A Mobile Logic Programming Multi-Agent System

Jianwen Chen [1,2]
1. IBM Australia, 20 Berry Street, North Sydney
2. School of Computing & IT, University of Western Sydney, Penrith South DC NSW 1797
Australia
jchen@au1.ibm.com

Yan Zhang
School of Computing & IT
University of Western Sydney
Penrith South DC NSW 1797
Australia
yan@cit.uws.edu.au

Abstract - **In this paper, we present a formalization for a logic programming multi-agent system in mobile environments. Such a system consists of a number of agents connected via wire or wireless communication channels, and we model the interactions between agents in our formalization. Agents communicate with each other by passing answer sets obtained by updating the information received from connected agents with their own private information. Our formalization has following advantages: 1) Our model is knowledge oriented; 2) Our formalization has declarative semantics; 3) Our model can be used to study the details of transaction in mobile environments. Based on this model, knowledge transaction can be studied in such a mobile multi-agent system.**

I. INTRODUCTION

Comparing to stationary environments, mobile environments have a few specific properties such as mobility and disconnection. The communication channels can be wire or wireless in mobile environments. The features of mobile environment have presented new challenges for researchers. We believe that research on multi-agent system and knowledge transaction in mobile environments is critical because this will help us to find a way to significantly improve current development of multi-agent system and mobile system.

However, there seems to be a separation between multi-agent systems and the intelligent agents community on one side, and the mobile system community on the other side [13, 10, 17, 18]. So far no formalization and model has been presented for multi-agent system in mobile environments and no study has been conducted for knowledge transaction in mobile multi-agent system. On mobile system community side, currently the work in paper [4, 5, 12] has introduce calculus to describe the movement of processes and devices in mobile ambient, and the work in [3, 11, 6] has presented a number of Java packages, libraries, and frameworks to implement functionalities for programming distributed and mobile systems. The approaches above have following disadvantage: 1) They are not suitable for knowledge oriented processing in mobile environments; 2) They have no declarative semantics. They are low level algorithms for "how to do" and have no high level "what to do" intelligent functionality. 3) The details of transaction can't be specified. On multi-agent and intelligent agent community side, a lot of framework/model have been developed for problem solving, knowledge representation and reasoning such as stable model/answer set, SMODEL, DLV and XSB model in paper [7, 15, 16]. These models are knowledge oriented with declarative semantics, and their specification language can specified the details of knowledge transaction, but these models are only discussed and limited in stationary environments, and haven't be extended to mobile environments. The motivation of our work is to propose a new framework/model with the following features: 1) It is knowledge based with declarative semantics; 2) It can specify details of knowledge transaction; 3) It can be used in mobile environments.

In this paper we present a formalism and definition for a mobile logic programming multi-agent system (MLPMAS). Such a system is very useful for modeling decision-problems in mobile environments, not just the solutions of the problem but also the evolution of the beliefs of and the interactions between the agents in mobile environments. With respect to previous work, we can characterize major features in our approach as follows: 1) We use extending logic programs and answer set semantics in our formalization so our model is knowledge oriented and has declarative semantics inherited from logic programming; 2) It can specify details of knowledge transaction. By using knowledge rules, input, output and knowledge base itself can be specified; 3) Our model can be used to process knowledge transaction in mobile environment. We give mobile semantics to extending logic programs. Based on this MLPMAS model, knowledge transaction can be studied in such a mobile multi-agent system. Studying knowledge transaction is the aim for us to model MLPMAS systems although we only discuss MLPMAS modeling in this paper.

A mobile logic programming multi-agent system consists of a set of agents in mobile environments. The agent resides on MH, MSS and HS levels respectively, connected through communication channels. The local knowledge base is located on each level of MH, MSS and HS, the agent on each level contains a logic program representing its local information and reasoning method. Agents use information received from their incoming channels as input for their reasoning, where the received information may be overridden by other concerns represented in their programs. The resulting model is communicated to the agent listening to the outgoing channels.

The rest of this paper is organized as follows. In section 2, we introduce our environmental model which combines the characteristics of intelligent agent and mobile environment. In section 3, we introduce relevant concepts and give an overview of extended logic programming. In section 4, we formalize our mobile logic programming multi-agent system (MLPMAS). In section 5, we give a study case to demonstrate how to specify a MLPMAS system in a particular problem domain. Finally, in section 6, we summarize our work.

