \mid M \Rightarrow Role \leftarrow C \mid M \Leftarrow Role \mid C \leftarrow M \Leftarrow Role$
C	:=	$Constant \mid P(Term, \ldots) \mid \neg C \mid C \wedge C \mid C \vee C$
$Type$	:=	$Term$
Id	:=	$Constant \mid Variable$
M	:=	$Term$
$Term$	:=	$Constant \mid Variable \mid P(Term, \ldots)$
$Constant$	:=	lower case character sequence or number
$Variable$	:=	upper case character sequence or number

Figure 1: LCC syntax

2 Generating Traces For Interaction Models

In our trace generator, a sequence of transitions is initiated by a single peer with some goal to achieve and the sequence terminates successfully only when that peer can infer from its local knowledge of the interaction that it has obtained that goal.

Expressions 1 and 2 in Figure 2 generate transition sequences. Expression 3 in Figure 2 describes the general form of any such sequence. Following through expression 3 for a peer $p1$ attempting to establish goal G_{p1}, it begins with the selection by $p1$ of an interaction model ($\Omega \overset{p1}{\ni} \mathcal{S}_1$); then selection of the local state pertaining to $p1$ from the shared model ($\mathcal{S}_1 \overset{s}{\supseteq} \mathcal{S}_{p1}$); then a transition of the local state for $p1$ according to the model ($\mathcal{S}_{p1} \xrightarrow{M_1,\mathcal{S}_1,M_2} \mathcal{S}'_{p1}$); then merging of the ensuing local state with the shared state to produce a new shared interaction state ($\mathcal{S}'_{p1} \overset{s}{\cup} \mathcal{S}_1 = \mathcal{S}_2$); then repeating the transitions until reaching a final state in which the required goal can be derived using $p1$'s local knowledge ($k_{p1}(\mathcal{S}_f) \vdash G_{p1}$).

Having defined, in Figure 2, a formal model of interaction, we describe in Sections 2.1 to 2.4 how this connects to the LCC language. The operations that our language must support in order to conform to Figure 2 are given as part of each section heading.

2.1 Initial Interaction State: $\Omega \overset{p}{\ni} \mathcal{S}$

For a peer, p, to initiate an interaction it must select an appropriate initial state, S, from the set of possible such initial states, Ω. This is determined by some choice made by p based on the interaction model, $\mathcal{P}_g$, and shared knowledge, K, components of the initial states (recall that in LCC the initial state is a term of the form $m(\phi, \mathcal{P}_g, K)$). We denote this choice as $c(p, \mathcal{P}_g, K)$ in the expression below but do not define here the mechanism by which that choice is made, since it varies according to application - anything from a fully automated choice to a decision made by a human operator.

$$\Omega \overset{p}{\ni} \mathcal{S} \;\; \leftrightarrow \;\; \mathcal{S} \in \Omega \;\wedge S = m(\phi, \mathcal{P}_g, K) \;\wedge\; c(p, \mathcal{P}_g, K) \tag{4}$$

2.2 State Selection by a peer: $\mathcal{S} \overset{s}{\supseteq} \mathcal{S}_p$

Given that in LCC the state of interaction always is expressed as a term of the form $m(\mathcal{P}_s, \mathcal{P}_g, K)$, the selection of the current state for a peer, p, simply requires the selection of the appropriate clause, $a(R, p) :: D$, defining (in D) the interaction stare for p when performing role R.

$$\mathcal{S} \overset{s}{\supseteq} \mathcal{S}_p \;\; \leftrightarrow \;\; \exists R, D.(\mathcal{S}_p \in \mathcal{S} \;\wedge\; \mathcal{S}_p = a(R, p) :: D) \tag{5}$$

$$\sigma(p, G_p) \quad \leftrightarrow \quad \Omega \overset{p}{\ni} \mathcal{S} \ \wedge \ i(\mathcal{S}, \phi, \mathcal{S}_f) \ \wedge \ k_p(\mathcal{S}_f) \vdash G_p \tag{1}$$

$$i(\mathcal{S}, M_i, \mathcal{S}_f) \quad \leftrightarrow \quad \mathcal{S} = \mathcal{S}_f \ \vee \begin{pmatrix} \mathcal{S} \overset{s}{\supseteq} \mathcal{S}_p \wedge \\ \mathcal{S}_p \xrightarrow{M_i, \mathcal{S}, M_n} \mathcal{S}'_p \wedge \\ \mathcal{S}'_p \overset{s}{\cup} \mathcal{S} = \mathcal{S}' \wedge \\ i(\mathcal{S}', M_n, \mathcal{S}_f) \end{pmatrix} \tag{2}$$

where:

