MODELLING MULTI-AGENT INTERACTION PROTOCOLS USING CATEGORIES ENRICHED OVER POINTED SETS

MARK W. JOHNSON, TIM MILLER, AND PETER MCBURNEY

Abstract. The study of multi-agent systems is proving to be an exciting and active research field within computer science. Communication is an important aspect of multi-agent systems, and the protocols that are used to provide a shared definition of what agents can say to each other, along with the meaning of what agents say, receive much attention from researchers. At present, there are a number of well-defined protocols, agent communication languages, and protocol specification languages available, yet, there is no formal mathematical theory of agent interaction protocols. The authors believe that such a theory would be beneficial to the field of agent communication and multi-agent systems in general. In this paper, we present an application of the mature field of category theory for modelling interaction protocols. Using category theory, specifically, a subsidiary theory of category theory known as enriched category theory, we demonstrate how to model the legal utterances of a protocol, and to assign the meaning of utterances, such as commitments to perform a certain action. Enriched categories obey the rules of categories, so we argue that using them to model protocols gives us a mature theory for modelling and reasoning about protocols, and the relationships between them.

1. Introduction

In recent years, the field of computing has transformed from a view of systems performing computations in isolation, to one in which many interconnected systems work together. The rise of the Internet has helped to promote this view to the current situation, in which we see groups of intelligent agents interacting with one another in dynamic environments using communication languages and protocols.

Considering this, interaction between these agents is an important part of modern computing. For this reason, it is important that suitable theories, frameworks, methodologies, and tools are provided to support the building of systems. As a result, identifying, designing, and studying protocols via which agents interact is an important and active research area within the multi-agent systems field. Researchers and practitioners in the field are designing and implementation protocols, as well as languages for defining them. Standardisation of these protocols and languages has been the focus of the IEEE Foundation for Intelligent Physical Agents standards group (FIPA)[1], for example.

The common view of multi-agent interaction is one of agents making *utterances* or *locutions*, which contain the information they want to communicate, but constraining how and when these locutions can be made via protocols. A string of locutions forms a *dialog* between agents; therefore, a protocol can be viewed as a set containing the legal dialogs. It is important that protocols are defined in an

Date: July 5, 2007.

[1]See `http://www.fipa.org/`.

unambiguous manner to provide a shared understanding of the dialogs permitted by the protocol, as well as the meaning of each of the locutions and of complete dialogs.

The authors believe that, in the same way that computer science benefits from Alan Turing's abstract model of computation, Turing Machines, interaction can benefit from an abstract framework for modelling protocols. By abstract, we mean that the framework should be independent of the implementation detail of the protocol, such as the communication language, the nature of the interaction, the topics of discussion, the agents using the protocol, and the type of protocol being defined, as well as independent of the software and hardware platforms used by the agents. Such a definition is useful because if offers a way to study the properties of protocols, either in isolation or in relation to each other, formally and rigorously. As far as the authors are aware, until now there has been only one other attempt to construct a mathematical model of multi-agent protocols — an approach based on finite state machines proposed by Fernandez and Endriss [10]. This work is discussed further in Section 7.

In this paper, we define such a framework using category theory: a mature mathematical theory that abstractly models mathematical objects and the relationships between them. In our framework, the mathematical objects are placeholders representing the points between utterances, while the relationships represent the utterances themselves; therefore, utterances are viewed as relationships between points. The outcomes of interactions, such as commitments to perform some action, are modelled for every dialog using a mature concept in category theory: *functors*, which are mappings between categories. Specifically, we use the subsidiary theory of *enriched category theory*, and model protocols as enriched symmetric monoidal categories.

Using the theory of enriched categories also presents us with a useful tool for reasoning about the relationships between protocols. Our protocols are modelled as *small* categories, and a particular construct in category theory is the category of small categories. Philosophically, this gives us a structure that is the category of protocols: a category in which the objects are protocols and the relationships are between protocols. Studying the relationships between protocols allows us to identify their differences and similarities, with the ultimate objective of characterising like collections of protocols.

A preliminary version of this framework was presented in [17]. Building on that work, we present the mathematical detail necessary in the framework, discuss the implications of the framework, and present several examples of the application of our framework to commonly used protocols.

