Tractable Model Checking for Fragments of Higher-Order Coalition Logic*

Patrick Doherty
Dept. of Computer and Information Science, Linköping University, Sweden
patrick.doherty@liu.se

Barbara Dunin-Kęplicz
Institute of Informatics, Warsaw University, Poland
and Institute of Computer Science, Polish Academy of Sciences, Warsaw, Poland
keplicz@mimuw.edu.pl

Andrzej Szałas
Institute of Informatics, Warsaw University Warsaw, Poland
and Dept. of Computer and Information Science, Linköping University, Sweden
andrzej.szalas@{mimuw.edu.pl,liu.se}

ABSTRACT

A number of popular logical formalisms for representing and reasoning about the abilities of teams or coalitions of agents have been proposed beginning with the Coalition Logic (CL) of Pauly. Ågotnes et al introduced a means of succinctly expressing quantification over coalitions without compromising the computational complexity of model checking in CL by introducing Quantified Coalition Logic (QCL). QCL introduces a separate logical language for characterizing coalitions in the modal operators used in QCL. Boella et al, increased the representational expressibility of such formalisms by introducing Higher-Order Coalition Logic (HCL), a monadic second-order logic with special set grouping operators. Tractable fragments of HCL suitable for efficient model checking have yet to be identified. In this paper, we relax the monadic restriction used in HCL and restrict ourselves to the diamond operator. We show how formulas using the diamond operator are logically equivalent to second-order formulas. This permits us to isolate and define well-behaved expressive fragments of second-order logic amenable to model-checking in PTIME. To do this, we appeal to techniques used in deductive databases and quantifier elimination. In addition, we take advantage of the monotonicity of the effectivity function resulting in exponentially more succinct representation of models. The net result is identification of highly expressible fragments of a generalized HCL where model checking can be done efficiently in PTIME.

Categories and Subject Descriptors

F.4 [**Mathematical Logic And Formal Languages**]: Miscellaneous

General Terms

Theory, Verification

*This work is partially supported by grants from the ELLIIT Excellence Center at Linköping-Lund in Information Technology, the Swedish Research Council (VR) Linnaeus Center CADICS, VR grant 90385701, NFFP5-The Swedish National Aviation Engineering Research Programme and grant N N206 399334 from the Polish MNiSW.

Cite as: Tractable Model Checking for Fragments of Higher-Order Coalition Logic, Patrick Doherty, Barbara Dunin-Kęplicz and Andrzej Szałas, *Proc. of 10th Int. Conf. on Autonomous Agents and Multiagent Systems (AAMAS 2011)*, Tumer, Yolum, Sonenberg and Stone (eds.), May, 2–6, 2011, Taipei, Taiwan, pp. 743-750.

Keywords

Coalition Formation, Coalition Logic, Model Checking, Complexity

1. INTRODUCTION

In recent years, developing formal techniques for representing and reasoning about the abilities of teams or coalitions has become a major focus of research in the areas of artificial intelligence and multiagent systems [2–10, 12, 16, 17, 19, 21–29]. In particular, combining ideas and techniques from game theory, logic and social theory has become highly relevant due to the widespread use of social software and trends in robotics and agent systems where cooperation among such agents is becoming increasingly important. A number of popular logical formalisms for representing and reasoning about the abilities of teams or coalitions of agents have been proposed beginning with the Coalition Logic (CL) of Pauly [19] which is a propositional multimodal logic. Recent trends in development of logical formalisms for reasoning about coalitions have tried to increase expressivity of such formalisms while retaining tractability in the reasoning components associated with such formalisms. For instance, Ågotnes et al [3] introduce a means of succinctly expressing quantification over coalitions without compromising the computational complexity of model checking in CL by introducing Quantified Coalition Logic (QCL). More recently, Boella et al [7], increased the representational expressibility of such formalisms by introducing Higher-Order Coalition Logic (HCL), a monadic second-order logic with special set grouping operators. HCL subsumes both CL and QCL representationally, and includes a sound and complete axiomatization for weakly playable frames, but currently lacks a tractable reasoning component.

Due to the modal nature of many of these formalisms which are based on the use of effectivity functions as part of coalition frames, the major computational problem has been that of model checking.

> Given a *succinct* representation of a model $\mathcal{M}$, a state s, and a formula φ of your favorite coalition logic $\mathcal{L}$, is it the case that $\mathcal{M}, s \models_{\mathcal{L}} \varphi$?

In addition to the representational problem, research focus associated with the reasoning problem has been placed on finding succinct representations of models in $\mathcal{L}$, extending the expressivity of formulas φ and trying to guarantee tractability of the model checking problem for the full language or its fragments used in $\mathcal{L}$.

Higher-order logic is particularly suited as a representation language for modeling the abilities and interactions between coalitions. It is representationally expedient in the sense that coalitions are in fact sets of agents and one wants to represent such sets and their properties in as direct a way as possible. This of course can be done directly and succinctly in higher-order logic. Boella et al [7] provide very convincing arguments in this respect. On the other hand, reasoning in higher-order logic is more problematic. Yet there are well-behaved fragments of such logics that deserve investigation and surprisingly, one can isolate fragments which are representationally expressive and also computationally tractable.

This is both the focus and contribution of this paper. Our contribution is to propose a higher-order logic HCL* which essentially subsumes HCL representationally and semantically. Additionally, we isolate a number of interesting fragments of HCL* which are amenable to tractable model checking in PTIME using succinct implicit representations of model frames. The techniques used to do this involve quantifier elimination [13] and the use of standard techniques from deductive database theory [1, 18].

In Sections 2 and 3 we give an overview of coalition logic [19], quantified coalition logic [3] and higher-order coalition logic [7] to set the context and provide scientific continuity. In Section 4 we propose the higher-order logic HCL* which is a generalization of HCL. Section 5 discusses various succinct representations of models using deductive database techniques. In Section 6 we define several fragments of HCL* and provide lemmas showing that formulas from these fragments are amenable to model-checking in PTIME. Section 7 summarizes assertion types that can be model checked in PTIME. We then conclude with some comments and future work in Section 8.

2. COALITION LOGIC

Coalition Logic (CL) [19] is a propositional modal logic, with modalities indexed by coalitions. The semantics of CL is based on the concept of an *effectivity function* developed in social choice theory to model the ability of a group of individuals. In CL, it is relativized to state and has the form

$$\mathcal{E} : \mathcal{P}(Ag) \times S \longrightarrow \mathcal{P}(\mathcal{P}(S)) \tag{1}$$

where Ag is a set of agents, S is a set of states and $\mathcal{P}(.)$ denotes the powerset of a given set.

