A NETWORK OF COMMUNICATING LOGIC PROGRAMS AND ITS SEMANTICS

Susumu Yamasaki

Department of Information Science, Kyoto University
Sakyo, Kyoto, Japan

ABSTRACT

In this paper, a network of communicating loqic programs is proposed as a model for parallol/concurrent programming based on logic programs. This network is regarded as an extension of Kahn's pure dataflow in the sense that nodes are logic programs which have atoms for receiving and sending messages as well as queues to accept and memorize them. The nodes' behaviour is unboundedly nondeterministic. On the assumption that the channels' denotations should be defined by using sequence domains, the main concern of the present paper is whether or not mathematical semantics is always well-defined for a given network even when it is of unbounded nondeterminism.

This paper will show that the proposed network is reduced to a dataflow when a kind of 'fairness' is asked for nondeterminism in which logic programs receive inputs and produce outputs based on their computations. The network, then, has a (least) fixpoint semantics; it is regarded as one of mathematical semantics of the network, since it can satisfy the recursive relations among the channels' denotations. It is also stated that the network with fair merge operators is applicable to the realization of a computational mechanism for sequential ligic programs.

1. NTRODUCTION

Kahn's pure dataflow network is one of the established models for parallel/concurrent programming in the sense that its mathematical semantics is clearly defined by the least fixpoint solution to the set of equations associated with the channels of the network [4,5].

When establishing a model for parallel/concurrent programming based on logic programs, Kalin's pure dataflow can be extended to a network whose nodes are not functional but relational in accordance with logic programs. Jf we establish a network according to such a model, there would be a strict question as to whether or not mathematical semantics of the network is well-defined even when it is of unbounded nondeterminism. The question arises from the standpoint that mathematical semantics should be made clear in order for operational semantics to be given so that implementation of the system is possible. In both PARLOG (2,3] and Concurrent Prolog [10] the channels' semantics are indirectly defined, hence the communications between processes seem complicated.

In general, unbounded nondeterminism due to the computation of logic programs should be respected and mathematical semantics should be defined based on the relations among the denotations of channels. Hence the fixpoint theory in [4] is not appropriate in this case. Even theories concerning powerdomain constructions in [9,11] are unsuccessful for the network we intend to define, since, in essence, the network is described by relations among the denotations of the channels.

On the other hand, *one* way to extend Kahn's model to a more general one is to establish *a* class of nondeterministic computing networks whose nodes are either asynchronous, continuous functions or fair merge operators. Fixpoint semantics of such networks is well-defined in [8]. The 'fairness' applied to such networks might be useful to solve the question of mathematical semantics of the unbounded nondeterministic network of logic programs. However, it seems that nodes consisting of logic programs contain more nondeterminism than the fair merge operations, since the choices of outputs produced by their computations are not only due to the relative timings of inputs but due to the decisions concerning their relations. At the same time, there is a crucial problem of how 'fairness* can be realized for nondeterminism in which logic programs are computed.

In this paper, we have a fair sequence in which any natural number occurs an arbitrary number of times. By means of the fair sequence it will be shown that fairness is well realized for nondeterministic receivinq of messages, and sending of messages related to the computations of logic programs, in the network. The network consisting of communicating logic programs might be reduced to a dataflow when fairness due to the fair sequence is asked. Thus, the network has *a* (least) fixpoint semantics for the reduced dataflow, according to [8]. The fixpoint semantics could be regarded as one of the mathematical semantics for the original network. According to the fixpoint semantics, an operational semantics will be shown on the condition that logic programs in the network are fairly nondeterministic in receiving inputs and sending outputs. If the intended network has fair merge operators, it can express a computational mechanism for sequential logic programs. It means that the semantics of sequential logic programs might be defined by applying the semantics of our network with fair merge operators.