II. ENVIRONMENTAL MODEL

When we study the transaction processing in mobile environments, we use the three level mobile environment model in the paper [9, 14] to represent the salient features of mobile environments. There is a Home Server (HS) acting as permanent storage of Mobile hosts' (MH) Files. There are Mobile Support Stations (MSS) providing services to a MH when it is within its cell. The MSS is connected to the HS via hardwires. The MH is continuously connected to a MSS via a wireless link while accessing data. It may become disconnected either voluntarily or involuntarily. When a MH registers with a MSS, a proxy is created on its behalf. There is a centralized database residing in the HS. In classical environments, an intelligent agent is an active object with the ability to perceive, reason and act. We assume that an agent has explicitly represented knowledge and a mechanism for operating on or drawing inferences from its knowledge. We also assume that an agent has the ability to communicate. In a distributed computing system, intelligent agent has been introduced to communicate with each other in order to achieve their goals.

Here we propose a new environment model to study knowledge base in mobile environments. This model integrates the features of both mobile environment [13, 10] and intelligent agents [17, 18, 2] as shown in Fig 1.

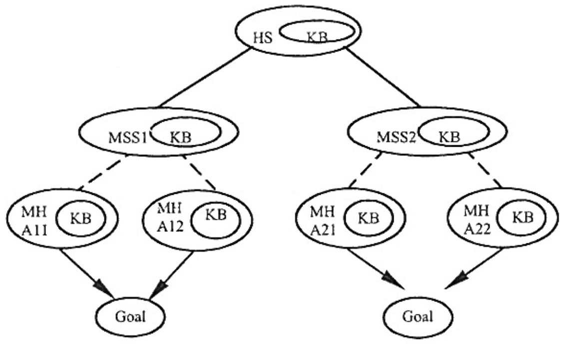

Fig 1. Knowledge Study Environment Model

In this environment model, we assume that every Mobile Host (MH) has its own knowledge base (KB) and intelligent agent (A11, A12, A21, A22), every MSS has knowledge base and agent residing on it as well, MSS1 and MSS2 represent different MSS in different geographic areas. Home Server (HS) level has a knowledge base and an agent that represents a set of rules of knowledge base. Every intelligent agent on MH will work on behalf of MH that resides on, all the agents in the same geographic area will negotiate, communicate, and cooperate with each other to achieve the goal for themselves and their systems. Such as agents can do decision making based on the rules in every knowledge base.

Our mobile logic programming multi-agent systems will be defined and formalized based on this three layer environmental model.

III. EXTENDED LOGIC PROGRAMMING

Logic programming has been proved to be one of the most promising logic based formulations for problem solving, knowledge representation and reasoning. It has great advantages on both declarative semantics (e.g. stable model/answer set semantics [8]) and efficient proving procedures (e.g. SMODEL, DLV and XSB), which make this method be more applicable in the real world problem domains.

In stationary environments, traditional logic programming is used as a knowledge representation tool. An important limitation of this method is that logic programming does not allow us to deal directly with incomplete information, and therefore we only can get either *yes* or *no* answer from a query. Each ground atom is assumed to be false if it does not been certified to be true in the program. When we study knowledge transaction in mobile environments, we should clearly understand that there is a major different between the scenario that the transaction fails and the transaction hangs on due to mobile user's sleep. The first scenario is transaction fails in the sense of its negation succeeds, it is a *no* answer for a query. The second scenario is transaction doesn't succeed because of incomplete information, the answer is *unknown* for a query transaction, but may become a definite answer *yes* or *no* after sometime. Therefore, in mobile environments, we need a method which can deal with incomplete information explicitly, and this method should handle the transaction fails in the sense of its negation succeeds and the transaction that does not succeed in the mobile situation. The extended logic program [2, 8, 1] can overcome such a limitation. It contains classical negation $\neg$ in addition to negation-as-failure *not,* and includes explicit negative information. In the language of extended programs, we can distinguish between a query which fails in the sense that it does not succeed and a query which fails in the stronger sense that its negation succeeds. In mobile semantics, classical negation $\neg$ is defined as explicit negative information, is explicit *no* when transaction is explicit fail. In the situation the mobile host is in voluntary or involuntary sleep, and therefore the information is incompleteness, we say it is absent of atom A, noted by not A, therefore it is unkown.