This paper is structured as follows. Section 2 presents the basic mathematical details behind the framework, with more complex mathematical material relegated to an appendix. Section 3 presents dialog game protocols, which are the type of protocols that primarily we aim to model using the framework. Section 4 presents our approach for modelling dialog game protocols using enriched categories, while Section 5 discusses uses of the framework beyond the definition of protocols themselves. Section 6 presents several examples that we have modelled using this framework — mainly negotiation protocols. Section 7 presents related work, while Section 8 concludes the paper and discusses future work. The appendix provides a description of two other mathematically equivalent presentations of our framework, including the proof that they are equivalent, which does not seem to be well-known.

2. Enriched Categories over Pointed Sets

In this section, we describe the mathematical details of enriching categories over pointed sets, which we use to distinguish legal behaviour from illegal behaviour.

2.1. Categories and Functors.

It is not the aim of this paper to give a detailed description of category theory. Such a description can be found in the literature [18]. However, we give a brief overview of category theory for those readers who may not already be familiar with it.

Definition 2.1. A *category*, $\mathcal{C}$, contains three components:

(1) a class of objects;
(2) a class of arrows, called *morphisms*, between these objects, with each arrow having a head and tail object. $\mathcal{C}(A, B)$ represents the class of all arrows between objects A and B; and
(3) a composition law

$$\mathcal{C}(B,C) \times \mathcal{C}(A,B) \to \mathcal{C}(A,C)$$

specifying that for every object A, B, and C, every arrow f between objects A and B, and every arrow g between B and C, there exists an arrow between A and C. Therefore, for every pair of arrows in which the head object of one is the tail object of the other, we combine them to form a longer arrow, as demonstrated by the following diagram.

(1)
$$\begin{tikzcd} A \arrow[r, "f"'] \arrow[dr, "g\circ f"'] & B \arrow[d, "g"] \\ & C \end{tikzcd}$$

If we have arrows $f : A \to B$ and $g : B \to C$, then we have an arrow, $g \circ f : A \to C$, in which $g \circ f$ represents applying f followed by g. Note that this is written $g \circ f$, not $f \circ g$, as one may expect. This is because if one were to apply f and then g to an object A, it would be written $g(f(A))$, therefore, in category theory one maintains the order of g and f by writing $g \circ f$. The head of $g \circ f$ is the source of f, and the tail is the tail of g. Additionally, a category, $\mathcal{C}$, has the following properties:

(1) for all arrows f, g, and h, $(h \circ g) \circ f = h \circ (g \circ f)$ – that is, composition is associative; and
(2) if A is an object in $\mathcal{C}$, then there exists an arrow $id_A : A \to A$, called the *identity* morphism of A;
(3) for the arrow $f : A \to B$, $f \circ id_A = f = id_B \circ f$

Example 2.2. (1) The partially ordered set $\{0, 1, \ldots, n\}$ may be considered as a category $[n]$ with a unique non-identity map $k \to j$ if and only if $k < j$. The composition law holds due to the properties of $<$.
(2) Any directed graph can be represented as category. Nodes in the graph represent the objects, and paths represent the morphisms. Concatenating the paths to bring about longer paths is equivalent to composition.

Definition 2.3. A *functor*, $F : \mathcal{C} \to \mathcal{D}$, is a structure-preserving function from one category to another such that the following properties hold:

(1) every object, A, in $\mathcal{C}$ has a corresponding object, $F(A)$, in $\mathcal{D}$;
(2) every morphism, $f : A \to B$, in $\mathcal{C}$ has a corresponding morphism, $F(f)$, in $\mathcal{D}$ such that $F(f) = F(A) \to F(B)$ — that is, the head and tail of $F(f)$ are the objects mapped from the head and tail of f;
(3) for any object, A, in $\mathcal{C}$, $F(id_A) = id_{F(A)}$ — that is, the identity morphisms are preserved; and
(4) for composable morphisms f and g in $\mathcal{C}$, $F(g) \circ F(f)$ is a morphism in $\mathcal{D}$, and $F(g \circ f) = F(g) \circ F(f)$.

