Eliminating Opportunism using an Epistemic Mechanism

Jieting Luo*  
Centrum Wiskunde & Informatica  
Amsterdam, the Netherlands  
jieting.luo@cwi.nl

Max Knobbout  
Triple  
Alkmaar, the Netherlands  
m.knobbout@wearetriple.com

John-Jules Meyer  
Utrecht University  
Utrecht, the Netherlands  
J.J.C.Meyer@uu.nl

ABSTRACT

Opportunism is a behavior that takes advantage of knowledge asymmetry and results in promoting agents’ own value and demoting other agents’ value. It is important to eliminate such a selfish behavior in multi-agent systems, as it has undesirable results for the participating agents. However, as the context we study here is multi-agent systems, system designers actually might not be aware of the value system for each agent thus they have no idea whether an agent will perform opportunistic behavior. Given this fact, this paper designs an epistemic mechanism to eliminate opportunism given a set of possible value systems for the participating agents: an agent’s knowledge gets updated so that the other agent is not able to perform opportunistic behavior, and there exists a balance between eliminating opportunism and respecting agents’ privacy.

KEYWORDS

Opportunism; Mechanism Design; Logic; Multi-agent Systems

1 INTRODUCTION

Consider a scenario: A seller sells a cup to a buyer and it is known by the seller beforehand that the cup is actually broken. The buyer buys the cup without knowing it is broken. The seller exploits the knowledge asymmetry about the transaction to achieve his own gain at the expense of the buyer. Such behavior which is intentionally performed by the seller was named opportunistic behavior (or opportunism) by economist Williamson [15]. Opportunistic behavior is a behavior that takes advantage of relevant knowledge asymmetry and results in promoting an agent’s own value and demoting another agent’s value. On the one hand, it is common in distributed multi-agent systems that agents possess different knowledge, which enables the performance of opportunism; on the other hand, opportunistic behavior has undesirable results for other agents who participate in the system. Thus, we want to design mechanisms to eliminate opportunism. This paper investigates an epistemic mechanism, which allows us to eliminate the performance of opportunism in the system by revealing updates. In papers [6] [7], opportunism is monitored and predicted given a value system for an agent, i.e., an agent performed and will perform opportunistic behavior if he has the value system as we assume. However, as the context we study here is multi-agent systems, system designers might not be aware of the value system for each agent before designing any mechanism to eliminate opportunism in the system. The goal of this paper is thus to design mechanisms to eliminate opportunism given a set of possible value systems of agents, which contains the value systems with opportunistic propensity.

In mechanism design, a mechanism is an institute, procedure, or game for determining outcomes [8] [11]. Differently, we in this paper consider an operation to the system as an indirect mechanism: a revealing update that can eliminate opportunism through updating the knowledge of the agent. More precisely, we remove the precondition of opportunism (knowledge asymmetry) by revealing knowledge to agents such that agents will not be able to perform opportunistic behavior. Since agents’ value systems are unknown to the system designer, there might exist privacy norms that prevent agents from having the knowledge for eliminating opportunism. We prove formal properties that allow us to check whether we can eliminate opportunism and respect agents’ privacy as well.

2 FRAMEWORK

In this section, we introduce the model we use for multi-agent systems as in [7]. A transition system consists of agents, states of the world, actions, agents’ epistemic accessibility relations, transitions which go from one state to another by an action, and a valuation function that returns for each state the properties of the environment.

Definition 2.1. Let \( \Phi = \{p, q, \ldots \} \) be a finite set of atomic propositional variables. A transition system over \( \Phi \) is a tuple

\[
T = (Agt, S, Act, \pi, \mathcal{K}, \mathcal{R}, s_0)
\]

where

- \( \text{Agt} = \{1, \ldots, n\} \) is a finite set of agents;
- \( S \) is a finite set of states;
- \( \text{Act} \) is a finite set of actions;
- \( \pi: S \to 2^\Phi \) is a valuation function mapping a state to a set of propositions that are considered to hold in that state;
- \( \mathcal{K}: \text{Agt} \to 2^{S \times S} \) is a function mapping an agent in \( \text{Agt} \) to a reflexive, transitive and symmetric binary relation between states; that is, given an agent \( i \), for all \( s \in S \) we have \( s\mathcal{K}(i)s \); for all \( s, t, u \in S \) \( s\mathcal{K}(i)t \) and \( t\mathcal{K}(i)u \) imply that \( s\mathcal{K}(i)u \); and for all \( s, t \in S \) \( s\mathcal{K}(i)t \) implies \( t\mathcal{K}(i)s \); \( s\mathcal{K}(i)s \) is interpreted as state \( s' \) is epistemically accessible from state \( s \) for agent \( i \); we also use \( \mathcal{K}(i,s) = \{ s' \mid s\mathcal{K}(i)s' \} \) to denote the set of agent \( i \)'s epistemically accessible states from state \( s \);
In the interest of simplicity, we only consider one action that takes place at a transition, thus the model is not concurrent.

Now we define the language we use. The language $\mathcal{L}_{\text{KA}}$, propositional logic extended with knowledge and action modalities, is generated by the following grammar:

$$\varphi ::= p \mid \neg \varphi \mid \varphi_1 \lor \varphi_2 \mid K_i \varphi \lor \langle a \rangle \varphi \quad (i \in \text{Agt}, a \in \text{Act})$$

The semantics of $\mathcal{L}_{\text{KA}}$ are defined with respect to the satisfaction relation $\models$. Given a transition system $\mathcal{T}$ and a state $s$ in $\mathcal{T}$, a formula $\varphi$ of the language can be evaluated as follows:

- $\mathcal{T}, s \models p$ if and only if $p \in \pi(s)$;
- $\mathcal{T}, s \models \neg \varphi$ if and only if $\mathcal{T}, s \not\models \varphi$;
- $\mathcal{T}, s \models \varphi_1 \lor \varphi_2$ if and only if $\mathcal{T}, s \models \varphi_1$ or $\mathcal{T}, s \models \varphi_2$;
- $\mathcal{T}, s \models K_i \varphi$ if and only if for all $t$ such that $s \mathcal{R} (t)$, $\mathcal{T}, t \models \varphi$;
- $\mathcal{T}, s \models \langle a \rangle \varphi$ if and only if there exists $s'$ such that $(s, a, s') \in \mathcal{R}$ and $\mathcal{T}, s' \models \varphi$.