Generally speacking, an extended logic program is a finite set of rules:

$$L_0 \leftarrow L_1, \dots, L_m, not\ L_{m+1}, \dots, not\ L_n, \qquad \text{(I)}$$

Where $n \geq m \geq 0$, and each L_i is a literal. A literal is a formula of the form A or $\neg A$, where A is an atom. In the body of rule (I), both classical negation $\neg$ and weak negation not are allowed to be presented. The meaning of (I) is as follows: if $L_1, \dots, L_m$ hold, and there is no explicit evidence to show that $L_{m+1}, \dots, L_n$ hold, then L_0 holds.

The answer set of an extended logic program is normally defined as follows [2];Let Π be an extended program without variables that doesn't contain not, and let Lit be the set of ground literals in the language of Π. The answer set of Π is the smallest subset S of Lit such that

(i) for any rule $L_0 \leftarrow L_1, \dots, L_m$ from Π, if $L_1, \dots, L_m \in S$, then $L_0 \in S$;

(ii) if S contains a pair of complementary literals, then S = Lit.

We say Π entails a literal L if L is always true in all answer sets of Π, this is denoted by $\Pi \models L$.

IV. A MOBILE LOGIC PROGRAMMING MULTI-AGENT SYSTEM (MLPMAS)

In this section we formalize and define systems of communicating agents in mobile environments, where each agent is represented by an extended logic program that contains its knowledge about itself and other agents. Agents communicate via communication channels through which the conclusions derived by the agent at the source of the channel are passed on to the agents at the other end.

In MLPMAS systems, we use extended logic programming to represent knowledge base because this method can deal with incomplete information explicitly, it contains classical negation $\neg$, in addition to negation-as-failure *not*. The extended logic programming is a method that can represent the features of knowledge base and knowledge transaction in wireless environments. We define and formalize MLPMAS systems based on three layer environmental model. A model of MLPMAS system is shown in Fig. 2.

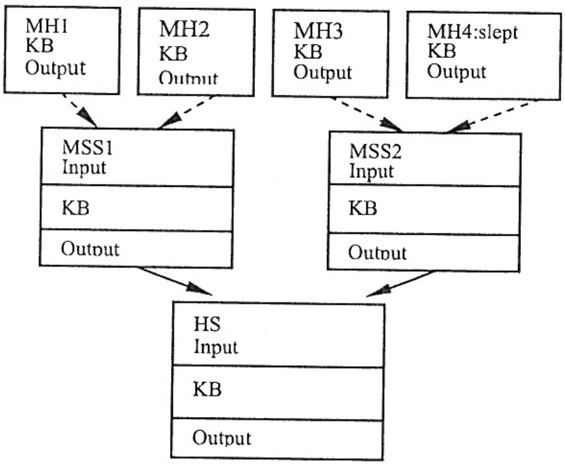

Fig2: A MLPMAS system

A mobile logic programming multi-agent system includes MH, MSS and HS three levels, the local knowledge base is located on each level. The system consists of a set of agents in mobile environments. The agent resides on MH, MSS and HS levels respectively, connected through communication channels. The agent on each level contains its own logic program representing its local information and reasoning method. Agents use information received from their incoming channels as input for their reasoning, where the received information may be overridden by other concerns represented in their programs. Agents produce output to their outgoing communication channels.

Definition 1: *A mobile logic programming multi-agent system*, or *MLPMAS*, is a pair $F = <A, C>$, where A is a set of agents: $A = A_{MH} \cup A_{MSS} \cup A_{HS}$, and $C \subseteq A \times A$ is a reflexive relation representing the communication channels between agents. For any $a_1, a_2 \in A$, if $<a_1, a_2> \in C$, then we say agents a_1 and $a2$ have a *communication channel*. Each agent $a \in A$, there is an associated extended logic programs *LocalKB(a)* which represents agent *a's local knowledge base*.