- p is a unique identifier for a peer.
- G_p is a goal that peer P wants to achieve.
- $\mathcal{S}$, is a state of interaction which contains the interaction model used to coordinate peers; the knowledge shared by peers participating in the interaction; and a description of their current progress in pursuing the interaction. Ω is the set of all the available initial interaction states. $\Omega \overset{p}{\ni} \mathcal{S}$ selects an initial interaction state, $\mathcal{S}$, for peer P from Ω.
- M is the current set of messages sent by peers. The empty set of messages is ϕ.
- $\sigma(P, G_p)$ is true when goal G_p is attained by peer P.
- $i(\mathcal{S}, M_1, \mathcal{S}_f)$ is true when a sequence of interactions allows state $\mathcal{S}_f$ to be derived from $\mathcal{S}$ given an initial set of messages M_1.
- $k_p(\mathcal{S})$ is a function giving the knowledge visible to peer P contained in state $\mathcal{S}$.
- $\mathcal{S} \overset{s}{\supseteq} \mathcal{S}_p$ selects the state, $\mathcal{S}_p$, pertaining specifically to peer P from the interaction state $\mathcal{S}$.
- $\mathcal{S}_p \xrightarrow{M_i, \mathcal{S}, M_n} \mathcal{S}'_p$ is a transition of the state of peer P to a new state, $\mathcal{S}'_p$, given the current set of inter-peer messages, M_i, and producing the new set of messages M_n.
- $\mathcal{S}_p \overset{s}{\cup} \mathcal{S}$ is a function that merges the state, $\mathcal{S}_p$, specific to peer P with interaction state $\mathcal{S}$ (replacing any earlier interaction state for peer P).

Every successful, terminating interaction satisfying $\sigma(p1, G_{p1})$ can then be described by the following sequence of relations (obtained by expanding the 'i' relation within expression 1 using expression 2):

$$\Omega \overset{p1}{\ni} \mathcal{S}_1 \overset{s}{\supseteq} \mathcal{S}_{p1} \xrightarrow{M_1, \mathcal{S}_1, M_2} \mathcal{S}'_{p1} \overset{s}{\cup} \mathcal{S}_1 = \mathcal{S}_2 \overset{s}{\supseteq} \mathcal{S}_{p2} \xrightarrow{M_2, \mathcal{S}_2, M_3} \mathcal{S}'_{p2} \overset{s}{\cup} \mathcal{S}_2 = \mathcal{S}_3 \ldots k_{p1}(\mathcal{S}_f) \vdash G_{p1} \tag{3}$$

Figure 2: Formal model of linearised peer interaction

2.3 State Transition by a peer: $\mathcal{S}_p \xrightarrow{M_i,\mathcal{S},M_n} \mathcal{S}'_p$

Recall that, from Section 2.2 we can know the state of a specific role in the interaction by selecting the appropriate clause. This clause gives us $\mathcal{S}_p$ and we now explain how to advance the state associated with this role to the new version of that clause, $\mathcal{S}'_p$, given an input message set, M_i, and producing a new message set, M_n, which contains those messages from M_i that have not been processed plus additional messages added by the state transition. Since we shall need a sequence of transitions to the clause for $\mathcal{S}_p$ we use C_i to denote the start of that sequence and C_j the end. The rewrite rules of Figure 3 are applied to give the transition sequence of expression 6.

$$C_i \xrightarrow{M_i,\mathcal{S},M_n} C_j \;\leftrightarrow\; \exists R, D.(C_i = a(R,p) :: D) \wedge \left(\begin{array}{lll} C_i & \xrightarrow{R_i,M_i,M_{i+1},\mathcal{S},O_i} & C_{i+1} \wedge \\ C_{i+1} & \xrightarrow{R_i,M_{i+1},M_{i+2},\mathcal{S},O_{i+1}} & C_{i+2} \wedge \\ \cdots & & \\ C_{j-1} & \xrightarrow{R_i,M_{j-1},M_j,\mathcal{S},O_j} & C_j \end{array} \right) \wedge M_n = M_j \cup O_j \tag{6}$$