Other classical logical connectives (e.g., "\&", "\rightarrow") are assumed to be defined as abbreviations by using $\neg$ and $\lor$ in the conventional manner. As standard, we write $\mathcal{T} \models \varphi$ if $\mathcal{T}, s \models \varphi$ for all $s \in S$, and $\models \varphi$ if $\mathcal{T} \models \varphi$ for all multi-agent systems $\mathcal{T}$. Notice that we can also interpret $\langle a \rangle \varphi$ as the ability to achieve $\varphi$ by action $a$. In addition, the $\mathcal{K}$-relation being S5, we also place restrictions of no-forgetting and no-learning based on Moore’s work [9] [10] to simplify our model. It is specified as follows: given a state $s$ in $S$, if there exists $s'$ such that $s \mathcal{R} (a) \mathcal{K} (i) (s')$ holds, then there is a $s''$ such that $s \mathcal{R} (i) (s'')$ and $s'' \mathcal{R} (a) (s')$ hold; if there exists $s'$ and $s''$ such that $s \mathcal{R} (i) (s')$ and $s'' \mathcal{R} (a) (s')$ hold, then $s (a) \mathcal{K} (i) (s'')$. Following this restriction, we have

$$\models K_i ((a) \varphi) \iff \langle a \rangle K_i \varphi.$$

In other words, if an agent has knowledge about the effect of an action, he will not forget about it after performing the action; and the agent will not gain extra knowledge about the effect of an action after performing the action.

One important feature of opportunism is that it promotes agents’ own value but demotes others’ value. In this section we will specify agents’ value system, as it is the standard of agents’ consideration about the performance of opportunistic behavior. A value can be seen as an abstract standard according to which agents have their preferences over states. For instance, if we have a value denoting equality, we prefer the states where equal sharing or equal rewarding hold. Related work about values can be found in [12] and [13]. Because of the abstract feature of a value, it is usually interpreted in more detail as a state property, which is represented as a $L_{\text{KA}}$ formula. The most basic value we can construct is simply a proposition $p$, which represents the value of achieving $p$. More complex values can be interpreted such as $K_p$, meaning that it is valuable to achieve knowledge. More examples can be found in [7].

We argue that agents can always compare any two values, as we can combine two equivalent values as one value. Thus, we then define a value system as a total order (representing the degree of importance) over a set of values, which means that agents can always compare any two values. In other words, every element in the set of values is comparable to each other and none of them is logically equivalent to each other. One can see similar approaches in [3] for the definition of preferences and in [1] for the definition of goals.

**Definition 2.2 (Value System).** A value system $V = (\text{Val}, \prec)$ is a tuple consisting of a finite set $\text{Val} = \{v, \ldots, v'\} \subseteq L_{\text{KA}}$ of values together with a strict total ordering $\prec$ over Val. When $v \prec v'$, we say that value $v'$ is more important than value $v$ as interpreted by value system $V$. A value system profile $(V_1, V_2, \ldots, V_{\text{Ag}})$ is a tuple containing a value system $V_i$ for each agent $i$.

We also use a natural number indexing notation to extract the value of a value system, so if we have the ordering $v < v' < \ldots$ for a value system $V$, then $V[0] = v$, $V[1] = v'$, and so on. Note that different agents may or may not have different value systems. We now define a multi-agent system as a transition system together with agents’ value systems. Formally, a multi-agent system $\mathcal{M}$ is an $(n + 1)$-tuple:

$$\mathcal{M} = (\mathcal{T}, V_1, \ldots, V_n),$$

where $\mathcal{T}$ is a transition system, and for each agent $i$ in $\mathcal{T}$, $V_i$ is a value system.

We now define agents’ preferences over two states in terms of values, which will be used for modeling agents’ decision making and the effect of opportunism. We first define how a value gets promoted and demoted along a state transition:

**Definition 2.3 (Value Promotion and Demotion).** Given a value $v$ and an action $a$, we define the following shorthand formulas:

- $\text{promoted}(v, a) := \neg v \land (a)v$
- $\text{demoted}(v, a) := v \land (a)\neg v$.

We say that a value $v$ is promoted along the state transition $(s, a, s')$ if and only if $s \models \text{promoted}(v, a)$, and we say that $v$ is demoted along this transition if and only if $s \models \text{demoted}(v, a)$.

An agent’s value $v$ gets promoted along the state transition $(s, a, s')$ if and only if $v$ doesn’t hold in state $s$ and holds in state $s'$; an agent’s value $v$ gets demoted along the state transition $(s, a, s')$ if and only if $v$ holds in state $s$ and doesn’t hold in state $s'$. We secondly define a function $\text{Mpreferred}(i, s, s')$ that maps a value system and two different states to the most preferred value that changes when going from state $s$ to $s'$ for agent $i$. In other words, it returns the value that the agent most cares about, i.e., the most important change between these states for the agent, and all the values that are more important than that value remain unchanged from state $s$ to state $s'$.
We write $s_{M} \preceq s_{M}'$ for short if $M$ is clear from context.

For example, given agent $i$’s value system $u \prec v \prec w$, if formula $u \prec v$ and $\neg w$ hold in state $s$ and formula $u, v$, and $\neg w$ hold in state $s'$, function $M_{\text{preferred}}(i, s, s')$ will return $v$. This is because value $w$ remains the same in both states and value $v$ changes from $\neg v$ to $v$. With this function we can define agents’ preferences over two states. We use a binary relation $\succeq$ over states to represent agents’ preferences.