The outline of the network is as follows: The network consists of channels and processes. The channels are only ways by which the processes may communicate with each other. There may be channels for inputs and/or outputs. The message is transmitted one way through each channel within a sufficiently short time. Each process is a logic program consisting of definite clauses with atoms corresponding to receiving and sending messages as well as unbounded FIFO queues each of which is connected with an input channel transmitting messages. In each process, the messages in the nondeterministically chosen positions of the queues are received for the computation of the logic programs, simultaneously. After the logic program*s finite computation, each process nondeterministically provides either one of possible outputs or no output.

* Present Affiliation: Department of Information Science, Okayama University, Okayama, Japan.

**This report describes work initiated at Department of Computer Science, University of Warwick, the United Kingdom. It was supported in part by Yamada Science Foundation.

2 NETWORK OF COMMUNICATING LOGIC PROGRAMS (NCLP)

A logic program is a set of definite Horn clauses. A definite Horn clause is either an expression $A \leftarrow B_1 \ldots B_n$ or $A \leftarrow$ for atoms A and B_i. An atom is an expression $P(t_1,\ldots,t_m)$, where P is a predicate symbol and $t_1,\ldots,t_m$ are terms. If P(t) is an atom for a term t, P is called monadic. A term is defined recursively as follows: (1) A constant or a variable is a term. (2) $f(t_1,\ldots,t_k)$ is a term if f is a function symbol and $t_1,\ldots,t_k$ are terms.

The atom on the left-hand side of <- in a definite Horn clause is called the head of the clause. The set of atoms (which might be empty) on the right-hand side of <- in a definite Horn clause is called its body.

A network of communicating logic programs (NCLP) is N=(C,L,I,O,CombI,CombO),

where: (1) $C=\{C_1,\ldots,C_m\}$ is a set of channels.

(2) $L=\{L_1,\ldots,L_n\}$ is a set of logic programs with queues. For $1 \le j \le n$, $L_j=(S_j,q_j)$, where

(i) S_j is a logic program such that

(a) it contains the atoms whose predicate symbols are $Se^k\text{-}D_{jk}$ (monadic) for D_{jk} in C ($D_{jk} \cap I$ is empty, $k=1,\ldots,n_j$, and $D_{jk} \cap D_{jh}$ is empty iff $k \ne h$) in the heads of some clauses in S_j, and

(b) it might contain the atoms $Re\text{-}C_e(x_e)$ for channels C_e and variables x_e in the bodies of some clauses in S_j,

(ii) q_j is a tuple of unbounded queues.

(3) I is a subset of C, called an input channel set. O is a subset of C, called an output channel set.

(4) CombI is a function from L to 2^C such that for j $\mathrm{CombI}(L_j)=\{C_e \mid Re\text{-}C_e \text{ appears in } S_j\}$.

CombO is a relation on $L \times 2^C$ such that for each L_j, (L_j,D_{jk}) is in CombO. From now on, $\mathrm{CombO}(L_j,D_{jk})$ is used to mean that (L_j,D_{jk}) is in CombO.

3 DENOTATIONAL SEMANTICS OF NCLP

It is assumed in the present paper that the logic programs in NCLP are communicating based on their finite computations. On this assumption, the semantics of logic programs are given by their minimal Herbrand models [3,12].

3.1. Semantic Domain

(1) Hu denotes the Herbrand universe constructed over the set of constants and function symbols in $s_1,\ldots,s_n$ and in the channels of I.

(2) H_B means the Herbrand base formed by the predicate symbols in $s_1,\ldots,s_n$ and by the terms in Hu.

(3) Let hi be a special symbol not in Hu. (Note that hi is a hiaton used in [8], which is to denote a time-delay.)

(4) Let $D=Hu \cup \{hi\}$, and let D^∞ denote the set of all finite and infinite sequences from D. λ denotes the empty sequence in D^∞. A partial order $\sqsubseteq$ on D^∞ is defined by: $u \sqsubseteq v$ for u, v in D^∞ iff v=uw for some w in D^∞.

Note that any chain $\{u_1 \sqsubseteq u_2 \sqsubseteq \ldots \sqsubseteq u_n\}$ has a least upper bound, which is denoted by $\sqcup_p u_p$.