For a given coalition $C \subseteq Ag$ and a state $s \in S$, C can cooperate to ensure that for any $T \in \mathcal{E}(C, s)$, the next state will be in T regardless of the actions of other agents outside C. A variety of possible strategies can lead to a set of possible outcomes. To gain some intuitions concerning such functions consider the following example based on one considered in [19].[1]

EXAMPLE 1. *Angelina has to decide whether she wants to marry Edwin, the Judge, or stay single. Edwin and the Judge each can similarly decide whether they want to stay single or marry Angelina. This situation can be modeled using a function $\mathcal{E}$ of the form (1) as follows. The set of agents is $Ag = \{a, e, j\}$ and the set of states is $S = \{s_0, s_s, s_e, s_j\}$, where s_0 is an initial state, where Angelina, Edwin and Judge are singles, s_s is a state where Angelina remains single, s_e where she marries Edwin, and s_j where she marries the Judge.*

Angelina has the right to remain single, so $\{s_s\} \in \mathcal{E}(\{a\}, s_0)$. Edwin can only guarantee that he does not marry Angelina, so we have $\{s_s, s_j\} \in \mathcal{E}(\{e\}, s_0)$. Analogously, for the Judge, we have $\{s_s, s_e\} \in \mathcal{E}(\{j\}, s_0)$. Angelina and Edwin together can achieve any situation except the one where Angelina marries the Judge, and hence $\{s_s\}, \{s_e\} \in \mathcal{E}(\{a, e\}, s_0)$. Again, the situation is analogous for the Judge: $\{s_s\}, \{s_j\} \in \mathcal{E}(\{a, j\}, s_0)$. Edwin and the Judge can together guarantee that Angelina remains single, so $\{s_s\} \in \mathcal{E}(\{e, j\}, s_0)$.

Note that Angelina can act as a dictator forcing everybody to stay single. On the other hand, neither Edwin nor the Judge have such a strong strategy. □

Coalition Logic is a propositional multimodal logic, where formulas are defined by the following grammar:

$$\varphi ::= \top \mid p \mid \neg\varphi \mid \varphi \vee \varphi \mid [C]\varphi \tag{2}$$

where Ag is fixed, $C \subseteq Ag$ and p ranges over the set of Boolean variables Φ_0. The intended meaning of $[C]\varphi$ is that coalition C has the ability to achieve φ.

Given Ag, a *model* $\mathcal{M}$ is a triple $\langle S, \mathcal{E}, \pi \rangle$, where

- $S = \{s_1, \ldots, s_n\}$ is a finite non-empty set of *states*
- $\mathcal{E}$ is an effectivity function
- $\pi : S \longrightarrow \mathcal{P}(\Phi_0)$ is a *valuation function*, which for every state $s \in S$ gives the set of Boolean variables satisfied at s.

The satisfaction relation is defined as usual for $\top$, atomic variables and connectives. For the modal case we have:

$$\mathcal{M}, s \models_{CL} [C]\varphi \quad \text{iff there is } T \in \mathcal{E}(C, s) \text{ such that for all } t \in T \text{ we have } \mathcal{M}, t \models_{CL} \varphi.$$

Table 1: Properties of effectivity function (1).

For every $C \subseteq Ag$, $s \in S$ and $X \subseteq Y \subseteq S$, if $X \in \mathcal{E}(C, s)$ then $Y \in \mathcal{E}(C, s)$	*outcome monotonicity*
For every $C \subseteq D \subseteq Ag$ and $s \in S$, $\mathcal{E}(C, s) \subseteq \mathcal{E}(D, s)$	*coalition monotonicity*
For all $X \subseteq S$ and $s \in S$, if $X \in \mathcal{E}(C, s)$ then $\bar{X} \notin \mathcal{E}(\bar{C}, s)$	*C-regularity*
For all $C \subseteq Ag$, $\mathcal{E}$ is C-regular	*regularity*
For all $X \subseteq S$ and $s \in S$, if $\bar{X} \notin \mathcal{E}(\bar{C}, s)$ then $X \in \mathcal{E}(C, s)$	*C-maximality*
For all $C \subseteq Ag$, $\mathcal{E}$ is C-maximal	*maximality*
For all $X, Y \subseteq S$, $C \subseteq D \subseteq Ag$ and $s \in S$, if $C \cap D = \emptyset$, $X \in \mathcal{E}(C, s)$ and $Y \in \mathcal{E}(D, s)$ then $X \cap Y \in \mathcal{E}(C \cup D, s)$	*superadditivity*

In [20] the some important properties of effectivity functions are studied. These properties are shown in Table 1, where $\bar{C} \stackrel{\text{def}}{=} Ag \backslash C$ and $\bar{X} \stackrel{\text{def}}{=} S \setminus X$. Restricting effectivity functions with certain properties determines particular classes of models.

An effectivity function $\mathcal{E}$ is *playable* provided that for all $C \subseteq Ag$ and $s \in S$, (i) $\emptyset \notin \mathcal{E}(C, s)$, (ii) $S \in \mathcal{E}(C, s)$, (iii) $\mathcal{E}$ is Ag-maximal, (iv) $\mathcal{E}$ is outcome monotonic and (v) $\mathcal{E}$ is superadditive. The term *playability* is justified by the fact that an effectivity function is playable iff it is an effectivity function of a strategic game (see Theorem 3.2 in [20]).

Coalition monotonicity will play an important role in the context of our model checking results. The following lemma (see, e.g., [19, 20]) guarantees this property for playable effectivity functions.

LEMMA 2. *Every playable effectivity function is regular and coalition monotonic.* □

[1]Characters are actually taken from the comic opera *Trial by Jury* and the example originates from [15].

3. EXTENSIONS OF COALITION LOGIC

Recent work with logical formalisms for representing and reasoning about the abilities of coalitions from a game-theoretic perspective have focused jointly on issues of expressivity and tractability, balancing the two against each other. The major computational problem in this respect is model checking. In this section, we briefly describe two prominent logical formalisms, Quantified Coalition Logic [3] and Higher-Order Coalition logic [7] which attempt to generalize Coalition Logic in several respects. This summary is intended to provide a context for the higher-order logic HCL* and model checking results for fragments of this logic that we introduce in Section 4.

3.1 Quantified Coalition Logic

Quantified Coalition Logic (QCL) [3] is an extension of CL that permits a limited form of quantification over coalitions. Although it provides no increase in expressivity, it is exponentially more succinct than CL and computationally no worse with respect to model checking. Rather than providing C directly in formulas of the form $[C]\varphi$, it allows C to be specified in a special language for coalition predicates given by the grammar:

$$P ::= subseteq(C) \mid supseteq(C) \mid \neg P \mid P \vee P \qquad (3)$$

This allows one to express coalitions based on being a subset (a superset) of a coalition. There are two modalities in QCL:[2]

- $\langle\psi\rangle\varphi$ – there is a coalition satisfying ψ which can achieve φ
- $[\psi]\varphi$ – every coalition satisfying ψ can achieve φ.

3.2 Higher-Order Coalition Logic

Recently, Boella et al [7] introduced Higher-Order Coalition Logic (HCL) as a more general and expressive way to quantify over coalitions. HCL is a monadic second-order logic with special set grouping operators which can be used to characterize different coalitions. Both CL and QCL can be effectively embedded into HCL and there is no need for separate languages to represent coalitions and the effect coalitions have, as is the case with QCL. An axiomatization is provided for HCL and it is shown to be sound and complete for weakly playable semantic structures. Tractable fragments of HCL suitable for efficient model checking have yet to be identified, although this paper will identify a number of such fragments for a related logic HCL*.

We write $\sigma[x := u]$ (respectively, $\sigma[X := U]$) to denote the assignment which differs from σ only in assigning u to x (respectively, U to X).