Definition 2.5 (State Preferences). Given a multi-agent system $\mathcal{M}$, an agent $i$ and two states $s$ and $s'$, agent $i$ weakly prefers state $s'$ to state $s$, denoted as $s \succeq_{i} M s'$, iff

$$M, s \models M_{\text{preferred}}(i, s, s').$$

We write $s \succeq_{i} s'$ for short if $M$ is clear from context. As standard, we also define $s \prec_{i} s'$ to mean $s \succeq_{i} s'$ and $s' \not\succeq_{i} s$, and $s \prec_{i} s'$ to mean $s \succeq_{i} s'$ and $s \preceq_{i} s'$. Moreover, we write $s \succ_{i} s'$ for sets of states $S$ and $S'$ whenever $\forall s \in S, \forall s' \in S': s \succ_{i} s'$.

The intuitive meaning is that agent $i$ weakly prefers state $s'$ to $s$ if and only if the agent’s most preferred value does not get demoted (either stays the same or gets promoted). Using the same example for the illustration of function $M_{\text{preferred}}$, given agent $i$’s value system $u \prec v \prec w$, if formula $u, v, \neg w$ hold in state $s$ and formula $u, v, w$ hold in state $s'$, the agent prefers state $s'$ to state $s$. Clearly there is a correspondence between state preferences and value promotion or demotion by an action: given a model $\mathcal{M}$ with agent $i$, state $s$ and available action $a$ in $s$, and let $v^* = M_{\text{preferred}}(i, s, s(a))$,

$$s <_{i} s(a) \Leftrightarrow M, s \models M_{\text{promoted}}(v^*, a),$$

$$s >_{i} s(a) \Leftrightarrow M, s \models M_{\text{promoted}}(v^*, a),$$

$$s \sim_{i} s(a) \Leftrightarrow M, s \models \neg(M_{\text{promoted}}(v^*, a) \lor M_{\text{promoted}}(v^*, a)).$$

One can refer to [7] for the proof. Moreover, the $\succ_{i}$ relation is reflexive, transitive and total, which have been proved in [7]. It is possible that agents have different preferences over states, since they might not share the same value system.

Since we have already defined values and value systems as agents’ basis for decision making, we can start to apply decision theory to reason about agents’ decision-making. Given a state in the system, there are several actions available to an agent, and he has to choose one in order to go to the next state. Before choosing an action to perform, an agent must think about which actions are available to him. We have already seen that, for a given state $s$, the set of available actions is $\mathcal{A}(s)$. However, since an agent only has partial knowledge about the state, we argue that the actions that an agent knows to be available is only part of the actions that are physically available to him in a state. For example, an agent can call a person if he knows the phone number of the person; without this knowledge, he is not able to do it, even though he is holding a phone. Recall that

Definition 2.4 (Most Preferred Value). Given a multi-agent system $\mathcal{M}$, an agent $i$ and two states $s$ and $s'$, function $M_{\text{preferred}}: \mathcal{A}(s) \times S \times S \rightarrow \text{Val}$ is defined as follows:

$$M_{\text{preferred}}(i, s, s')_{\mathcal{M}} := v_i[\min\{j \mid \forall k > j : M, s \models v_i[k] \Leftrightarrow M, s' \models v_i[k]\}].$$

We write $M_{\text{preferred}}(i, s, s')$ for short if $M$ is clear from context.

Because a stuttering action $sta$ is always included in $\mathcal{A}(s)$ for any state $s$, we have that $sta \in \mathcal{A}(i, s)$ for any agent $i$. When only $sta$ is in $\mathcal{A}(i, s)$, we say that the agent cannot do anything because of his limited knowledge. Obviously an agent’s subjectively available actions are always part of his physically available actions ($\mathcal{A}(i, s) \subseteq \mathcal{A}(s)$). Based on agents’ rationality assumptions, he will choose an action with his partial knowledge of the current state and the next state. Given a state $s$ and an action $a$, an agent considers the next possible states as the set $\mathcal{K}(i, s(a))$. For another action $a'$, the set of possible states is $\mathcal{K}(i, s(a'))$. The question now becomes: How do we compare these two possible set of states? Clearly, when we have $\mathcal{K}(i, s(a)) <_{i} \mathcal{K}(i, s(a'))$, meaning that all alternatives of performing action $a'$ are more desirable than all alternatives of choosing action $a$, it is always better to choose action $a'$. However, in some cases it might be that some alternatives of action $a$ are better than some alternatives of action $a'$ and vice-versa. In this case, an agent cannot decisively conclude which of the actions is optimal, which implies that the preferences over actions (namely sets of states) is not total. This leads us to the following definition:

Definition 2.7 (Rational Alternatives). Given a state $s$, an agent $i$ and two actions $a, a' \in \mathcal{A}(i, s)$, we say that action $a$ is dominated by action $a'$ for agent $i$ in state $s$ iff $\mathcal{K}(i, s(a)) <_{i} \mathcal{K}(i, s(a'))$. The set of rational alternatives for agent $i$ in state $s$ is given by the function $a^*_i: S \rightarrow 2^{\mathcal{A}(i, s)}$, which is defined as follows:

$$a^*_i(s) = \{a \in \mathcal{A}(i, s) \mid \neg\exists a' \in \mathcal{A}(i, s) : a \neq a' \text{ and } a' \text{ dominates } a \text{ for agent } i \text{ in state } s\}.$$
An agent will perform opportunistic behavior when he has the state preferences: $\text{Ac}(i, s) = \text{Ac}(i, s) \land \text{Ac}(i', s')$, agent $i$ knows that only $sta$, $a_1$ and $a_2$ are available to him in state $s$.

Next we talk about agent $i$'s rational alternatives in state $s$. Given agent $i$'s value system $V_i = \{u < v < w\}$, and the following valuation: $u$, $\neg v$ and $\neg w$ hold in $K(i, s)$, $\neg u$, $\neg v$ and $w$ hold in $K(i, s(a_1))$, and $u$, $v$ and $\neg w$ hold in $K(i, s(a_2))$, we then have the following state preferences: $K(i, s) < K(i, s(a_1))$, $K(i, s) < K(i, s(a_2))$ and $K(i, s(a_2)) < K(i, s(a_1))$, meaning that action $a_2$ and the stuttering action $sta$ are dominated by action $a_1$. Thus, we have $\text{a}^*_s(s) = \{a_1\}$.