The partial order $\sqsubseteq$ is extended to the one on $(D^\infty)^n$ by: $(u_1,\ldots,u_n) \sqsubseteq (v_1,\ldots,v_n)$ iff $u_p \sqsubseteq v_p$ for $1 \le p \le n$.

ω denotes the set of natural numbers.

3.2. Denotations of Channels

For $i=1,\ldots,m$, X_i in D^∞ means the denotation of C_i in C, which is to be defined by the relations as shown in 3.4.

3.3. Denotations of Logic Programs

For $j=1,\ldots,n$,

(1) (i) the denotation of q_j is $d\text{-}q_j=(E(X_{j1}),\ldots,E(X_{jh}))$ where (a) $\{C_{j1},\ldots,C_{jh}\}=\mathrm{CombI}(L_j)$, and

(b) $E: D^\infty \to Hu^\infty$ is recursively defined by: $E(\lambda)=\lambda$; $E(hi.u)=E(u)$; $E(a.u)=a.E(u)$ for $a \ne hi$,

(ii) $Q_j\text{-}C_{ji}$ means $E(X_{ji})$ of $d\text{-}q_j$, for C_{ji} in $\mathrm{CombI}(L_j)$,

(2) the denotation of S_j, for the inputs of length p, is defined by

$$R_j(p,d\text{-}q_j)=\bigcup_{r_1,\ldots,r_h \le p} V_j(r_1,\ldots,r_h,d\text{-}q_j),$$

where $V_j(r_1,\ldots,r_h,d\text{-}q_j)=(\min I)T_j(r_1,\ldots,r_h,d\text{-}q_j,I)$ for $T_j: \omega^h \times (Hu^\infty)^{\#\mathrm{CombI}(L_j)} \times \mathrm{Pow}(H_B) \to \mathrm{Pow}(H_B)$ ($\mathrm{Pow}(H_B)$: the set 2^{H_B}, #Y: the cardinal number of a set Y), defined by

$$T_j(r_1,\ldots,r_h,d\text{-}q_j,I) = \{A\theta \text{ in } H_B \mid A \leftarrow B_1 \ldots B_x \text{ is in } S_j, \text{ and } \{B_1\theta,\ldots,B_x\theta\} \subseteq I \cup \bigcup_{C_{ji} \text{ in } \mathrm{CombI}(L_j)} \{Re\text{-}C_{ji}(Q_j\text{-}C_{ji}(r_j))\}\}$$

and for

$$(\min I)T_j(r_1,\ldots,r_h,d\text{-}q_j,I)=\bigcap\{J \mid T_j(r_1,\ldots,r_h,d\text{-}q_j,J) \subseteq J\}.$$

3.4. Relations among Denotations of Channels

For $1 \le j \le n$,

(1) let $W_{jk}(r_1,\ldots,r_h,d\text{-}q_j)$ be

$\{t \text{ in } Hu \mid Se^k\text{-}D_{jk}(t) \text{ is in } V_j(r_1,\ldots,r_h,d\text{-}q_j)\}$ $(k=1,\ldots,n_j)$,

(2) for each C_i in D_{jk} such that $\mathrm{CombO}(L_j,D_{jk})$, and for any $r_1,\ldots,r_h$ in ω,

$$X_i(p) \text{ is in } \bigcup_{r_1,\ldots,r_h \le p} W_{jk}(r_1,\ldots,r_h,d\text{-}q_j)$$

if $|E(X_e)| \ge p$ ($|u|$ means the length of u) for some e such that C_e is in $\mathrm{CombI}(L_j)$ or if $\#\mathrm{CombI}(L_j)=0$, and

$X_i(p)=hi$ otherwise,

(3) for each C_i in I,

X_i = the element in D^∞, provided through C_i.

The semantics of the network N is $(E(X_1),\ldots,E(X_m))$ for $(X_1,\ldots,X_m)$ which is in

$\mathrm{Sem}(N)=\{(X_1,\ldots,X_m) \mid X_1,\ldots,X_m$ satify the relations in (2) and (3)$\}$.