HCL is a well-behaved fragment of second-order logic, where second-order quantifiers are restricted to binding unary relation variables. Free and bound variables (V_I), relation symbols, connectives $\wedge, \neg$, quantifiers $\forall, \exists$ are defined as in classical first-order logic. To obtain the monadic second-order language, the first-order language is extended by a countable set V_S of set variables (one-argument relation variables) and formulas of HCL are defined using the following grammar:

$$\begin{aligned}\varphi ::= \quad & F(x_1,\ldots,x_k) \mid X(x) \mid \neg\varphi \mid \varphi \vee \varphi \mid \\ & \forall X\varphi \mid \forall x\varphi \mid [\{x\}\varphi]\varphi \mid \langle\{x\}\varphi\rangle\varphi\end{aligned} \qquad (4)$$

where

- $F(x_1,\ldots,x_k)$ is a first-order atomic formula
- $x \in V_I$ and $X \in V_S$
- $\{x\}\varphi$ is a grouping operator which denotes the set of all elements d such that $\varphi[x := d]$ holds.

[2]Note that these modalities are not dual to each other (see [3]).

The intended meaning of modalities in HCL is the same as in the case of QCL. However, HCL offers a much richer language to express properties of coalitions. Consider the following examples, mainly from [7], illustrating the expressiveness of HCL:

- $\forall x(super\_user(x) \rightarrow user(s))$ – any super user is a user
- $\forall X(\forall x(X(x) \rightarrow user(x)) \rightarrow \langle\{y\}X(y)\rangle\varphi)$ – there is a coalition, where all $users$ can achieve φ
- $\langle\{x\}\psi(x)\rangle\varphi \rightarrow \langle\{y\}\exists x(\psi(x) \wedge collaborates(y,x))\rangle\varphi$ – whenever there is a coalition, say C, satisfying ψ which can achieve φ, there is also a coalition consisting of collaborators of at least one member of C which can achieve φ.

Semantic structures for HCL correspond to weak playability. An effectivity function $\mathcal{E}$ is *weakly playable* if for all $C \subseteq D \subseteq Ag$ and $s \in S$, (i) $\emptyset \notin \mathcal{E}(Ag, s)$, (ii) $\emptyset \in \mathcal{E}(D, s)$ implies $\emptyset \in \mathcal{E}(C, s)$, (iii) $\emptyset \notin \mathcal{E}(\emptyset, S)$ implies $S \in \mathcal{E}(C, s)$, (iv) $\mathcal{E}$ is Ag-maximal, (v) $\mathcal{E}$ is outcome monotonic and (vi) $\mathcal{E}$ is superadditive.

HCL uses a *general* or *Henkin* semantics. A more detailed discussion about the semantic basis for HCL is provided in Section 4.

4. COALITION LOGIC HCL*

The goal of this paper is to provide a logical formalism for reasoning about the abilities of coalitions that has high expressiveness, yet is still amenable to tractable model-checking. HCL certainly has high expressiveness and is more general than both CL and QCL. On the other hand, it currently lacks nice computational properties for different fragments of the language. Our results are intended to provide both well-behaved fragments and tractable model checking techniques for higher-order logic using HCL*

In this section, we introduce the higher-order logic HCL*. For the purposes of continuity and context, before providing formal definitions, we list the difference between HCL and HCL* and then remark on some of these differences.

1. HCL restricts quantification over relations to unary (monadic) predicates. HCL* relaxes this restriction and permits quantification over relations of arbitrary arity.

2. HCL includes both the diamond and box operators in the language. HCL* excludes the box operator from the language. Since box can be defined in terms of diamond, this is done for technical reasons pertaining to model-checking and does not limit expressivity of the language in general.

3. HCL uses a *general* or *Henkin* semantics which approximates the standard semantics for higher-order logic. HCL* uses the standard semantics for higher-order logic.

4. HCL restricts frames to those whose effectivity function is weakly playable. HCL* only requires frames to be monotonic (both outcome and coalition monotonic) for our model checking results to apply.

Before commenting on these differences, we provide the syntax and semantics of HCL*.

The syntax of HCL* is given by the following grammar:

$$\begin{aligned}\varphi ::= \quad & \top \mid F(x_1,\ldots,x_k) \mid X(x_1,\ldots,x_k) \mid \\ & \neg\varphi \mid \varphi \vee \varphi \mid \forall X\varphi \mid \forall x\varphi \mid \langle\{x\}\varphi\rangle\varphi\end{aligned} \qquad (5)$$

A *coalition frame* is a tuple $\langle Ag, S, \mathcal{E}\rangle$, where Ag is a finite, nonempty set (of agents), S is a finite set of states and $\mathcal{E}$ is an effectivity function.

A coalition frame is *monotonic* if its effectivity function is both outcome monotonic and coalition monotonic. Monotonicity

is the weakest requirement necessary for our model-checking results. In the rest of the paper we assume that the frames considered are monotonic.

A *coalition model based on a frame* $\langle Ag, S, \mathcal{E}\rangle$ is a tuple $\mathcal{M} = \langle Ag, S, \mathcal{E}, \mathcal{I}, \sigma\rangle$, where:

- $\mathcal{I}$ is a first-order interpretation, for any first order formula α and $s \in S$ it assigns a set of tuples $\alpha^{\mathcal{I}}(s)$ satisfying α in state s
- σ assigns in states: (i) values in Ag to individual variables, (ii) sets of tuples of respective arity to relation variables.[3]

Let $\mathcal{M} = \langle Ag, S, \mathcal{E}, \mathcal{I}, \sigma\rangle$ be a coalition model and $s \in S$. We define the *satisfaction relation* as follows, where $\mathcal{M}'{=}\langle Ag, S, \mathcal{E}, \mathcal{I}, \sigma[x := d]\rangle$ and $\mathcal{M}''{=}\langle Ag, S, \mathcal{E}, \mathcal{I}, \sigma[X := D]\rangle$:

- $\mathcal{M}, s \models \top$
- $\mathcal{M}, s \models F(x_1, \ldots, x_k)$ iff $\langle \sigma(x_1), \ldots, \sigma(x_k)\rangle \in F^{\mathcal{I}}(s)$
- $\mathcal{M}, s \models \neg\varphi$ iff $\mathcal{M}, s \not\models \varphi$
- $\mathcal{M}, s \models \varphi \vee \psi$ iff $\mathcal{M}, s \models \varphi$ or $\mathcal{M}, s \models \psi$
- $\mathcal{M}, s \models X(x_1, \ldots, x_k)$ iff $\langle \sigma(x_1), \ldots, \sigma(x_k)\rangle \in \sigma(X, s)$
- $\mathcal{M}, s \models \forall x\varphi$ iff for all $d \in Ag$, $\mathcal{M}', s \models \varphi$
- $\mathcal{M}, s \models \forall X\varphi$, for a k-argument relation variable X, iff for all $D \in \mathcal{P}(Ag^k)$, $\mathcal{M}'', s \models \varphi$
- $\mathcal{M}, s \models \langle\{x\}\psi\rangle\varphi$ iff there is $C{=}\{d \mid \mathcal{M}', s \models \psi\}$ and $T \in \mathcal{E}(C, s)$ such that for all $t{\in}T$, $\mathcal{M}, t{\models}\varphi$.