### 3 DEFINING OPPORTUNISTIC PROPENSITY

An agent will perform opportunistic behavior when he has the ability and the desire of doing it, which is called opportunistic propensity in [7]. By intuition, we can eliminate opportunism in the system by removing the ability or the desire. In this section, we will provide the definition of opportunistic propensity, serving as a prerequisite of the mechanism design for eliminating opportunism. Opportunism is a selfish behavior that takes advantage of relevant knowledge asymmetry and results in promoting one agent’s own value and demoting another agent’s value. It means that it is performed with the precondition of relevant knowledge asymmetry and the effect of value opposition. Firstly, knowledge asymmetry is defined as follows.

**Definition 3.1 (Knowledge Asymmetry).** Given two agents $i$ and $j$, and a formula $\phi$, knowledge asymmetry about $\phi$ between agent $i$ and $j$ is the abbreviation:

$$\text{Knowasym}(i, j, \phi) = K_i \phi \land \neg K_j \phi \land K_i (\neg K_j \phi).$$

It holds in a state where agent $i$ knows $\phi$ while agent $j$ does not know $\phi$ and this is also known by agent $i$. It can be the other way around for agent $i$ and agent $j$. But we limit the definition to one case and omit the opposite case for simplicity. Now we can define opportunism as follows:

**Definition 3.2 (Opportunism Propensity).** Given two agents $i$ and $j$, the assertion Opportunism($i$, $j$, $a$) that action $a$ performed by agent $i$ is opportunistic behavior is defined as:

$$\text{Opportunism}(i, j, a) := \text{Knowasym}(i, j, a) \land \text{Mpreferred}(i, a, b),$$

where $a$ is the proposition that agent $i$ prefers $a$ to $b$. We use OPP($i, j, s$) to denote the set of opportunistic behavior that can be performed by agent $i$ to agent $j$ in state $s$. That is,

$$\text{OPP}(i, j, s) = \{a \in \text{Ac}(i, s) \mid M, s \models \text{Opportunism}(i, j, a)\}.$$

This definition shows that if the precondition, Knowasym, is satisfied in a given state then the performance of action $a$ will be opportunistic behavior. As the definition is given with the value systems of agent $i$ and agent $j$, a value system profile ($V_i, V_j$) corresponds to one type of opportunistic behavior. The asymmetric knowledge that agent $i$ has is about the change of the truth value of $v^*$ and $w^*$ along the transition by action $a$, where $v^*$ and $w^*$ are the propositions that agent $i$ and agent $j$ most prefer along the transition respectively. It follows that agent $j$ is partially or completely not aware of it. Definition 3.2 follows our definition of opportunism for reasoning about opportunistic propensity of an agent in a state. As is stressed in [5], opportunistic behavior is performed by intent rather than by accident. In this paper, instead of explicitly modeling intention, we interpret it from agents’ rationality that they always intentionally promote their own values. We can derive a proposition from the definition, which is the effect of opportunism.

**Proposition 3.3 (Value Promotion and Demotion).** Given a multi-agent system $M$ and an opportunistic behavior $a$ performed by agent $i$ to agent $j$ in state $s$, action $a$ will promote agent $i$'s value but demote agent $j$'s value, which can be formalized as

$$M, s \models \text{Opportunism}(i, j, a) \text{ implies } s \preceq_s a(s) \text{ and } s >_j s(a).$$

**Proof.** From $M, s \models \text{Opportunism}(i, j, a)$ we have: $M, s \models K_i (\text{promoted}(v^*, a) \land \text{demoted}(w^*, a))$. And thus, since all knowledge is true, we have that $M, s \models \text{promoted}(v^*, a) \land \text{demoted}(w^*, a)$. Since $v^* = \text{Mpreferred}(i, s(a))$ and $w^* = \text{Mpreferred}(j, s(a))$, using Definition 2.5, we can conclude $s \prec_s a(s)$ and $s >_j a(s)$.

**Example 3.4.** Figure 1 shows the example of selling a broken cup: The action selling a cup is denoted as $\text{sell}$ and we use two value systems $V_i$ and $V_j$ for the seller and the buyer respectively. State $s_1$ is the seller’s epistemic alternative, while state $s_1$ and $s_2$ are the buyer’s epistemic alternatives. We also use a dashed circle to represent the buyer’s knowledge $K(b, s_1)$ (not the seller’s). In this example, $K(s, s_1) \subset K(b, s_1)$. Moreover,

$$hm = \text{Mpreferred}(s, s_1, s_2(\text{sell})),$$

$$\neg hb = \text{Mpreferred}(b, s_1, s_2(\text{sell})),$$

meaning that the seller only cares if he gets money from the transition, while the buyer only cares about if he doesn’t have a broken cup from the transition. Note that having a broken cup ($hb$) is not the same as the cup is broken. We also have

$$M, s_1 \models K_i (\text{promoted}(hm, sell) \land \text{demoted}(\neg hb, sell)),$$

meaning that the seller knows the transition will promote his own value while demote the buyer’s value in state $s_1$. For the buyer, action sell is available in both state $s_1$ and $s_2$. However, $hb$ doesn’t hold in both $s_1(\text{sell})$ and $s_2(\text{sell})$, so he doesn’t know whether he has a broken cup or not after action sell is performed. Therefore, there is knowledge asymmetry between the seller and the buyer about the value changes from $s_1$ to $s_2(\text{sell})$. Action sell is potentially opportunistic behavior in state $s_1$.