Note that $V_j(r_1,\ldots,r_h,d\text{-}q_j)$ and $W_{jk}(r_1,\ldots,r_h,d\text{-}q_j)$ depend on the denotation of q_j.

Note that for C_i in D_{jk} such that $\mathrm{CombI}(L_j, D_{jk})$, there are $1+\#(\bigcup_{r_1,\ldots,r_h \leq p} W_{jk}(r_1,\ldots,r_h, d\text{-}q_j))$ choices for $X_i(p)$.

Such nondeterminism is due to

(1) what numbers $r_1,\ldots,r_h$ are chosen among $\{1,\ldots,p\}$, to indicate the positions of queues of q_j on which the denotations of 'Re-C_e'-atoms depend, and

(2) what element is chosen in $W_{jk}(r_1,\ldots,r_h, d\text{-}q_j)^{\omega}\{hi\}$, to indicate the denotation of 'Se-D_{jk}'-atoms.

Therefore, there may be as an infinitely many number of choices as ω, when X_i is fixed.

We shall have a class of fair sequences in ω^{ω} as oracles in order that the choices for the positions on the queues and for the atoms' denotations may be made fairly under such nondeterminism. If the oracles are used, the denotations of channels can be described by a continuous, asynchronous function from a direct product of D^{∞} to itself.

4.1. Fair Sequence in ω^{ω}

DEFINITION 1 u in ω^{ω} is fair iff for any k in ω and for any h in ω, k occurs in u, h times.

DEFINITION 2 Let S_f denote one of sequences in ω^{ω}, provided by the network as shown below, where:

(1) T: $\omega^{\infty} \to \omega^{\infty}$ is defined by

$$T(\lambda) = \lambda \; ; \quad T(a.u)=(a+1).T(u) \quad \text{for } u \text{ in } \omega^{\infty}.$$

(2) FM: $\omega^{\infty} \times \omega^{\infty} \to \omega^{\infty}$ means a fair merge operator, that is, FM(u,v)=w iff w is a sequence obtained by interleaving u and v in a fairly nondeterministic way. (For the fair merge, see [6,7,8]).

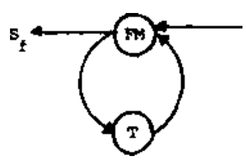

Fig. 1 A Network to Provide Fair Sequences

DEFINITION 3 A function F: $D^{\infty} \to Hu^{\infty}$ is asynchronous if F(E(u))=F(u) for any u in D^{∞}.
F is continuous if $F(\uparrow_p u_p) = \uparrow_p F(u_p)$ for any chain $\{u_1 \uparrow u_2 \uparrow \ldots \uparrow u_n \uparrow \ldots\}$.

PROPOSITION 1

(1) S_f is fair in ω^{ω}.

(2) $S_f(j)+1 \leq j$.

Proof (1) Note that $S_f = 0^{+}.u$ for some u in ω^{ω}. Now let Occur(k,h) mean that k occurs h times in u.

(i) It is seen that Occur(0,h) for any h in ω, because 0^{ω} is an input of FM and thus 0 appears as its output an arbitrary number of times.

(ii) Assume that for some h, Occur(n,h) for $n \leq k$. Then, since k+1 should be at the same time an output of T and hence an input of FM, Occur(k+1,h). By mathematical induction, for some h in ω, Occur(k,h) (for any k).

(iii) Since Occur(0,1) from (1), it follows from (2) that Occur(n,1) for any n in ω. Suppose that Occur(k,h) for any k in ω and $h \leq m$. Since k should be an output of T, k-1 should be an input of T, that is, k-1 occurs in u as many times as k. Thus Occur(k-1,h). Finally Occur(0,h). It follows from (1) that Occur(0,h+1). Therefore there is no case that Occur(k,h+1) does not hold. This completes the induction step of mathematical induction. Thus Occur(k,m) for any k and for any m.