HCL uses a *general* or *Henkin* semantics which approximates the standard semantics used by HCL* for higher-order logic. Henkin semantics is weaker than the standard semantics. In general $\models_H \varphi$ implies $\models \varphi$. This is in large part due to restriction of second-order quantification solely to definable sets which is a prerequisite for showing completeness of the proof system associated with Henkin semantics. The standard semantics is not restricted to definable sets and in HCL*, second-order quantifiers range over *all* relations of respective arity, which directly reflects intuitions behind them. On the other hand, HCL* is undecidable. However, in the context of model-checking, when a given finite structure is fixed and the language includes equality ($=$) as well as constants denoting domain elements, then every set becomes definable and both semantics become compatible in the sense that for any finite model M, $M \models_H \varphi$ iff $M \models \varphi$. Note that the required constants and equality is always available given the unique names and closed world assumptions.

HCL* permits quantification over relation variables of any arity, not only monadic ones, as required in HCL. Due to this, HCL* provides increased expressivity. For example, the following HCL* formula is outside of the HCL syntax:

$$\begin{aligned}&\forall u\forall X\big((\forall x\forall y(X(x,y) \rightarrow Cn(x,y))\wedge\\ &\forall x\forall z((Cp(x,z) \vee \exists y(X(x,y) \wedge X(y,z))) \rightarrow X(x,z)))\\ &\rightarrow \langle\{y\}\exists x(X(x,y) \wedge S(x))\rangle W(u)\big).\end{aligned} \quad (6)$$

If, for example, Cn stands for "controls", Cp for "being able to cooperate", S for "smart" and W for "wins" then (6) states that

> for every agent u, there is a coalition formed from agents that are able to cooperate with one another and are controlled transitively by smart agents, that can make u a winner,

where "controlled transitively" is formalized by a transitive X containing relation Cp and contained in Cn.

On the other hand, the box operator, $[\{x\}\psi]$, while included in HCL syntax, is not part of HCL* syntax. The box operator, $[\{x\}\psi]$, is definable by means of the diamond $\langle\{x\}\psi\rangle$ operator (see [3]). However, using such definitions results in an exponential blow up in the length of formulas. In the context of model-checking and quantifier elimination, dealing with formulas which include the box operator directly is problematic, as they are defined by a formula using the sequence of quantifiers $\forall\exists\forall$. The first two alternating quantifiers binding relational variables cause substantial technical problems.

HCL* restricts frames to those whose effectivity function has the property of monotonicity (both outcome and coalition monotonic). Observe that playability implies both conditions: outcome monotonicity (by definition of playability) and coalition monotonicity (by Lemma 2). HCL considers frames whose effectivity functions have the property of being weakly playable. If one considers weak playability only, outcome monotonicity is assumed by definition, however one has to additionally assume the coalition monotonicity property.

5. REPRESENTATION OF MODELS

Querying deductive databases and model-checking are very similar. When querying a deductive database, we can view the database as a model and the query as a formula which is being checked for satisfaction relative to the database. We will in fact take advantage of this analogy when doing model-checking in HCL*. Since functions are typically not allowed in deductive databases, we will equivalently replace effectivity functions $\mathcal{E}$ by *effectivity relations*:

$$E \subseteq \mathcal{P}(Ag) \times S \times \mathcal{P}(S) \quad (7)$$

such that $E(C, s, T) \stackrel{\text{def}}{\equiv} T \in \mathcal{E}(C, s)$. This representation then views a model frame in HCL* as a deductive database containing the relation E and model checking as satisfying a query relative to that database. One can then study the complexity of model checking relative to the language fragments of HCL* used in the query language by using results from deductive database theory and descriptive complexity. Observe that one can identify any set with its characteristic relation, i.e., rather than using set X, we may use the unary relation $X(x) \stackrel{\text{def}}{\equiv} x \in X$.

To simplify the presentation, coalition models will be represented in a deductive database using the relation $E()$ defined above, but more succinct representations are also possible, as discussed in the end of this section. The extensional part will contain facts represented as $E()$ atoms and the intensional part will contain a rule encapsulating monotonicity assumptions:

$$(E(X, x, Y) \wedge X \subseteq X' \wedge Y \subseteq Y') \rightarrow E(X', x, Y'). \quad (8)$$

This, in fact, ensures a succinct representation of coalition models when doing model checking. Since we assume that coalition models are monotonic, we do not have to include information that follows from monotonicity. The same representation is used in [19] for outcome monotonicity only. Our representation of the effectivity relation is more succinct, since we also use coalition monotonicity. This may result in exponentially smaller representations. For example, if $E(\{a\}, s, \{s\})$ holds, we do not have to include an exponential number of facts of the form $E(C, s, \{s\})$ for all C with $a \in C$ in the model. Consequently, we avoid the problem of explicit model checking criticized elsewhere in the literature (e.g., [3]). It is important to note that rule (8) is not intended to generate a possibly exponential number of facts. It is only used to

[3]In this definition, we restrict models to a single sort for agents, but our results are also valid for many-sorted structures.

reduce the size of models and to model check facts involving $E()$ literals as is demonstrated in the proof of Lemma 6.

We can also increase succinctness of the model representation by applying the Closed World Assumption, allowing one not to list negative facts. Using this approach, model checking then becomes similar to querying deductive databases (see, e.g., [1]). The following example illustrates the representation used for coalition models.

EXAMPLE 3 (EXAMPLE 1 CONTINUED). *The model considered in the introductory example consists of the following facts:*

$$E\big(\{a\}, s_0, \{s_s\}\big), E\big(\{e\}, s_0, \{s_s, s_j\}\big), E\big(\{j\}, s_0, \{s_s, s_e\}\big)$$
$$E\big(\{a, e\}, s_0, \{s_s\}\big), E\big(\{a, e\}, s_0, \{s_e\}\big)$$
$$E\big(\{a, j\}, s_0, \{s_s\}\big), E\big(\{a, j\}, s_0, \{s_j\}\big), E\big(\{e, j\}, s_0, \{s_s\}\big).$$

Recall that the rule (8) reflecting the monotonicity of E is in the intensional part of the database. Observe that due to (8), one can remove facts $E\big(\{a, e\}, s_0, \{s_s\}\big)$ and $E\big(\{a, j\}, s_0, \{s_s\}\big)$.

Also, no matter how many other agents and states are involved, the above model of this particular scenario does not need to be extended. □

Using our representation, the *size of a coalition frame* $\mathcal{F} = \langle Ag, S, \mathcal{E}\rangle$ is given by:

$$|\mathcal{F}| \stackrel{\text{def}}{=} \max\Big\{|Ag|, |S|, \sum_{E(C,s,X)\in\mathcal{E}} \big(|C| + |X| + 1\big)\Big\}. \quad (9)$$

Complexity results will be provided w.r.t. size of models. The input or query formula is considered to be fixed, thus has a constant length. This is standard practice. Observe that the size of models can, in the worst case, be exponential w.r.t. both $|Ag|$ and $|S|$.