![Figure 1: Selling a broken cup](image-url)
4 ELIMINATING OPPORTUNISM USING AN 
EPISTEMIC APPROACH

One possible way to eliminate opportunism in the system is to 
remove the possibility of being opportunistic for agents. Since the 
precondition of opportunistic behavior is knowledge asymmetry 
in all states, we can simply prevent the satisfaction of knowledge 
asymmetry so that it is impossible for agents to perform oppor-
tunistic behavior. If we are interested in how the system will behave 
after updating agents’ knowledge, we enter the field of dynamic 
epistemic logic. Dynamic Epistemic Logic is the study of modal 
logics of model change by epistemic and doxastic consequences 
of actions such as public announcements and epistemic actions 
[2][14]. Opportunism can be eliminated through revealing certain 
information to the agent involved, such that knowledge asymmetry 
is removed. This requires the system or someone else in the system 
to be aware of the information that needs to be revealed. Since the 
system is not aware of the value system of each agent but has a 
finite set of possible value systems for each agent, we argue that it 
is still practical for the system to reveal the important facts to 
the agent involved. For example, given two possible value systems 
of the buyer, namely one that cares about the usage of the cup and 
the other one that cares about the appearance of the cup, the system 
can make a 3D scan of the cup and then send it to the buyer, 
so that the buyer gets valuable information about the transaction 
to decide whether to buy the cup. The event or the procedure is 
called a revealing update that is performed by the system and re-
sults in updating agents’ knowledge, and we want to study how 
to eliminate opportunism by revealing updates in this section. In 
this paper, we denote a revealing update as reveal(φ) that reveals 
whether or not formula φ is true. Given a multi-agent system, our 
logical language \( \mathcal{L}_{KA} \) is an extension of \( \mathcal{L}_{KA} \) as follows:

\[
\phi ::= p | \neg \phi | \phi_1 \vee \phi_2 | K_i \phi | \phi_1 \phi_2 | \text{reveal}(\phi_1)_{i/\psi} \quad (i \in \text{Agt}, a \in \text{Act})
\]

As is standard, formulas with revealing updates are evaluated as follows:
given a multi-agent system \( M \) and a state \( s \) in \( M \),

- \( M, s \models \text{reveal}(\phi_1)_{i/\psi} \iff M \text{reveal}(\phi_1), s \models \psi \)

where

\( M \text{reveal}(\phi_1) = (\text{Agt}, S, \text{Act}, \pi, \mathcal{K}', \mathcal{R}, s_0, V_1, ..., V_n) \)

and \( \mathcal{K}' \) is defined as follows:

\( sK'(i)s' \iff (sK(i)s')' \) and \( (M, s \models \phi \iff M, s' \models \phi) \).

The above semantics shows that, after the system performs the 
revealing update reveal(\( \phi \)) to agent \( i \), agent \( i \)’s knowledge about 
\( \phi \) gets updated, in the way that the access regarding to the indi-
istinguishability of the truth value of \( \phi \) is removed while the rest 
of the model remains unchanged. In other words, if \( \phi \) is true in 
state \( s \), the epistemic access of agent \( i \) that connects state \( s \) with 
the states where \( \phi \) is false will be removed; if \( \phi \) is false in state \( s \), 
The epistemic access of agent \( i \) that connects state \( s \) with the states 
where \( \phi \) is true will be removed. Notice that, after performing a 
revealing update, it is always possible to make the system consistent 
with our no-learning and no-forgetting restriction by repeatedly 
removing corresponding epistemic access. As this part of making 
consistent is not what we want to study in this paper, we skip its 
formal definition. We can also see update reveal(\( \phi \)) as a process 
of monitoring performed by the system for the given agent, distin-
guishing states which satisfy \( \phi \) from those which do not satisfy \( \phi \). Since this procedure returns a value from the set \{\( \phi, \neg \phi \)\}, in 
the rest of the paper we always discuss two cases where \( \phi \) holds 
and doesn’t hold in the actual state for any definition and proof.

We have the following validity, given a multi-agent system \( M \), a 
revealing update reveal(\( \phi_i \)),

\[
M \models \phi \implies [\text{reveal}(\phi_i)_{i}]K_i \phi,
\]

which means that if \( \phi \) holds then agent \( i \) knows \( \phi \) after \( \phi \) is revealed. 
Further, if the system reveals something to an agent that he has 
already knew, the model will remain the same. We formalize it as 
if \( M \models K_i \phi \), then \( M \text{reveal}(\phi_i) = M \).

This is because the revealing update will not cause any epistemic 
access removal from the model.

In this paper, we want to investigate how to eliminate the per-
formance of opportunism, typically through removing knowledge 
asymmetry in the system. In order to do that, we firstly introduce 
the notion Eliminating Opportunism by a Revealing Update: we say 
that a revealing update can eliminate opportunism if and only if 
the revealing update disables its performance, namely precondition 
knowledge asymmetry is removed by the revealing update. Formally,

Definition 4.1 (Eliminating Opportunism by a revealing update). 
Given a multi-agent system \( M \), an opportunistic behavior \( a \) per-
formed by agent \( i \) to agent \( j \) in state \( s \), and a revealing update 
\( \text{reveal}(\xi)_j \), we say the revealing update can eliminate opportunistic 
behavior \( a \) iff

\[
M, s \models [\text{reveal}(\xi)_j] \neg \text{Knowasym}(i, j, \text{promoted}(v^*, a) \land \text{demoted}(w^*, a)),
\]

where \( v^* = \text{Mpreferred}(i, s, a) \) and \( w^* = \text{Mpreferred}(j, s, s(a)) \).