(2) (i) If j=1, the proposition holds, since $0 \uparrow S_f$.
(ii) Assume that $S_f(h)+1 \leq h$ for $h \leq k$. Since one of the inputs of FM is $S_f(k)+1$ or 0, $E(S_f)(k+1)=E(S_f)(k)+1$ or 0. Thus $S_f(k+1)+1 \leq (S_f(k)+1)+1 \leq k+1$. This completes the induction step.

4.2. A Fixpoint Semantics of NCLP

By means of fair sequences in ω^{ω} as oracles, the denotations of channels can be defined as determinant for each logic programs.

For j=1,...,n, let $P_{jk}(p)$: $W_{jk}(I_j(p), d\text{-}q_j) \to \omega$ (for p in ω) be a one-to-one function such that

$$P_{jk}(p)(t) \leq \#W_{jk}(I_j(p), d\text{-}q_j)$$

for t in $W_{jk}(I_j(p), d\text{-}q_j)$, where k=1,...,h and $I_j: \omega \to \omega^h$ is a bijection such that any element of $I_j(p) \leq p$.
Let S_f-j denote a sequence in ω^{ω}, given by DEFINITION 2. Also let

$$\mathrm{Ord}(S_f\text{-}j(p)) = \#\{\, r \mid S_f\text{-}j(r)=S_f\text{-}j(p) \text{ and } r \leq p \,\}.$$

For $k=1,\ldots,n_j$ and each C_i in D_{jk} such that $\mathrm{CombO}(L_j, D_{jk})$, we define

$$X_i(p) = P_{jk}(S_f\text{-}j(p)+1)^{-1}(\min\{\, S_f\text{-}j(\mathrm{Ord}(S_f\text{-}j(p)))+1,\ \#W_{jk}(I_j(S_f\text{-}j(p)+1), d\text{-}q_j)\})$$

if $|E(X_e)| \geq p$ for some e such that C_e is in $\mathrm{CombI}(L_j)$ or if $\#\mathrm{CombI}(L_j)=0$, and

$X_i(p)=hi$ otherwise.

PROPOSITION 2 For i=1,...,m, there exist asynchronous, continuous functions $F_i: (D^{\infty})^m \to D^{\infty}$ such that

$$X_i = F_i(X_1,\ldots,X_m).$$

Proof Since either C_i is in I or C_i is in D_{jk} for some D_{jk} in 2^C such that $\mathrm{CombO}(L_j, D_{jk})$, X_i is defined by either a given element in D^{∞} through C_i or by the above formula.

Since $P_{jk}(S_f\text{-}j(p)+1)^{-1}$ is a one-to-one function from ω to $W_{jk}(I_j(S_f\text{-}j(p)+1), d\text{-}q_j)$, and the denotation of $d\text{-}q_j$ is defined by X_e for e such that C_e is in $\mathrm{CombI}(L_j)$, we see $X_i(p)$ is a function of $X_e[p]$. (By $X_e[p]$, the first sequence of length p in X_e is meant.) Thus, we could put

$$X_i = F_i(X_1,\ldots,X_m) \text{ for } F_i: (D^{\infty})^m \to D^{\infty}.$$

It follows from $d\text{-}q_j = (E(X_{j1}),\ldots,E(X_{jh}))$ that F_i are asynchronous. Since $X_i[p]=F_i(X_1[p],\ldots,X_m[p]) \uparrow X_i = F_i(X_1,\ldots,X_m) =$

$F_i(\uparrow_p X_1[p],\ldots,\uparrow_p X_m[p])$,

$\uparrow_p F_i(X_1[p],\ldots,X_m[p]) \uparrow F_i(X_1,\ldots,X_m)$.

On the other hand, for any p in ω, $X_i[p]=F_i(X_1,\ldots,X_m)[p] \uparrow$

$F_i(X_1[p],\ldots,X_m[p])$, since $X_i(p)$ is determined by $X_1[p],\ldots,X_m[p]$. Therefore

$$F_i(X_1,\ldots,X_m)=X_i=\uparrow_p X_i[p] \uparrow \uparrow_p F_i(X_1[p],\ldots,X_m[p]).$$

This completes the proof of the continuity of F_i.