In this context, what do we mean by *succinct* representation of models? Our claim is that a large class of models used in practical applications can be succinctly represented by leveraging formal results from deductive database theory. Recall that we represent an effectivity function as a relation $E()$ and then represent that relation (usually defined in terms of atoms) in a deductive database with an intensional rule for monotonicity. In fact, one can use any equivalent formula in first-order fixpoint logic to represent the effectivity relation. This formula may include fixpoints, quantifiers and relations other than $E()$. Additionally, we can use other intensional rules in addition to the monotonicity rule.

Why is this fundamentally important? Well, any model that is polynomial in the size of agents and states can be equivalently represented as a fixpoint formula and this fixpoint formula can be polynomially compiled into a deductive database. The tractability of model checking obviously applies to this class of models, too.[4] Let's illustrate this idea with the following Example 4.

EXAMPLE 4. *Consider n sax players, m bass players and k drummers ($n, m, k \geq 1$). To organize a concert, one needs at least a trio consisting of a sax player, a bass player and a drummer. Let s_0 be the initial state, s_c be the state where a concert is possible and s_n where it is not. Let $S(x), B(x)$ and $D(x)$ stand for "x is a sax player", "x is a bass player" "x is a drummer", respectively. Then in this model we need $n + m + k$ facts:*[5]

$$\{S(s) \mid s \text{ is a sax player}\} \cup \{B(b) \mid b \text{ is a sax player}\} \cup$$
$$\{D(d) \mid d \text{ is a drummer}\},$$

in addition to facts reflecting that coalitions consisting of all sax players (of all bass players or of all drummers) have the power to block the concert:

$$E(S, s_0, \{s_n\}), E(B, s_0, \{s_n\}), E(D, s_0, \{s_n\})$$

as well as rule (8) and the following intensional rule expressing that any suitable trio makes the concert possible:

$$(S(x)\wedge C(x)\wedge B(y)\wedge C(y)\wedge D(z)\wedge C(z)) \rightarrow E(C, s_0, \{s_c\}). \quad (10)$$

Note that the size of the model is $O(n+m+k)$ (and after unwinding rule (10), it is $O(n+m+k+n*m*k)$) rather than $O(2^{n+m+k})$, when rules (8) and (10) are not used. □

To our knowledge, these techniques for reducing the size of models resulting in succinct representations is novel and quite powerful. It also shows how the integration of the model-checking techniques developed in this paper together with deductive database techniques results in an expressive and efficient representational technique. Additionally, one has a more formal characterization of what is meant by *succinct* representation.

6. MODEL CHECKING

When checking satisfiability of a formula from HCL*, we do this relative to a model and a state. Given an arbitrary formula in HCL*, we will introduce a translation operator Tr which maps each formula into another second-order formula in HCL*. This operator has two purposes.

1. It parameterizes all relational predicates in the formula with an additional state argument.
2. It translates any instance of the diamond modality into a second-order formula which is equivalent.

The net result is that a query is now reduced to an arbitrary second-order formula without modalities in HCL* whose satisfiability we would like to check relative to a coalition model. Transforming the model checking problem into the problem of a 2nd-order query to a deductive database representing a coalition model has great advantages. We can now isolate fragments of second-order logic which, through the use of quantifier elimination reduce the problem to a 1st-order or fixpoint query on a relational database. Results from deductive database theory ensure us that this can be done efficiently relative to the size of the database which we know contains a succinct representation of a coalition model due to the advantageous use of the monotonicity constraint.

We now provide the translation operator *Tr*. The translation *Tr*(φ, s) results in a formula expressing the fact that formula φ is satisfied in state s. To define *Tr*, with every k-argument symbol like F, X of the HCL* language we associate respectively a fresh $(k+1)$-argument symbol F', X' not appearing in the original HCL* language:

- *Tr*$(F(x_1, \ldots, x_k), s) \stackrel{\text{def}}{=} F'(s, x_1, \ldots, x_k)$
- *Tr*$(\neg\varphi, s) \stackrel{\text{def}}{=} \neg$*Tr*$(\varphi, s)$
- *Tr*$(\varphi \vee \psi, s) \stackrel{\text{def}}{=}$ *Tr*$(\varphi, s) \vee$ *Tr*(ψ, s)
- *Tr*$(X(x_1, \ldots, x_k), s) \stackrel{\text{def}}{=} X'(s, x_1, \ldots, x_k)$
- *Tr*$(\forall x\varphi, s) \stackrel{\text{def}}{=} \forall x$*Tr*$(\varphi, s)$
- *Tr*$(\forall X\varphi, s) \stackrel{\text{def}}{=} \forall X$*Tr*$(\varphi, s)$
- *Tr*$(\langle\{x\}\psi\rangle\varphi, s) \stackrel{\text{def}}{=} \exists X\big(\forall x(X(x) \equiv$ *Tr*$(\psi, s)) \wedge$
  $\exists Y(E(X, s, Y)) \wedge \forall y(Y(y) \rightarrow$ *Tr*$(\varphi, y))\big)$.

[4] In fact, any equivalent representation of the class of fixpoint formulas such as stratified Datalog would also do as a representational mechanism.

[5] We implicitly assume that all players are different. For example, no sax player is at the same time a bass player, etc.

Observe that the last clause of the *Tr* operator above translates any instance of the diamond operator into an equivalent second-order formula. The following important lemma allows us to replace these translations of the diamond operator in a query formula with a more efficient but equivalent query about $E()$ without second-order quantifiers.

LEMMA 5 (DIAMOND ELIMINATION LEMMA). *For every coalition model* $\mathcal{M} = \langle Ag, S, \mathcal{E}, \mathcal{I}, \sigma\rangle$ *and* $s \in S$,

$$\mathcal{M}, s \models \mathit{Tr}(\langle\{x\}\psi\rangle\varphi, s) \equiv E(\{x\}\mathit{Tr}(\psi, s), s, \{y\}\mathit{Tr}(\varphi, y)).$$

PROOF.
($\rightarrow$) Assume that $\mathcal{M}, s \models \mathit{Tr}(\langle\{x\}\psi\rangle\varphi, s)$. By definition,

$$\mathcal{M}, s \models \exists X\big(\forall x(X(x) \equiv \mathit{Tr}(\psi, s)) \wedge \exists Y(E(X, s, Y)) \wedge \forall y(Y(y) \rightarrow \mathit{Tr}(\varphi, y))\big),$$

In particular, $\mathcal{M}, s \models \exists X\forall x(X(x) \rightarrow \mathit{Tr}(\psi, s))$. By monotonicity of E we have that $\mathcal{M}, s \models E(\{x\}\mathit{Tr}(\psi, s), s, \{y\}\mathit{Tr}(\varphi, y))$.

($\leftarrow$) Assume that $\mathcal{M}, s \models E(\{x\}\mathit{Tr}(\psi, s), s, \{y\}\mathit{Tr}(\varphi, y))$. Let $X(x) \stackrel{\text{def}}{\equiv} \mathit{Tr}(\psi, s)$ and $Y(y) \stackrel{\text{def}}{\equiv} \mathit{Tr}(\varphi, y)$. Such X and Y obviously satisfy

$$\mathcal{M}, s \models \forall x(X(x) \equiv \mathit{Tr}(\psi, s)) \wedge E(X, s, Y) \wedge \forall y(Y(y) \rightarrow \mathit{Tr}(\varphi, y)).$$

Therefore,

$$\mathcal{M}, s \models \exists X\big(\forall x(X(x) \equiv \mathit{Tr}(\psi, s)) \wedge \exists Y(E(X, s, Y)) \wedge \forall y(Y(y) \rightarrow \mathit{Tr}(\varphi, y))\big),$$

so, by definition of *Tr*, $\mathcal{M}, s \models \mathit{Tr}(\langle\{x\}\psi\rangle\varphi, s)$, which completes the proof. □

Given this lemma and the following lemma, we can already show that formulas in the fragment of HCL* containing arbitrary instances of the diamond operator, but no other 2nd-order quantifiers can be model-checked for satisfiability in PTIME.