Now we explain the definition of MLPMAS system above through the following Example 1. In our example, investor agent resides on MH, group agent resides on MSS, and fund manager agent resides on HS. Investor agent manages the local knowledge base and provides output to group agent on behalf of MH. Group agent collects information from investor agents, manages local knowledge base on MSS and sends output to fund manager agent. Fund manager agent collects information from group agents, does the investment decision and manages the local knowledge base on HS. Investor agent, group agent and fund manager agent are represented by a_{MH}, a_{MSS} and a_{HS} respectively.

Example 1:

We have a mobile logic programming multi-agent system $F = <A, C>$, in this MLPMAS system, we have four mobile hosts MH1, MH2, MH3 and MH4, the investor agent resides on each MH:

$$A_{MH} = \{a_{MH1}, a_{MH2}, a_{MH3}, a_{MH4}\}$$

We have two mobile support station MSS1 and MSS2, group agent resides on each MSS:

$$A_{MSS} = \{a_{MSS1}, a_{MSS2}\}$$

We have one home server HS, fund manager agent resides on HS:

$$A_{HS} = \{a_{HS}\}$$

MH1 and MH2 are in geographic location of MSS1, MH3 and MH4 are in geographic location of MSS2. We have wireless communication channel between MH and MSS:

$$<a_{MH1}, a_{MSS1}> \in C,\ <a_{MH2}, a_{MSS1}> \in C,$$
$$<a_{MH3}, a_{MSS2}> \in C,\ <a_{MH4}, a_{MSS2}> \in C$$

We have wire communication channel between MSS and HS:

$$<a_{MSS1}, a_{HS}> \in C, <a_{MSS2}, a_{HS}> \in C$$

We assume there is communication channel between MH and HS as well:

$$<a_{MH1}, a_{HS}> \in C,\ <a_{MH2}, a_{HS}> \in C,$$
$$<a_{MH3}, a_{HS}> \in C,\ <a_{MH4}, a_{HS}> \in C$$

As we mentioned earlier, each agent is associated with an extended logic program of its local knowledge base. If there is no answer set, it means local knowledge base is not well designed. If there are multiple answer sets, each answer set represents a possible knowledge state.

We define input and output of agents in MLPMAS systems as follows.

Definition 2. Let $F = <A, C>$ be a *MLPMAS*, where $A = A_{MH} \cup A_{MSS} \cup A_{HS}$. At MH, MSS or HS level, for $\forall a \in A$, we have two parts of inputs: *message input* and *knowledge input*, denoted by $MessageInput(a, X)$ and $KnowledgeInput(a, Y)$ respectively. That is,

$$Input(a) = < MessageInput(a, X), KnowledgeInput(a, Y) >$$

here $X \subseteq A, Y \subseteq A$, X, Y are subsets of A. Agent a collects message input from agents in X, and collects knowledge input from agents in Y, where

for $\forall b \in X$, we have $<a, b> \in C$, or $<b, a> \in C$ and

for $\forall b' \in Y$, we have $<a, b'> \in C$, or $<b', a> \in C$

i.e. we know there is a communication channel between agent a and agent b, and agent a and agent b' respectively. Message input is the information that an agent sends to another agent for the communication purpose. Such as one agents informs another agent that it will move into another MSS geographic area. This information will not cause any influence to the other agent's local knowledge base. While knowledge input is the information produced by the other agent's local knowledge base, and will be taken into the agent's local knowledge base, i.e. the answer set of a logic program.

For $\forall a \in A$, we have two parts of output, *message output* and *knowledge output*, denoted by $MessageOutput(a,X)$ and $KnowledgeOutput(a,Y)$ respectively. That is,

$$Output(a) = < MessageOutput(a,X), KnowledgeOutput(a,Y) >$$

here $X \subseteq A, Y \subseteq A$, agent a sends message output to agents in X, and sends knowledge output to agents in Y.

Message output is information output for communication purpose, this information will not cause any influence to the other agent's local knowledge base. Knowledge output is the information that produced by the agent's local knowledge base and will have impact for the other agent's knowledge base.

Definition 3: We define *knowledge input and output* in MLPMAS systems on MH level as follows.