This proposition states that the network is reducible to a dataflow by means of fair sequences in ω^ω. Finally we have the following proposition:

PROPOSITION 3
For the network N=({$C_1,\ldots,C_m$}, {$L_1,\ldots,L_n$}, I, O, CombI, CombO) there exists a function $F_N: (D^\infty)^m \to (D^\infty)^m$ such that the (least) fixpoint of F_N is in Sem(N).

Proof F_N is given by $(F_1,\ldots,F_m)$, where F_i are asynchronous, continuous functions. Thus there exists a (least) fixpoint of F_N [8]. F_i are in accordance with the formula we have already obtained. Hence X_i satisfy the relations as shown in 3.4 (2). Finally the fixpoint of F_N is in Sem(N).

4.3. Operational Semantics of NCLP

In this section, we have an operational semantics of NCLP based on its fixpoint semantics given in the previous section. In operational semantics, the hiaton hi used in denotational semantics, corresponds to the states of waiting messages and/or producing nothing. The denotations of 'Re-C_e'-atoms correspond to receiving messages and those of 'Se-D_{jk}'-atoms correspond to sending messages.

4.3.1. Assumptions

(1) The messages sent to each channel C_e should be stored in the unbounded FIFO q_j of L_j, when C_e is in CombI(L_j).

(2) When a logic program L_j wants a longer queue, in order to generate (send) the next output, it must wait for the necessary messages arriving at its queue.

(3) Unless a logic program can get something by its computations, it never sends any message to channels.

DEFINITION 4 For a nonempty set A, FairChoice(A) means some b in A is chosen at any time such that the sequence of chosen elements is fair in A^ω. (A sequence is fair in A^ω if any b in A occurs an arbitrary number of times in the sequence.)

DEFINITION 5 For a set S of definite clauses, Comput(S) denotes the set of ground atoms, which is the minimal Herbrand model of S.

4.3.2. Procedure for NCLP

DEFINITION 5
Let N=({$C_1,\ldots,C_m$}, {$L_1,\ldots,L_n$} ,I,O,CombI,CombO).
Let $q_j=(Hu^\infty)^{\#CombI(L_j)}$, that is, a tuple of queues. The length of q_j is the maximum length among lengths of sequences stored in q_j. Q_j-C_e means the content of queue of q_j, connected to the channel C_e such that C_e is in CombI(L_j). For p in ω, Q_j-$C_e(p)$ denotes the p-th content it holds.

The procedure f_j for L_j is given as follows:

```
procedure f_j;
 begin
 p_j<-1;
 while p_j in ω do
   begin
   repeat wait until (the length q_j)≥p_j;
   (r_j1,...,r_jh)<-I_j(min { p_j,FairChoice( ω )+1 } );
   for each D_jk such that CombO(L_j,D_jk) do
      H_j(r_j1,...,r_jh)
     <-H_j(r_j1,...,r_jh)
      ∪{ t in Hu | Se^k-D_jk(t) is in
              Comput(S_j ∪{ Re-C_j1(Q_j-C_j1(r_j1))}
                          ......
                          ......
                        ∪{ Re-C_jh(Q_j-C_jh(r_jh))} } ;
     if H_j(r_j1,...,r_jh) is not empty
     then send t in FairChoice(H_j(r_j1,...,r_jh))
     to each channel of D_jk
     else send nothing;
   p_j<-p_j+1
   end
 end
```

(Note: $H_j(r_{j1},\ldots,r_{jh})$ is initially defined as empty for each tuple of $r_{j1},\ldots,r_{jh}$.)

The procedure for N is defined as follows:

program N;
 cobegin f_1; f_2;...; f_n coend

5. APPLICATION OF NCLP

This section is an overlook of the application of NCLP to the transformation of sequential logic programs to dataflow networks with fair merge operators. Any sequential logic program denotes an NCLP with fair merge operbors. It follows from the results of the present paper and [8] that the denotational semantics of a logic program might be defined as a fixpoint over sequence domains.