LEMMA 6. *Model checking for formulas without second-order quantifiers is in* PTIME.

PROOF. Let $\mathcal{M}$ be a coalition model and s a state in $\mathcal{M}$. We first eliminate diamonds from the input formula.

Checking whether $\mathcal{M}$ is a model for a formula without occurrences of effectivity relation can be done in polynomial time in the standard way (see, e.g., [1, 18]).

Checking the truth value of a given expression of the form $E\big(\{x\}\mathit{Tr}(\psi, s), s, \{y\}\varphi(y)\big)$ can be done by traversing facts in the model and checking whether there is a fact $E\big(C, s, T\big)$ such that $\mathcal{M}, s \models E\big(C, s, T\big) \rightarrow E\big(\{x\}\mathit{Tr}(\psi, s), s, \{y\}\varphi(y)\big)$. Such a fact exists iff $\mathcal{M}, s \models E\big(\{x\}\mathit{Tr}(\psi, s), s, \{y\}\varphi(y)\big)$. To check the required implication we use monotonicity: we simply check whether:

- the set C is included in the set being the value of $\{x\}\mathit{Tr}(\psi, s)$ in $\mathcal{M}$ and s
- the set T is included in the set being the value of $\{y\}\varphi(x)$ in $\mathcal{M}$ and s.

Computing the sets $\{x\}\mathit{Tr}(\psi, s)$ and $\{y\}\varphi(x)$ can be done in polynomial time by an obvious extension of the technology of querying databases in logic (see, [1]). □

Let us now assume that our queries use both the diamond operator and additional 2nd-order quantifiers. Our next task is to identify additional fragments of HCL* where these additional quantifiers can be eliminated. Any formulas in such fragments are then guaranteed to be amenable to model-checking in PTIME based on the results which follow.

Since universal quantifiers are definable by existential ones (using the standard definition $\forall = \neg\exists\neg$), in what follows we will focus on the existential fragment of HCL* without any loss in expressivity. The *existential fragment* of HCL* is the smallest set containing arbitrary HCL* formulas without any universal quantifiers, formulas of the form

$$\exists X_1 \ldots \exists X_r\varphi, \tag{11}$$

where φ contains no second-order quantifiers, and which is closed under Boolean connectives, first-order quantifiers and modalities.

The first fragment of well-behaved formulas will be those that are positive. By the *positive fragment* of HCL*, we mean formulas in the existential fragment with the additional restriction that for formulas of the form (11), φ is positive w.r.t. all relation variables $X_1 \ldots X_r$. The standard definition of positive formulas [13] will have to be slightly modified since relations such as the effectivity relation E have arguments that might be formulas.

An occurrence of a relation variable X is *positive* (*negative*) in φ, if it appears under an even (respectively, odd) number of negation signs.[6] A formula φ is *positive* (*negative*) w.r.t. X if all occurrences of X in φ are positive (respectively, negative).

For example, formula

$$\neg X(a) \vee Y(b) \vee \neg E\big(\{x\}X(x), s, \{y\}\neg Y(y)\big)$$

is negative w.r.t. X and positive w.r.t. Y.

We could deal with the monotonic fragment of HCL* instead. The reason we do not is that in general, checking monotonicity or down-monotonicity is not decidable, while checking positivity and negativity can be done in time linear in the length of the considered formula. Since positivity implies monotonicity and negativity implies down-monotonicity, it is algorithmically convenient to use positivity and negativity rather than monotonicity.

We now have the following lemma.

LEMMA 7. *Model checking for the positive fragment of* HCL* *is in* PTIME.

PROOF. In light of Lemma 6 it suffices prove the claim for formulas of the form (11), where φ is positive w.r.t. $X_1 \ldots X_r$. Consider φ' obtained from φ by replacing all occurrences of $X_1 \ldots X_r$ by $\top$. It appears that φ' is equivalent to formula $\exists X_1 \ldots \exists X_r\varphi$. To show this, consider a coalition model $\mathcal{M}$ and its state s.

($\rightarrow$) If $\mathcal{M}, s \models \varphi'$ then there are $X_1 \ldots X_r$ such that $\mathcal{M}, s \models \exists X_1 \ldots \exists X_r\varphi$ (it suffices to define all of them to be $\top$).

($\leftarrow$) Assume that $\mathcal{M}, s \models \exists X_1 \ldots \exists X_r\varphi$. Formula φ is positive w.r.t. $X_1 \ldots \exists X_r$, so monotone w.r.t. $X_1 \ldots \exists X_r$.[7] Since for each $1 \leq i \leq r$, formula $X_i(\ldots) \rightarrow \top$ is a tautology, by monotonicity we get that $\mathcal{M}, s \models \varphi'$. □

The final fragment of well-behaved formulas we will focus on are the semi-Horn formulas. By the *semi-Horn fragment* of HCL* we mean formulas in the existential fragment of HCL* which have the following form:

$$\exists\bar{X}\big\{\forall\bar{x}[\alpha(\bar{X}, \bar{x}, \bar{z}) \rightarrow X_i(\bar{x})] \wedge \beta(\bar{X})\big\}$$

where $\bar{X}$ stands for $X_1, \ldots, X_r$, $1 \leq i \leq r$, α is positive w.r.t. each of $X_1, \ldots, X_r$ and β is negative w.r.t. each of $X_1, \ldots, X_r$.

We will now show that semi-Horn formulas in the existential fragment can be reduced to logically equivalent fixpoint formulas without higher-order quantifiers and that such formulas can be

[6] Here, as usual, each implication $\varphi \rightarrow \psi$ is replaced by $\neg\varphi \vee \psi$ and each equivalence $\varphi \equiv \psi$ is replaced by $(\neg\varphi \vee \psi) \wedge (\neg\psi \vee \varphi)$.
[7] Monotonicity of effectivity relations is used here, too.

model-checked in PTIME. To show tractability of model checking for the semi-Horn fragment of HCL*, we will use the following theorem from [14] (see also [13]), where by LFP $X(\bar{x})\ [\alpha(X,\bar{x},\bar{z})]$ and GFP $X(\bar{x})\ [\alpha(X,\bar{x},\bar{z})]$ we denote the least and the greatest fixpoint of $\alpha(X,\bar{x},\bar{z})$, i.e., the least and the greatest (w.r.t. inclusion) relation satisfying $X(\bar{x}) \equiv \alpha(X,\bar{x},\bar{z})$.[8] For a detailed discussion of fixpoint calculus and their use in databases see, e.g., [1, 18]. For our purposes it is important to show that fixpoint queries are computable in PTIME w.r.t. size of the database (in our case w.r.t. size of the model).