This definition shows how a revealing update eliminates oppor-
tunistic behavior: revealing update \( \text{reveal}(\xi)_j \) disables the perfor-
mance of opportunistic behavior \( a \) by making knowledge asymme-
try false in the new system. Notice that based on the semantics of 
our framework, action \( a \), which was opportunistic, is still not re-
moved. However, since there is no knowledge asymmetry between 
agent \( i \) and agent \( j \), agent \( j \) can prevent agent \( i \) from performing 
opportunistic behavior \( a \), or can still accept it. In the latter case, 
action \( a \) is no longer opportunistic as knowledge asymmetry is 
false. For instance, sell and buy are synchronized to be one action. 
After the system reveals to the buyer that the cup is broken, the 
buyer will not buy the cup so that the deal cannot be done, or the 
buyer will still buy the broken cup as it is his only choice, but the 
latter case is not opportunistic behavior since there is no knowledge 
asymmetry about the deal. Moreover, as the system is not aware of 
the value system of each agent, the system reveals to agent \( j \) all 
the information that he might most care about in the transition, given 
a set of possible value systems of agent \( j \). We can immediately have 
the following proposition, which shows the relationship between 
revealing updates and asymmetric knowledge:

Proposition 4.2. Given a multi-agent system \( M \), an opportunistic 
behavior \( a \) performed by agent \( i \) to agent \( j \) in state \( s \) and a revealing 
update \( \text{reveal}(\xi)_j \), the revealing update can eliminate opportunistic 
behavior \( a \) if
In principle, given a set of possible value system profiles and a privacy norm, the system has to consider every possible value system profile in order to identify an action to be opportunistic, and then think about whether there exists a revealing update that can eliminate opportunistic behavior and respect the privacy norm as well. In this paper, we skip the first part for simplification, assuming that opportunistic behavior is given, in order to focus on the study about the trade-off between eliminating opportunistic behavior and respecting the privacy norm. Namely, suppose we already identified an action to be opportunistic behavior with a possible value system profile, a question arises: Given opportunistic behavior and a privacy norm, does there exist a revealing update that can eliminate opportunistic behavior and respect the privacy norm as well? Intuitively, an agent gets to know something after something was revealed to the agent, but the revealing update might disrespect another agent’s privacy, which is stated by our privacy norms in the system. The following proposition shows that in which case a revealing update respects a privacy norm:

**Proposition 4.4.** Given a multi-agent system $M$ in a state $s$, a privacy norm Knowasym($i, j, y$) $\in \Pi(s)$ with respect to formula $y$, and a revealing update $\text{reveal}(\xi)$, the revealing update respects privacy norm $\text{Knowasym}(i, j, y)$ if:

- in the case $M, s \models \xi, M, s \models \neg K_j(\xi \rightarrow y)$,
- in the case $M, s \models \neg \xi, M, s \models \neg K_j(\neg \xi \rightarrow y)$.

**Proof.** In order to respect privacy norm $\text{Knowasym}(i, j, y)$, according to Definition 3.1, we need to ensure $M, s \models [\text{reveal}(\xi)]\neg K_jy$ so that $M, s \models [\text{reveal}(\xi)]\text{Knowasym}(i, j, y)$. In the case where $\xi$ holds, $M, s \models [\text{reveal}(\xi)]K_j\xi$ after the revealing update is performed to agent $j$. Furthermore, $M, s \models [\text{reveal}(\xi)](\neg \xi \rightarrow y)$ implies that there exists $s' \in \mathcal{K}(j, s) : M, s' \models (\neg \xi \rightarrow y)$, which is equivalent to $M, s' \models \xi \land \neg y$. Since agent $j$’s epistemic access which connects $\neg \xi$-state to state $s$ gets removed after the revealing update is performed, state $s'$ where $\xi \land \neg y$ holds is still in agent $j$’s knowledge set. In other words, there exists $s' \in \mathcal{K}(j, s) : M, s' \models \text{reveal}(\xi), s' \models \xi \land \neg y$. Therefore, we can conclude that $M, s \models [\text{reveal}(\xi)](\neg K_jy)$ and it leads to $M, s \models [\text{reveal}(\xi)]\text{Knowasym}(i, j, y)$. We can prove it in a similar way when $\neg \xi$ holds in state $s$.

The proposition shows that privacy norm $\text{Knowasym}(i, j, y)$ is respected if agent $j$ is not aware of the inference. Conversely, if the above statement doesn’t hold, the revealing update will reveal the information that the system wants to keep in private between agents. From Proposition 4.2 and Proposition 4.4, we can see our research problem is equivalent to the problem whether there exists a formula $\xi$ such that the formulas from both propositions hold. Therefore,

**Proposition 4.5.** Given a multi-agent system $M$ in state $s$, an opportunistic behavior $a$ performed by agent $i$ to agent $j$, a privacy norm $\text{Knowasym}(i, j, y) \in \Pi(s)$ and a revealing update $\text{reveal}(\xi)$, $\text{reveal}(\xi)$ can eliminate opportunistic behavior $a$ and respect privacy norm $\text{Knowasym}(i, j, y)$ if:

- in the case $M, s \models \xi, M, s \models K_j(\text{promoted}(v^*, a) \land \text{demoted}(w^*, a))$.
- in the case $M, s \models \neg \xi, M, s \models K_j(\neg \xi \rightarrow (\text{promoted}(v^*, a) \land \text{demoted}(w^*, a)))$.

**Proof.** When $\xi$ holds in state $s$, $M, s \models K_j(\text{promoted}(v^*, a) \land \text{demoted}(w^*, a))$ is performed. As $M \models K_j(\text{promoted}(v^*, a) \land \text{demoted}(w^*, a))$ implies $M \models K_j(\text{promoted}(v^*, a) \land \text{demoted}(w^*, a))$, we have $M, s \models K_j(\text{promoted}(v^*, a) \land \text{demoted}(w^*, a))$. Thus, there is no knowledge asymmetry between agent $i$ and agent $j$ about formula $\text{promoted}(v^*, a) \land \text{demoted}(w^*, a)$, Therefore, according to Definition 4.1, $\text{reveal}(\xi)$ eliminate opportunistic behavior $a$. We can prove it in a similar way when $\neg \xi$ holds in state $s$. □
where \( v^* = M\text{preferred}(i, s, s(a)) \) and \( w^* = M\text{preferred}(j, s, s(a)) \).