Now let L be a l c program { $H_1,\ldots,H_n$ } , r e each H_i is a definite clause A_i<-$B_{i1}\ldots B_{in_i}$ for atoms

$A_i = P_i(t_{i1}, \ldots, t_{im_i})$ and $B_{ij} = Q_{ij}(u_{ij1}, \ldots, u_{ijk_{ij}})$.

Let $\rightarrow\!\!>$ be a subset of $L \times L$ such that (H_i, H_j) is in $\rightarrow\!\!>$ (which is denoted by $H_i \rightarrow\!\!> H_j$) iff P_i is equal to Q_{jk} for some $1 \le j \le n$ and $1 \le k \le n_j$. NOND: $L \rightarrow\!\!> \omega$ is defined by: $NOND(H_j) = \# \{ H_i \mid H_i \rightarrow\!\!> H_j \}$. Let $FM(p,q)$ mean a fair merge operator with p input-tuples of q channels and one output-tuple of q channels, which provides a sequence of tuples by interleaving, in a fairly nondeterministic way, p input-tuples of sequences.

For each H_i in L, we define

$$
\begin{aligned}
& S(H_i) \\
= \{ & H_i, \\
& Q_{i1}(u_{i11}, \ldots, u_{i1k_{i1}}) \leftarrow Re\text{-}C_{i11}(x_{i11})\ x_{i11} = u_{i11} \\
& \qquad \ldots\ldots \\
& \qquad Re\text{-}C_{i1k_{i1}}(x_{i1k_{i1}})\ x_{i1k_{i1}} = u_{i1k_{i1}}, \\
& \ldots\ldots, \\
& \ldots\ldots, \\
& Q_{in_i}(u_{in_i1}, \ldots, u_{in_ik_{in_i}}) \leftarrow Re\text{-}C_{in_i1}(x_{in_i1})\ x_{in_i1} = u_{in_i1} \\
& \qquad \ldots\ldots \\
& \qquad Re\text{-}C_{in_ik_{in_i}}(x_{in_ik_{in_i}}) \\
& \qquad x_{in_ik_{in_i}} = u_{in_ik_{in_i}}, \\
& Se\text{-}\{ C_{H_{i1}} \} (t_{i1}) \leftarrow P_i(t_{i1}, \ldots, t_{im_i}), \\
& \ldots\ldots, \\
& Se\text{-}\{ C_{H_{im_i}} \} (t_{im_i}) \leftarrow P_i(t_{i1}, \ldots, t_{im_i}) \}
\end{aligned}
$$

where

$(C_{i11}, \ldots, C_{i1k_{i1}})$ is an input-tuple of channels for $S(H_i)$,

......,

$(C_{in_i1}, \ldots, C_{in_ik_{in_i}})$ is an input-tuple of channels for $S(H_i)$, and

$(C_{H_{i1}}, \ldots, C_{H_{im_i}})$ is an output-tuple of channels for $S(H_i)$.

Let $L_i = (S(H_i), q_i)$, where q_i is a tuple of queues connected with all the input channels for $S(H_i)$.

Next, whenever $H_i \rightarrow\!\!> H_j$, the output-tuple of channels for $S(H_i)$ is to be connected with some appropriate input-tuples for $S(H_j)$, via an $FM(NOND(H_j), m_i)$. The appropriate input-tuple of channels is chosen on the condition that P_i is the same as the predicate symbol in the body of a definite clause in $S(H_j)$, which has 'Re-'-atoms corresponding to the input channels for $S(H_j)$.

The total network constructed for L in the way mentioned above is denoted by N(L). Then N(L) is regarded as an NCLP with fair merge operators. Let us define the denotational semantics of channels with L_i corresponding to $S(H_i)$ and their operational behaviour as the same as those of the NCLP--N(L). Since each $S(H_i)$ contains only H_i as a definite clause without 'Se-'-atoms and 'Re-'-atoms, if we assume fair nondeterminism for choosing the positions of queues in each L_i to get messages and provide outputs, then the output-tuple of sequences from each $S(H_i)$ is an asynchronous, continuous function of input-tuples of sequences. FM(p,q) is an extention of an ordinary fair merge operator. Thus N(L) is regarded as the network presented and discussed by Park [8]. This means that the semantics of N(L) is well-defined. This is a theoretical guarantee from the point of semantics for regarding the computation mechanism of logic programs as dataflows with fair merge operators.