2.4 Merging Interaction State: $\mathcal{S}_p \overset{s}{\cup} \mathcal{S} = \mathcal{S}'$

The interaction state, $\mathcal{S}$, is a term of the form $m(\mathcal{P}_s, \mathcal{P}_g, K)$ and the state relevant to an individual peer, $\mathcal{S}_p$, always is a LCC clause of the form $a(R,p) :: D$. Merging $\mathcal{S}_p$ with $\mathcal{S}$ therefore is done simply by replacing in $\mathcal{S}$ the (now obsolete) clause in which p plays role R with its extended version $\mathcal{S}_p$.

$$(a(R,p) :: D) \overset{s}{\cup} \mathcal{S} \;=\; (\mathcal{S} \overset{s}{-} \{a(R,p) :: D')\}) \cup \{a(R,p) :: D)\} \tag{8}$$

2.5 Interaction-Specific Knowledge: $k_p(\mathcal{S}) \vdash G_p$

Shared knowledge in LCC is maintained in the set of axioms, K, in the interaction state $m(\mathcal{P}_s, \mathcal{P}_g, K)$ so a peer's goal, G_p, can be satisfied if it is satisfiable from K or through the peer's own internal satisfiability mechanisms. This corresponds to the $satisfied$ relation introduced with the rewrite rules of Figure 3.

$$k_p(\mathcal{S}) \vdash G_p \;\leftrightarrow\; satisfied(\mathcal{S}, G_p) \tag{9}$$

This completes our operational specification for LCC when combined with the style of linear deployment given in Figure 2.

$$
\begin{array}{lll}
R :: B \xrightarrow{R_i,M_i,M_o,\mathcal{S},O} A :: E & \text{if} & B \xrightarrow{R,M_i,M_o,\mathcal{S},O} E \\
A_1 \; or \; A_2 \xrightarrow{R_i,M_i,M_o,\mathcal{S},O} E & \text{if} & \neg closed(A_2) \wedge \\
 & & A_1 \xrightarrow{R_i,M_i,M_o,\mathcal{S},O} E \\
A_1 \; or \; A_2 \xrightarrow{R_i,M_i,M_o,\mathcal{S},O} E & \text{if} & \neg closed(A_1) \wedge \\
 & & A_2 \xrightarrow{R_i,M_i,M_o,\mathcal{S},O} E \\
A_1 \; then \; A_2 \xrightarrow{R_i,M_i,M_o,\mathcal{S},O} E \; then \; A_2 & \text{if} & A_1 \xrightarrow{R_i,M_i,M_o,\mathcal{S},O} E \\
A_1 \; then \; A_2 \xrightarrow{R_i,M_i,M_o,\mathcal{S},O} A_1 \; then \; E & \text{if} & closed(A_1) \wedge \\
 & & A_2 \xrightarrow{R_i,M_i,M_o,\mathcal{S},O} E \\
C \leftarrow M \Leftarrow A \xrightarrow{R_i,M_i,M_i-\{m(R_i,M \Leftarrow A)\},\mathcal{S},\emptyset} c(M \Leftarrow A) & \text{if} & m(R_i, M \Leftarrow A) \in M_i \wedge \\
 & & satisfy(C) \\
M \Rightarrow A \leftarrow C \xrightarrow{R_i,M_i,M_o,\mathcal{S},\{m(R_i,M \Rightarrow A)\}} c(M \Rightarrow A) & \text{if} & satisfied(\mathcal{S}, C) \\
null \leftarrow C \xrightarrow{R_i,M_i,M_o,\mathcal{S},\emptyset} c(null) & \text{if} & satisfied(\mathcal{S}, C) \\
a(R, I) \leftarrow C \xrightarrow{R_i,M_i,M_o,\mathcal{S},\emptyset} a(R, I) :: B & \text{if} & clause(\mathcal{S}, a(R, I) :: B) \wedge \\
 & & satisfied(\mathcal{S}, C)
\end{array}
$$

An interaction model term is decided to be closed as follows:

$$
\begin{array}{l}
closed(c(X)) \\
closed(A \; then \; B) \leftarrow closed(A) \wedge closed(B) \\
closed(X :: D) \leftarrow closed(D)
\end{array}
\tag{7}
$$

$satisfied(\mathcal{S}, C)$ is true if constraint C is satisfiable given the peer's current state of knowledge.
$clause(\mathcal{S}, X)$ is true if clause X appears in the interaction model $\mathcal{S}$, as defined in Figure 1.

Figure 3: Rewrite rules for expansion of an interaction model clause