There is no input for MHs at MH level because this is the first level in MLPMAS systems,i.e.

$$KnowledgeInput(a_{MH}, Y) = \phi \quad (1)$$

The knowledge output can be derived from the equation:

$$KnowledgeOutput(a_{MH}, a_{MSS})$$
$$= \text{an answer set of}$$
$$[LocalKB(a_{MH}) \cup KnowledgeInput(a_{MH}, Y)] \quad (2)$$

i.e. knowledge output is an answer set of the program formed by the local logic program of agent a_{MH} with extending of knowledge input from Y for agent a_{MH}. $LocalKB(a)$ is an extended logic program as we defined in Definition 1, $KnowledgeInput(a,Y)$ is a set of facts (beliefs). Note that $LocalKB(a_{MH}) \cup KnowledgeInput(a_{MH}, Y)$ is viewed as a new logic program while fact $e \in KnowledgeInput(a,Y)$ is treated as a rule $e \leftarrow$.

Definition 4: We define *knowledge input and output* in MLPMAS systems on MSS level as follows.

The knowledge input can be derived from the equation:

$$KnowledgeInput(a_{MSS}, Y)$$
$$= cons(\bigcup_{a_{MH} \in Y} KnowledgeOutput(a_{MH}, a_{MSS}), S_F) \quad (3)$$

where *cons(X)* represents the maximal consistent subnet. So the knowledge input of a_{MSS} is the maximal consistent subset of knowledge output from Y to agent a_{MSS} with respect to the select function S_F.

S_F is the *selection function* of system *F*. For knowledge output, $\bigcup_{b \in Y} knowledgeOutput(b,a)$ may be inconsistent, S_F is introduced to solve such inconsistency by taking proper preference in the domain. Note that S_F is domain dependent, it can be a special logic programming rule for specific problem domain.

The knowledge output can be derived from the equation:

$$KnowledgeOutput(a_{MSS}, a_{HS})$$
$$= \text{an answer set of}$$
$$[LocalKB(a_{MSS}) \cup KnowledgeInput(a_{MSS}, Y)] \quad (4)$$

i.e. knowledge output is an answer set of the program formed by the local logic program of agent a_{MSS} with extending of knowledge input of agent a_{MSS}.

Definition 5: *knowledge input and output* in MLPMAS systems on HS level as follows.

The knowledge input can be derived from the equation:

$$KnowledgeInput(a_{HS}, Y)$$
$$= cons(\bigcup_{a_{MSS} \in Y} KnowledgeOutput(a_{MSS}, a_{HS}), S_F) \quad (5)$$

i.e. knowledge input of a_{HS} is the maximal consistent subset of knowledge output from Y to agent a_{HS} with respect to the select function S_F.

The knowledge output can be derived from the equation:

$$KnowledgeOutput(a_{HS})$$
$$= \text{an answer set of}$$
$$[LocalKB(a_{HS}) \cup KnowledgeInput(a_{HS}, Y)] \quad (6)$$

i.e. knowledge output is an answer set of the program formed by the local logic program of agent a_{HS} with extending of knowledge input of agent a_{HS}.

V. A CASE STUDY FOR MLPMAS SYSTEM IN MOBILE ENVIRONMENTS

We will go through a study case in this section to specify a MLPMAS system according to the formalization in section 4. The model of MLPMAS systems has been shown in Fig. 2 in section 4.

Our study case is in a specific investment problem domain. As showed in Fig. 2, at MH level, we have MH1, MH2, MH3 and MH4. MH1 and MH2 are in the cell of MSS1, MH3 and MH4 are in the cell of MSS2. MSS1 and MSS2 are connected to the same HS. At MH level, each MH has a local knowledge base that includes a set of investment rules, investor agent resides on it. At MSS level, MSS has own knowledge base, MSS accepts the input from MHs and produces the output based on the input and own belief. The HS accepts the input from MSS level, it has own local knowledge base, investment decision will be made on HS level.

For the initial status, we assume MH1 and MH2 are all alive when transaction is processed in MSS1 cell. In MSS2 cell, the MH3 is alive, while MH4 is slept at the moment HS is requesting the transaction information from all related MH agents. The HS will need information from MH4 when the time it does the decision making.