It might be concluded that by using the NCLP, we can transform any logic program to a network of dataflows with fair merge operators. This is an interesting and significant aspect of NCLP.

6. CONCLUDING REMARKS

The NCLP is reduced to a dataflow when fair nondeterminism is asked for receiving and sending messages. In such a case, there is a (least) fixpoint semantics of a function associated with the NCLP. As we have seen, this is one of the denotational semantics defined in Section 3. Thus, the denotational semantics of NCLP is well-defined even when it has unbounded nondeterminism.

The operational semantics of NCLP based on its fixpoint semantics was defined in Section 4. The implementation of 'FairChoice' can be done by the fair sequences in W^w .

The expressiveness of the NCLP contains interesting aspects compared with other systems.

It is interesting to extend the NCLP so that each process can rewrite the contents of its queues.

It was briefly mentioned that the NCLP with fair merge operators can express a computation mechanism for sequential logic programs. This is a significant aspect of the proposed network as well as the nodeterministic computing network in [8].

ACKNOWLEDGMENTS

First I would like to thank Professor Park, whose work stimulated the author and inspired the present work during the author's visit to the Department of Computer Science, University of Warwick. The author appreciates the valuable comments of Professor Park concerning this work.

The author is also heartly grateful to Dr. Matthews for his hospitality during the visit.

The author is indebted to Professor Doshita for his encouragement during the author's stay in the United Kingdom.

References

[1] Apt,K.R., and van Emden,M.H., Contributions to the theory of logic programming, J. of ACM, 29, 3 (1982) 841-864.

(21 Clark,K. and Gregory,S., PARLOG: A parallel logic programming language. Research Report DOC 83/5, Dept. Computing, Imperial College (1983).

[3] Clark,K. and Gregory,S., PARLOG: Parallel programming, Research Report DOC 84/4, Dept. Computing, Imperial College (1984).

[4] Kahn,G., The semantics of a simple language for parallel programming, Proc. IFIP 74 (1974) 471-475.

[5] Kahn,G. and McQueen,D., Coroutines and networks of parallel processes, Proc. IFIP 77 (1977) 993-998.

[6] Park,D., On the semantics of fair parallelism, LNCS 86 (l960) 504-526.

[7] Park,D., Concurrency and automata on infinite sequences, LNCS 104 (1981) 167-183.

[8] Park,D., The "fairness" problem and nondeterministic computing networks, in: Foundations of Computer Science IV (de Bakker and van Leeuwen, eds.), Mathematisch Centrum, Amsterdam (1983) 133-161.

[9] Plotkin,G.D., A powerdomain construction, SIAM J. Computing, 5, 3 (1976) 452-487.

[10] Shapiro,E.Y., A subset of Concurrent Prolog and its interpreter, ICOT Technical Report, TR-003 (1983).

[11] Smyth,M.B., Powerdomains, JCSS, 16, 1, (1978) 23-36.

[12] Van Emden,N.H. and Kowalski,R.A., The semantics of predicate logic ae a programming language, J. of ACM, 29, 3 (1982) 841-869.

[13] Wadge,W.W., Away from the operational view of computer science, Theory of Computation Report, No. 32, Dept. Computer Science, University of Warwick (1987).

[14] Madge,W.W., An extensional treatment of dataflow deadlock, LNCS 70 (1979) 285-299.

[15] Yamasaki,S. et al., A new combination of input and unit deductions for Horn sentences, Information processing Letters, 18, 4 (1984) 209-213.

[16] Yamasaki,S., A network of communicating logic programs and its semantics,I, Technical Report Presented at the 106th Seminar of Dept. Information Science, Kyoto University (1986).

[17] Yamasaki,S. et al., A fixpoint semantics of Horn sentences based on substitution sets, to appear in Theoretical Computer Science.