Proof. The statement is the combination of the statements from Proposition 4.2 and Proposition 4.4. When agent \( j \) is aware of \( \xi \rightarrow (\text{promoted}(v^*, a) \land \text{demoted}(w^*, a)), \text{reveal}(\xi) \), can eliminate opportunistic behavior \( a \); when agent \( j \) is not aware of \( \xi \rightarrow \gamma, \text{reveal}(\xi) \), respects privacy norms Knowasym(i, j, γ). Again, we can prove it in a similar way when \( \neg \xi \) holds in state \( s \).

Essentially, the above proposition shows the relation among a revealing update, agents’ value systems and a privacy norm: if what an agent cares about, which his value system reflects, is not respected by the system through setting corresponding privacy norms, such a revealing update to the agent doesn’t exist. In other words, it is dependent on the compatibility between agents’ value systems and the privacy norms in the system. For example, for the case where \( \xi \) holds, in order to eliminate opportunistic behavior \( a \), the system has to reveal (verify) \( \xi \) to agent \( j \), who knows that \( \xi \) implies value opposition along the transition. However, if he is also aware of the formula \( \xi \rightarrow \gamma \), such a revealing update will reveal to agent \( j \) the information about \( \gamma \), which is against the privacy norm. Hence, there is no revealing update that can eliminate opportunistic behavior \( a \) and respect the privacy norm with respect to \( \gamma \) as well. Further, sometimes formula \( \xi \rightarrow \gamma \) is valid in \( M \) thus it becomes universal knowledge in the system. In that case, revealing update \( \text{reveal}(\xi) \) will always reveal the information about \( \gamma \) we want to keep in private. Thus, we have to remove privacy norm Knowasym(i, j, γ) so that it is allowed to perform revealing update \( \text{reveal}(\xi) \) to eliminate opportunistic behavior \( a \), which can be seen as an alternative normative approach.

Example 4.6. We again consider the scenario shown in Example 3.4. There is knowledge asymmetry between the seller and the buyer,

\[
\text{Knowasym}(s, b, \text{promoted}(hm, sell) \land \text{demoted}(\neg hb, sell)),
\]

which is equivalent to

\[
\text{Knowasym}(s, b, \neg hm \land (sell)hm \land \neg hb \land (sell)hb).
\]

In this scenario the seller knows the transition will promote his own value while demote the value of the buyer; but the buyer is not aware of the demotion part, as \( (sell)hb \) doesn’t hold in both state \( s_1 \) and state \( s_2 \). Now the buyer performs revealing update \( \text{reveal}(\neg fb) \) to check whether the cup is broken or not, and he also knows that his value will get demoted while the buyer’s value will get promoted if the cup is broken, that is,

\[
M, s \models Kb(broken \rightarrow (\text{promoted}(hm, sell) \land \text{demoted}(\neg hb, sell))),
\]

which implies

\[
M, s \models Kb(broken \rightarrow Kb(\text{promoted}(hm, sell) \land \text{demoted}(\neg hb, sell))).
\]

Since the cup is actually broken \( (M, s \models broken) \), the buyer gets to know the cup is broken after the system performs revealing update \( \text{reveal}(\neg fb) \) to him \( (M, s \models Kb(broken)) \) and thus he knows his value will get demoted while the buyer’s value will get promoted,

\[
M, s \models Kb(\text{promoted}(hm, sell) \land \text{demoted}(\neg hb, sell)).
\]

Therefore, there is no knowledge asymmetry about the transition between the seller and the buyer (shown in Fig. 2), which prevents the seller from opportunistically selling the broken cup to the buyer, according to Definition 3.2. Next we suppose a privacy norm Knowasym(s, b, oprice) in the system, which means that the seller should keep the original price in private. Since inference \( broken \rightarrow oprice \) is not valid in \( M \) intuitively, the buyer is not aware of it,

\[
M, s \models \neg Kb(broken \rightarrow oprice).
\]

Therefore, revealing update \( \text{reveal}(\neg fb) \), won’t reveal the original price to the buyer and privacy norm Knowasym(s, b, oprice) is still respected in the updated system.

![Figure 2: Update by revealing update reveal(broken)_b.](image)

5 RELATION TO MECHANISM DESIGN

Mechanism design is a field to design a game with desirable properties (outcomes) for various agents to play [8] [11]. Given agents’ preferences \( \preceq \) and an assumed solution concept \( g \) that defines agents’ way of finding optimal outcomes, we can make a prediction of the outcomes that will be achieved, which is represented as \( g(\preceq) \).

Given agents’ preferences \( \preceq \) and a social choice rule \( f \) that specifies the criteria of the desirable outcomes, we say that \( f(\preceq) \) are the set of social optimal outcome, which are the outcomes we want to have. Since agents’ preference might be unknown to us, our goal is to design mechanisms such that for all the possible preference \( \preceq \) the predicted outcomes \( g(\preceq) \) coincide with (or is a subset of) the desirable outcomes \( f(\preceq) \) (more elaboration can be found in [4]). In this paper, we take a slightly different view of mechanism design from the traditional one above: we consider a mechanism as an operation or an update to the system, which is a revealing update. When applying the theory of mechanism design to eliminate opportunism, we see agents’ rational alternatives as predicted outcomes, opportunistic behaviors as undesirable outcomes, and our goal is to design revealing updates to the system such that for all the possible value system profiles the intersection of an agent’s rational alternatives (using our decision theory) and opportunistic behaviors in the new system is empty. In this section, we will discuss how a revealing update implements non-opportunism respectively.
Given an opportune behavior, we know what kind of information the system needs to reveal to an agent for eliminating it. However, if we take into account an agent’s decision-making, it can be the case where it is not optimal for the agent to perform such an opportunistic behavior thus it is not necessary to eliminate it. In this sense, we connect revealing updates with rational alternatives. Hence, the goal of this paper is to find out an update, namely a revealing update, such that it is not optimal for the agent to behave opportunistically after it is implemented. Given a value system for agent \( i \), we know the set of agent \( i \)'s rational alternatives \( a_i^r(s) \). Given a value system profile for agent \( i \) and \( j \), we can identify the set of opportunistic behaviors \( \text{OPP}(i, j, s) \) that agent \( i \) and \( j \) are involved in. We use \( a_i^r(s) \) reveal(\( \xi_j \)) and \( \text{OPP}(i, j, s) \) reveal(\( \xi_j \)) to denote the set of rational alternatives and the set of opportunistic behaviors after reveal(\( \xi_j \)) is performed in state \( s \) respectively. Because opportunistic behavior is undesired from the perspective of the system and agents form their rational alternatives (possibly opportunistic) based on their value systems, it is important to know whether a revealing update implements non-opportunism. Formally, we define non-opportunistic implementation as follows:

**Definition 5.1 (Non-opportunistic Implementation).** Given a multi-agent system \( M \) with two agents \( i \) and \( j \) in state \( s \), and a revealing update reveal(\( \xi_j \)), we say that reveal(\( \xi_j \)) implements non-opportunism iff \( a_i^r(s) \) reveal(\( \xi_j \)) \( \cap \) \( \text{OPP}(i, j, s) \) reveal(\( \xi_j \)) \( = \) \( \emptyset \).

A revealing update implements non-opportunism if and only if the intersection between rational alternatives and opportunistic behaviors becomes an empty set after the revealing update is performed. Clearly, this concerns the update that a revealing update brings to the system. With our update logic of revealing updates, we now discuss how a revealing update influences an agent’s decision-making and the identification of opportunistic behavior.

**Proposition 5.2.** Given a multi-agent system \( M \) with two different agents \( i \) and \( j \) in state \( s \), and a revealing update reveal(\( \xi_j \)), agent \( i \)'s rational alternatives will remain the same after reveal(\( \xi_j \)) is performed in state \( s \), which is formalized as \( a_i^r(s) = a_i^r(s) \) reveal(\( \xi_j \)).

Proof. Since revealing update reveal(\( \xi_j \)) is performed by the system to agent \( j \), agent \( i \)'s epistemic structure will remain the same after reveal(\( \xi_j \)) is performed. Hence, according to Definition 2.6 and 2.7, agent \( i \)'s subjectively available actions and rational alternatives will remain the same after reveal(\( \xi_j \)) is performed. \( \Box \)

**Proposition 5.3.** Given a multi-agent system \( M \) with two different agents \( i \) and \( j \) in state \( s \), and a revealing update reveal(\( \xi_j \)), opportunistic behaviors performed by agent \( i \) to agent \( j \) will not become more after reveal(\( \xi_j \)) is performed, which is formalized as \( \text{OPP}(i, j, s) \supseteq \text{OPP}(i, j, s) \) reveal(\( \xi_j \)).

Proof. Given a value system profile for agent \( i \) and \( j \), we can identify the set of opportunistic behaviors \( \text{OPP}(i, j, s) \) in a state. Because reveal(\( \xi_j \)) causes update of agent \( j \)'s knowledge, knowledge asymmetry will become false after reveal(\( \xi_j \)), and thus some actions will become non-opportunistic. Because the system might reveal the information that is not relevant to any opportunistic behavior, it is possible that all the opportunistic behaviors remain unchanged. \( \Box \)

If we limit a revealing update to the one that is performed to agent \( j \), agent \( i \)'s rational alternatives will remain the same while opportunistic behaviors performed by agent \( i \) to agent \( j \) will remain the same or become less, after reveal(\( \xi_j \)) is performed. Therefore, if a revealing update can eliminate all the actions in the intersection of rational alternatives and opportunistic behavior, it implements non-opportunism. Notice that action \( a \), which was opportunistic behavior, is still in agent \( i \)'s rational alternatives, but it is not opportunistic any more because knowledge asymmetry regarding opportunistic behavior \( a \) is already removed. As for Example 4.6, we see that reveal(broken)\(_b\) can eliminate opportunistic behavior sell. Even though the seller can still sell the broken cup to the buyer, it is not opportunistic behavior any more because the buyer already know that he will have a broken cup. Therefore, we can conclude that given a set of value system profiles \( \bar{V} = (((V_s, V_b)) \) revealing update reveal(broken)\(_b\) implements non-opportunism.

### 6 CONCLUSION

Opportunism is a behavior that takes advantage of relevant knowledge asymmetry and results in promoting an agent’s own value and demoting another agent’s value. As opportunistic behavior has undesirable results for other agents who participate in the system, it is important to design mechanisms to eliminate opportunism. In this paper we developed an epistemic approach to eliminate opportunism in multi-agent systems: we eliminated opportunism by removing the precondition of opportunism knowledge asymmetry, which disables the performance of opportunism. Knowledge asymmetry is removed by agents’ revealing updates, which might reveal the information that the system wanted to keep private between agents through setting privacy norms. So we investigated the balance between eliminating opportunism and respecting agents’ privacy. Finally, we related our approach to the theory of mechanism design. However, we do recognize some downsides of our proposed mechanism. Firstly, in order to eliminate opportunism, the system has to reveal to an agent the information that he might care about for the transition given his possible value systems. This is indeed an ideal practice because it will be difficult to achieve it when the set of possible value systems become large. Secondly, in order to reveal useful information to agents, the system has to first identify if a given action is opportunistic behavior with a set of value system profiles for the agents involved, and then reveal appropriate information to the agents to eliminate opportunism. Those revealing updates should not be demotivated by the system through setting privacy norms. This indeed puts a burden on the designer before implementing any privacy norms, as agents’ value systems are initially unknown. An agent performs opportunistic behavior when he has the ability and the desire of doing that. Instead of removing the ability, future work can be done by removing the desire, namely making the choice of being opportunistic not optimal. As there exists trade-off between eliminating opportunism and respecting agents’ privacy, it will be interesting to eliminate opportunism through removing privacy norms.

### ACKNOWLEDGMENTS

The authors would like to thank Chinese Scholarship Council and Centrum Wiskunde & Informatica for their financial support.
REFERENCES