<u>MH Level:</u>

On MH level, there is no input for the agent on MH. According to equation (2), we have

$$KnowledgeOutput(a_{MH}, a_{MSS})$$
$$= \text{an answer set of}$$
$$[LocalKB(a_{MH}) \cup KnowledgeInput(a_{MH}, Y)]$$
$$= \text{an answer set of } [LocalKB(a_{MH})]$$

i.e. on MH level, the knowledge output is an answer set of local knowledge base. Based the local knowledge base on MHs, the knowledge outputs are derived as below on MH1, MH2, MH3 and MH4.

$$KnowledgeOutput(a_{MH1}, a_{MSS1}) =$$
$$\{profit(share1), risk(share1), \neg cost(share1)\}$$

i.e. it is high profit, high risk and low cost to invest share1 on MH1.

$KnowledgeOutput(a_{MH2}, a_{MSS1}) =$

$\{profit(share1), \neg risk(share1), \neg cost(share1)\}$

i.e. it is high profit, low risk and low cost to invest share1 on MH2.

$KnowledgeOutput(a_{MH3}, a_{MSS2}) =$

$\{profit(share1), \neg risk(share1), \neg cost(share1)\}$

i.e. it is high profit, low risk and low cost to invest share1 on MH3.

The MH4 is slept at the moment the information is retrieved from it.

MSS level:

On MSS level, according to Definition 2 in MLPMAS systems, input of MSS1agent equals:

$$Input(a_{MSS1}) =< MessageInput(a_{MSS1}, X), KnowledgeInput(a_{MSS1}, Y) >$$

where X= $\{a_{MH1}, a_{MH2}\}$ is a subset of A, including all investor agents who have message input for group agent on MSS1. Y= $\{a_{MH1}, a_{MH2}\}$ is a subset of A, including all investor agents who have knowledge input for group agent on MSS1.

According to equation (3), we have knowledge input on MSS1 as below:

$$KnowledgeInput(a_{MSS_1}, Y)$$

$$= cons(\bigcup_{a_{MH} \in Y} KnowledgeOutput(a_{MH}, a_{MSS_1}), S_F)$$
$$= cons(KnowledgeOutput(a_{MH1}, a_{MSS1}) \cup KnowledgeOutput(a_{MH2}, a_{MSS1}), S_F)$$

For agent a_{MSS1}, $risk(share1)$ is a belief in output of a_{MH1}, while $\neg risk(share1)$ is a belief in output of a_{MH2}, they are inconsistent. Here we assume selection function S_F takes positive atom as higher preference for investment risk, therefore $risk(share1)$ will become the input of a_{MSS1}.

We have knowledge input as below:

$$KnowledgeInput(a_{MSS1}, Y)$$

$$= \{profit(share1), risk(share1), \neg cost(share1)\}$$

We can see that different knowledge input is derived with considering selection function in specific problem domain, therefore different answer set is derived for decision making due to selection function.

Input of MSS2 agent equals:

$$Input(a_{MSS2}) =< MessageInput(a_{MSS2}, X), KnowledgeInput(a_{MSS2}, Y) >$$

where X= $\{a_{MH3}, a_{MH4}\}$ is a subset of A, including all investor agents who have message input for group agent on MSS2. Y= $\{a_{MH3}, a_{MH4}\}$ is a subset of A, including all investor agents who have knowledge input for group agent on MSS2.

The knowledge input of MSS2 agent equals:

$$KnowledgeInput(a_{MSS2}, Y)$$

$$= cons(\bigcup_{a_{MH} \in Y} KnowledgeOutput(a_{MH}, a_{MSS_2}), S_F)$$
$$= cons(KnowledgeOutput(a_{MH3}, a_{MSS2}), S_F)$$

$$= \{profit(share1), \neg risk(share1), \neg cost(share1)\}$$

On MSS1, we have rule r1 related to this investment in its knowledge base

$$\left\{\begin{array}{l} r1: holds(info-requested(HS, MHi), s) \\ \leftarrow holds(slept(MHi), s) \end{array}\right\}$$

On MSS2, we have rule r2 related to this investment in its knowledge base

$$\left\{\begin{array}{l} r2: holds(info-requested(HS, MHi), s) \\ \leftarrow holds(slept(MHi), s) \end{array}\right\}$$

The r1 and r2 denote that if MHi is slept at the time the HS agent requests transaction information from MHs, HS will request information from MHi when HS does the decision making for the transaction.