The following additional notation will be used in the theorem and proof. Let $\alpha(x,\bar{y})$ be a higher-order formula, $X(\bar{x})$ be a (higher-order) relation, $\gamma(\bar{x})$ be a (higher-order) formula with all free variables being $\bar{x}$. Then $\alpha^{X(\bar{x})}_{\gamma(\bar{x})}$ denotes the formula obtained from α by substituting all subformulas of the form $X(\bar{t})$ by $\gamma(\bar{t})$.

THEOREM 8. *Let X be a relation variable and $\alpha(X,\bar{x},\bar{z})$, $\beta(X)$ be formulas with relations of arbitrary order, where the number of distinct variables in $\bar{x}$ is equal to the arity of X. Let α be monotone w.r.t. X.*

If $\beta(X)$ is down-monotone w.r.t. X then

$$\exists X\big\{\forall\bar{x}[\alpha(X,\bar{x},\bar{z}) \rightarrow X(\bar{x})] \wedge \beta(X)\big\} \equiv \beta(X)^{X(\bar{x})}_{\text{LFP}\, X(\bar{x})\ [\alpha(X,\bar{x},\bar{z})](\bar{x})}. \tag{12}$$

If $\beta(X)$ is monotone w.r.t. X then

$$\exists X\big\{\forall\bar{x}[X(\bar{x}) \rightarrow \alpha(X,\bar{x},\bar{z})] \wedge \beta(X)\big\} \equiv \beta(X)^{X(\bar{x})}_{\text{GFP}\, X(\bar{x})\ [\alpha(X,\bar{x},\bar{z})](\bar{x})}. \tag{13}$$

The following example illustrates the use of Theorem 8

EXAMPLE 9. *Consider formula (6). It is universally quantified, so we first negate it to replace universal quantifiers by existential quantifiers:*

$$\begin{array}{l}\neg\exists u\exists X\big(\forall x\forall y(X(x,y) \rightarrow Cn(x,y)) \wedge \\ \forall x\forall z((Cp(x,z) \vee \exists y(X(x,y) \wedge X(y,z))) \rightarrow X(x,z)) \wedge \\ \neg\langle\{y\}\exists x(X(x,y) \wedge S(x))\rangle\, W(u)\big).\end{array} \tag{14}$$

Formula under $\exists u$ is semi-Horn. To apply our method we first have to translate the formula using translation Tr and applying Lemma 5. The result is:

$$\begin{array}{l}\neg\exists u\exists X'\big(\forall x\forall y(X'(s,x,y) \rightarrow Cn'(s,x,y)) \wedge \\ \forall x\forall z((Cp'(s,x,y) \vee \exists y(X'(s,x,y) \wedge X'(s,y,z))) \rightarrow X'(s,x,z)) \\ \wedge\ \neg E(\{y\}\exists x(X'(s,x,y) \wedge S'(s,x)), s, W'(s,u))\big).\end{array}$$

To apply the equivalence (12) we formally need a small trick,[9] *namely the second line of the above formula is equivalent to*

$$\forall t\forall x\forall z\big((t = s \wedge (Cp'(s,x,y) \vee \exists y(X'(s,x,y) \wedge X'(s,y,z)))) \rightarrow X'(t,x,z)\big)$$

Now we apply equivalence (12) of Theorem 8 and obtain the following equivalent formula:

$$\begin{array}{l}\neg\exists u\big(\forall x\forall y(X'(s,x,y) \rightarrow Cn'(s,x,y)) \wedge \\ \neg E(\{y\}\exists x(X'(s,x,y) \wedge S'(s,x)), s, W'(s,u))\big),\end{array} \tag{15}$$

where X' should be respectively replaced by

$$\text{LFP}\, X'(t,x,z)\ [t = s \wedge (Cp'(s,x,z) \vee \exists y(X'(s,x,y) \wedge X'(s,y,z)))]. \tag{16}$$

[8]We shall only use this notation in contexts where the least and the greatest relation exist.

[9]Which later will appear reversible.

Using the fact that $t = s$, we get the following equivalent of (16):[10]

$$\text{LFP}\, X'(s,x,z)\ [Cp'(s,x,z) \vee \exists y(X'(s,x,y) \wedge X'(s,y,z))]. \tag{17}$$

Formula (16), in which X's are respectively replaced by the least fixpoint formula (17), is the input to the model checking method. □

Since positivity implies monotonicity and negativity implies down-monotonicity, we have the following theorem as a consequence of equivalence (12) from Theorem 8.

THEOREM 10. *Model checking for the semi-Horn fragment of* HCL* *is in* PTIME.

PROOF. Observe that second-order quantifiers can be eliminated from semi-Horn formulas using (12). The resulting formula is a fixpoint formula. Checking whether it holds in a given model $\mathcal{M}$ can be done in time polynomial w.r.t. the size of $\mathcal{M}$.

Note that in Theorem 8 second-order quantification binds a single relation variable, while in semi-Horn formula there might be a longer tuple of existential quantifiers. However, such a tuple can be encoded by a single relation variable by adding a special argument (or a number of arguments) identifying "original" relations. For example, to encode $X_1,\ldots,X_r$, we can consider a relation variable $X(\hat{i},\bar{x})$, where $\hat{i}$ is the special argument, $\bar{x}$ is the list of arguments of the length being the maximum of lengths of arguments of $X_1,\ldots,X_r$. Now, rather than writing $X_i(\bar{x}_i)$ one can write $X(\hat{i},\bar{x}_i,\bar{y})$, where $\bar{y}$ is a tuple of dummy arguments, needed when the number of arguments of X_i is smaller than the number of arguments in $\bar{x}$. □

One can also define the dual form of semi-Horn formulas. By the *dual semi-Horn fragment* of HCL* we mean formulas in the existential fragment of HCL*, which have the following form:

$$\exists\bar{X}\big\{\forall\bar{x}[X_i(\bar{x}) \rightarrow \alpha(\bar{X},\bar{x},\bar{z})] \wedge \beta(\bar{X})\big\}$$

where $\bar{X}$ stands for $X_1,\ldots,X_r$, $1 \le i \le r$ and α,β are both positive w.r.t. each of $X_1,\ldots,X_r$.

By applying the equivalence (13) from Theorem 8, we have the following theorem.

THEOREM 11. *Model checking for the dual semi-Horn fragment of* HCL* *is in* PTIME. □

7. ASSERTION TYPES EXPRESSIBLE IN TRACTABLE FRAGMENTS OF HCL*

Let us summarize a number of useful types of assertions which can be represented in those fragments of HCL* which admit tractable model checking.

The first, obvious class of expressible formulas is provided by Lemma 6. This is quite a rich class of formulas. Probably the most interesting among them are *existence assertions* allowing one to express that, in a given circumstance $C(\bar{x})$, there is a coalition satisfying a certain condition A which can lead to a set of states guaranteeing that a given goal $G(\bar{z})$ is achieved:

$$C(\bar{x}) \rightarrow \langle\{y\}A(y)\rangle\ G(\bar{z}). \tag{18}$$

Note that both C and G can still contain diamonds. In applications, one can frequently expect queries of the form (18), often simplified to the case where $C(\bar{x})$ is true, i.e., when one is interested whether in a given situation there is a coalition able to achieve a given goal (i.e., $\langle\{y\}A(y)\rangle\ G(\bar{z})$).