Based on Definition 2, there are two parts for the output of MSS, message output and knowledge output. According to the equation (4), the knowledge output can be derived:

$KnowledgeOutput(a_{MSS}, a_{HS})$

=an answer set of

$[LocalKB(a_{MSS}) \cup KnowledgeInput(a_{MSS}, Y)]$

Thus, the knowledge output of MSS1 is derived as below:

$$KnowledgeOutput(a_{MSS1}, a_{HS})$$

$$= \{profit(share1), risk(share1), \neg cost(share1)\}$$

The knowledge output of MSS2 is derived as below:

$$KnowledgeOutput(a_{MSS2}, a_{HS})$$

$$= \left\{\begin{array}{l} profit(share1), \neg risk(share1), \neg cost(share1), \\ info-requested(HS, MH4) \end{array}\right\}$$

A new belief *info-requested(HS, MH4)* is added to the answer set on MSS2 because of rule r2 in its local knowledge base.

HS level:

On HS level, based on Definition 2 in MLPMAS systems, input of HS agent equals:

$$Input(a_{HS}) =< MessageInput(a_{HS}, X), KnowledgeInput(a_{HS}, Y) >$$

X= $\{a_{MSS1}, a_{MSS2}\}$ is a subset of A including all involved agents who have message information input for agent on HS. Y= $\{a_{MSS1}, a_{MSS2}\}$ is a subset of A including all involved agents who have knowledge input for agent on HS.

Based on the equation (5), knowledge input of HS agent equals:

$$KnowledgeInput(a_{HS}, Y)$$

$$= cons(\bigcup_{a_{MSS} \in Y} KnowledgeOutput(a_{MSS}, a_{HS}), S_F)$$
$$= cons(KnowledgeOutput(a_{MSS1}, a_{HS}) \cup KnowledgeOutput(a_{MSS2}, a_{HS}), S_F)$$

$$= \left\{\begin{array}{l} profit(share1), risk(share1), \neg cost(share1), \\ info-requested(HS, MH4) \end{array}\right\}$$

$risk(share1)$ is a belief of input on HS with considering the selection function.

We have rules r3-r9 in local knowledge base of HS.

$$\left\{\begin{array}{l} r3: holds(invest(share1),s) \leftarrow holds(profit(share1),s), \\ \neg holds(risk(share1),s), \neg holds(cost(share1),s), \\ holds(info-get(MHi), res(request-info(MHi),s)) \\ r4: \neg holds(invest(share1),s) \leftarrow holds(risk(share1),s) \\ r5: \neg holds(invest(share1),s) \leftarrow holds(cost(share1),s) \\ r6: \neg holds(risk(share1),s) \leftarrow notholds(risk(share1),s) \\ r7: \neg holds(cost(share1),s) \leftarrow notholds(cost(share1),s) \\ r8: \neg holds(invest(share1),s) \leftarrow \\ \neg holds(info-get(MHi), res(request-info(MHi),s)), \\ holds(info-requested(HS, MHi),s) \\ r9: \neg holds(info-get(MHi),s) \leftarrow \\ notholds(info-get(MHi),s), \\ holds(info-requested(MHi),s), holds(timeout(MHi),s) \end{array}\right\}$$

The r3 denotes if it is high profit, low risk, low cost to invest share1 and HS gets requested information from ever slept MHi, HS will do the decision to invest share1. The r4, r5 and r8 denote if share1 is high risk or high cost on any MHi, or can't get information from ever slept MHi, HS will make the decision that share1 won't be invested. The r6 and r7 denote that if share1 hasn't be specified to be high risk or high cost for any MHi, then it is considered to be low risk or low cost. The r9 denotes that if HS hasn't got requested information from slept MHi until time is out, then HS will assume no information is available from MHi.

Based on Definition 2, the output of HS agent has two parts, message output and knowledge output. The knowledge output is derived based on the equation (6):

$$KnowledgeOutput(a_{HS})$$
$$= \text{an answer set of}$$
$$[LocalKB(a_{HS}) \cup KnowledgeInput(a_{HS}, Y)]$$

$\{\neg invest(share1)\}$ is in every answer set of $[LocalKB(a_{HS}) \cup KnowledgeInput(a_{HS}, Y)]$, thus we know that $[LocalKB(a_{HS}) \cup KnowledgeInput(a_{HS}, Y)] \models (entails)$ $\{\neg invest(share1)\}$. $risk(share1)$ is a part of knowledge input of HS, according to the rule r4, we know share1 can't be invested if it is high risk to invest it. From output of HS agent on HS makes the decision that share1 won't be invested. In this case, no matter what information is got from MH4, HS will do the decision that share1 can't be invested, otherwise the information from MH4 will have impact for decision making. We also show from this example that selection function will impact the decision making. If no selection function in this example, $risk(share1)$ will not be part of input of HS agent, therefore share1 may be invested.

After HS has made decision that share1 will not be invested. The transaction decision will be sent to MSS, and all involved MHs will be noticed by broadcasting of MSS.

VI. CONCLUSION

In this paper, we have presented and formalized a logic programming multi-agent system (MLPMAS) in mobile environments. This kind of system consists of a number of agents connected via wire or wireless communication channels, we have modeled the interactions between agents in our formalization. Our formalization is knowledge oriented with declarative semantics, and it can be used to study the details of knowledge transaction in mobile environments.

VII. REFERENCES

[1] Baral, C., *Knowledge Representation, Reasoning and Declarative Problem Solving*, Cambridge University Press, 2003.

[2] Baral, C., and Gelfond, "M., Logic Programming and Knowledge Representation," *Logic Programming*, 1994, pp. 73-148.

[3] Bettini, L., et al, "KLAVA: a Java package for distributed and mobile applications," *Software Practice and Experience,* 32 (2002) 1365-1394.

[4] Cardelli, L., A. Gordon, D., "Mobile Ambients," *Theoretical Computer Science*, 240 (2000), 177-213.

[5] Cardelli, L., A. Gordon, D., "Types for the Ambient Calculus," *Information and Computation* 177 (2002), 160-194.

[6] Deugo D.,"Choosing a mobile agent messaging model," *Proceedings of ISADS 2001.* IEEE Press, 2001;278-286.

[7] Eiter, T., and et al., "A deductive system for nonmonotonic reasoning," *in Proceedings of the 4th International Conference on Logic Programming and Nonmonotonic Reasoning (LPNMR97)*, pp 363-374. LNAI, Vol. 1265, 1997.

[8] Gelfond, M., and Lifschitz, V., "Classical Negation in Logic Programs and Disjunctive Databases," *New Generation Computing*, 1991, pp. 365-385.

[9] Imielinski, T., and Korth, H.F., *Mobile computing*, Kluwer Academic Publishers, 1996.

[10] Komiya, T., et al, "Mobile Agent Model for Transaction Processing on Distributed Objects," *Information Sciences*, 2003, pp.1-16.

[11] Lange D, Oshima M. *Programming and Deploying Java Mobile Agents with Aglets,* Addision-Wesley: Reading, MA, 1998.

[12] Milner, R., et al, "A calculus of mobile processes, Parts 1-2," *Information and Computation*, 100 (1) (1992) 1-77.

[13] Milojicic, D., "Mobile Agent Applications," *IEEE Concurrency*, 1999, pp. 80-90.

[14] Mirghafori, N., and Fontaine, A., "A Design for File Access in a Mobile Environment," *Proceedings of the IEEE - Conference on Mobile Computing*, 1995, pp. 57-61.

[15] Nemela, I., and Simons, P., "Efficient implementation of the well-founded and stable model semantics," *in Proceeding of the International Joint Conference and Symposium on Logic Programming,* pp 289-303. MIT Press, 1996.

[16] Vos, M. D., and Vermeir, D., "Extending Answer Sets for Logic Programming Agents," *in Proceedings of the Logic in Artificial Intelligence (Jelia2000) workshop*, 2000.

[17] Weiss, G., *Multiagent Systems: A Modern Approach to Distributed Artificial Intelligence*, The MIT Press, Cambridge, Massachusetts, 1999.

[18] Wooldridge, M., *An Introduction to Multiagent Systems*, John Wiley & Sons, LTD, 2002.