[10]Explaining what we have meant by "reversibility" of the trick applied earlier.

Another significant class of formulas is provided by Lemma 7 together with Theorems 10 and 11. A very important subclass of such formulas is the one, where using existential second-order quantifiers, one can "transfer" coalitions among diamonds. We call such assertions *transfer assertions*, which typically can take the form

$$\exists X\big(\forall \bar{x}(A(X,\bar{x}) \rightarrow \langle\{y\}B(X,\bar{x},y)\rangle) \wedge C(X)\big). \qquad (19)$$

Of course, tractable model checking is possible when such a formula translates into a positive or (dual) semi-Horn formula, like in the following example,

$$\exists X\big(\forall \bar{x}(X(\bar{x}) \rightarrow (\langle\{y\}(X(y) \vee large(y))\rangle goal \wedge \\ \forall \bar{x}(strong(x) \rightarrow X(\bar{x}))\big),$$

expressing that there is a coalition consisting of all $strong$ agents in addition to possibly some $large$ agents, capable of achieving the $goal$.

Observe that the class of formulas which admit tractable model checking using the methods provided in this paper is not limited to the above types of assertions, but these assertion types do show the practical use of the fragments we deal with.

8. CONCLUSIONS

We have introduced the higher-order logic HCL* which can be used for reasoning about the abilities of coalitions and interactions between them. HCL* is a generalization of HCL which subsumes both CL and QCL. Additionally, we have isolated a number of expressive fragments of HCL* and shown that the model-checking problem for these fragments can be solved in PTIME by appealing to use of quantifier elimination, results from deductive database theory and descriptive complexity. Additionally, through advantageous use of monotonicity constraints on coalition frames and use of deductive database techniques one can often get exponentially more succinct representations of coalition models in the model-checking process.

For formulas outside of this fragment one could use extensions of the second-order quantifier elimination algorithm of [11] which, although often finding reductions outside these fragments, does not guarantee such reductions. For a presentation of this algorithm as well as other relevant techniques see also [13].

9. REFERENCES

[1] S. Abiteboul, R. Hull, and V. Vianu. *Foundations of Databases*. Addison-Wesley Pub. Co., 1996.

[2] T. Ågnotes, W. van der Hoek, and M. Wooldridge. On the logic of coalitional games. In *Proc. AAMAS'06*, pages 153–160. AAAI, 2006.

[3] T. Ågnotes, W. van der Hoek, and M. Wooldridge. Quantified Coalition Logic. In *Proc. IJCAI'07*, pages 1181–1186. AAAI, 2007.

[4] T. Ågotnes and H. van Ditmarsch. Coalitions and announcements. In *Proc. AAMAS '08*, pages 673–680, 2008.

[5] R. Alur, T.A. Henzinger, and O. Kupferman. Alternating-time Temporal Logic. *Journal of the ACM*, 49:672–713, 2002.

[6] P. Balbiani, O. Gasquet, A. Herzig, F. Schwarzentruber, and N. Troquard. Coalition games over Kripke semantics. In C. Dégremont, L. Keiff, and H. Rückert, editors, *Dialogues, Logics and Other Strange Things – Essays in Honour of Shahid Rahman*, pages 11–32. College Publications, 2008.

[7] G. Boella, D.M. Gabbay, V. Genovese, and L. van der Torre. Higher-Order Coalition Logic. In *Proc. ECAI'10*, pages 555–560, 2010.

[8] J. Broersen, A. Herzig, and N. Troquard. From coalition logic to STIT. *Electron. Notes Theor. Comput. Sci.*, 157(4):23–35, 2006.

[9] N. Bulling, J. Dix, and C.I. Chesnevar. Modelling coalitions: ATL + argumentation. In *AAMAS '08*, pages 681–688, 2008.

[10] V. Dignum, editor. *Handbook of Research on Multi-Agent Systems*. Information Science Reference, 2009.

[11] P. Doherty, W. Łukaszewicz, and A. Szałas. Computing circumscription revisited. *Journal of Automated Reasoning*, 18(3):297–336, 1997.

[12] B. Dunin-Keplicz and R. Verbrugge. *Teamwork in Multi-Agent Systems: A Formal Approach*. John Wiley & Sons, Ltd., 2010.

[13] D.M. Gabbay, R. Schmidt, and A. Szałas. *Second-Order Quantifier Elimination. Foundations, Computational Aspects and Applications*, volume 12 of *Studies in Logic*. College Publications, 2008.

[14] D.M. Gabbay and A. Szałas. Second-order quantifier elimination in higher-order contexts with applications to the semantical analysis of conditionals. *Studia Logica*, 87:37–50, 2007.

[15] A. Gibbard. A Pareto-consistent libertarian claim. *Journal of EconomicTheory*, 7(4):388–410, 1974.

[16] V. Goranko. Coalition games and alternating temporal logics. In *Proc. 8th Conf. on Theoretical Aspects of Rationality and Knowledge TARK*, pages 259–272. Morgan Kaufmann, 2001.

[17] S. Ieong and Y. Shoham. Marginal contribution nets: A complact representation scheme for coalitional games. In *Proc. ACM EC*, pages 170–179, 2006.

[18] N. Immerman. *Descriptive Complexity*. Springer, 1998.

[19] M. Pauly. *Logic for Social Software*. Ph.D., ILLC Dissertation Series. University of Amsterdam, 2001.

[20] M. Pauly. A modal logic for coalitional power in games. *Journal of Logic and Computation*, 12(1):149–166, 2002.

[21] T. Rahwan and N. R. Jennings. An improved dynamic programming algorithm for coalition structure generation. In *Proc. AAMAS'08*, pages 1417–1420, 2008.

[22] T. Rahwan, T. Michalak, N. R. Jennings, M. Wooldridge, and P. McBurney. Coalition structure generation in multi-agent systems with positive and negative externalities. In *Proc. IJCAI*, 2009.

[23] T. Sandholm, K. Larson, M. Andersson, O. Shehory, and F. Tohme. Coalition structure generation with worst case guarantees. *AIJ*, 1-2(111):209–238, 1999.

[24] T. Sandholm and V.R. Lesser. Coalitions among computationally bounded agents. *Artificial Intelligence*, 94:99–137, 1997.

[25] İ. Seylan and W. Jamroga. Description logic for coalitions. In *Proc. AAMAS'09*, pages 425–432. AAAI, 2009.

[26] Y. Shoham and K. Leyton-Brown. *Multi Agent Systems: Algorithmic, Game-Theoretic, and Logical Foundations*. Cambridge, 2009.

[27] F. Tohmé and T. Sandholm. Coalition formation processes with belief revision among bounded-rational self-interested agents. *Journal of Logic and Computation*, 9:793–815, 1999.

[28] W. van der Hoek and M. Wooldridge. On the logic of cooperation and propositional control. *Artif. Intell.*, 164(1-2):81–119, 2005.

[29] J. Wu, C. Wang, L. Zhang, and J. Xie. Coalitional planning in game-like domains via ATL model checking. In *ICTAI '09: Proc. 21st Int. Conf. on Tools with AI*, pages 645–652, 2009.