Example 2.4. (1) A *constant functor*, $F : \mathcal{C} \to \mathcal{D}$, is defined such that for every object A in $\mathcal{D}$, $F(A) = B$, in which B is some fixed object in $\mathcal{D}$, and for every morphism f in $\mathcal{D}$, $F(f) = id_B$.
(2) The *power set* functor, $\mathbb{P} : Set \to Set$, which maps a set to its power set, and for each function $f : X \to Y$ in Set, $\mathbb{P}(f)$ is a map that identifies every subset S of X to $f(S)$.

2.2. Pointed Sets.

Pointed sets are central in our framework, because they form the morphism sets of our categories such that the basepoint plays a special role in the composition law.

Definition 2.5. A *pointed set* is a set, S, with a distinguished element, $s_0 \in S$, called the *basepoint*. A morphism of pointed sets, or pointed map, is a function $f : S \to T$ with $f(s_0) = t_0$.

In our framework, the pointed sets are the sets of morphisms between two objects, and the basepoints in these sets are used to represent the morphisms that result from composing two other morphisms in a way that is not permitted by the protocol. Our basepoints are generically labelled $*$. Therefore, if we want the composition $g \circ f$ to be illegal, then we define the composition law such that it maps to the basepoint $*$.

2.3. The Smash Product of Pointed Sets.

The basis of any enrichment is a symmetric monoidal category, so we must begin by describing the symmetric monoidal structure we want to use on Sets_*, the category whose objects are pointed sets, and whose morphisms are basepoint preserving functions.

The categorical product of Sets_* is simply the Cartesian product with the ordered pair consisting of both basepoints as the basepoint of the product, e.g. $(*, *)$. This does induce a symmetric monoidal structure on Sets_*, but we would prefer a symmetric monoidal closed structure. That means we want to chose a variation of the Cartesian product where the *pointed* mapping set functor $\text{map}_*(Y, ?)$ is a right adjoint to the functor defined by taking the chosen product with Y. Essentially, the adjunction taking place in the pointed category means any *pointed* map

$$X \to \text{map}_*(Y, Z)$$

corresponds to a unique *pointed* map from this new product of X with Y, to Z. In particular, the basepoint of $\text{map}_*(Y, Z)$ is the constant map to the basepoint in Z. Assuming we build the new product from $X \times Y$, this means we must force any ordered pair $(*, y)$ to map to the basepoint of Z. As a consequence of the assumption of symmetry, we conclude that we must collapse the subset of ordered pairs with a basepoint in either component, usually indicated $X \vee Y$.

Definition 2.6. The *smash product* of two pointed sets, X and Y, written $X \wedge Y$, is the quotient of the Cartesian product $Y \times X$ where all pairs containing a basepoint in either component have been collapsed to the basepoint. Symbolically, this is written as $X \wedge Y := (X \times Y)/(X \vee Y)$.

This operation gives the symmetric monoidal closed structure on Sets$_*$ for which we were looking. Notice the unit in this operation is now S^0, the set with two points (one the basepoint) with $\nu_L : S^0 \wedge X \to X$ and the symmetric ν_R the unit bijections.

Given any set C, we will write C_+ for C with a (disjoint) basepoint added. Notice this is a symmetric monoidal functor from Sets to Sets$_*$, which means there is a natural bijection

$$C_+ \wedge D_+ \approx (C \times D)_+.$$

In order to explain the importance of this smash product of pointed sets to our framework, we must first discuss categories enriched over pointed sets.

2.4. **Categories Enriched over Pointed Sets.**

Both our locution categories and our outcome categories will be chosen as categories enriched over pointed sets. This has several advantages over alternative frameworks, and the comparison is studied in more detail in Appendix A. First, the theory of enriched categories is a mature theory, which means one can find the details of any desired construction somewhere in the literature. There are several good standard references for this general theory, including Chapter 6 of [6], and the more complete [18]. On the other hand, enriching over Sets$_*$ is convenient because it is relatively simple to describe (or to work around) and on a functional level may be addressed mostly without reference to the larger machinery of enriched category theory.

Definition 2.7. A small category $\mathcal{C}$ enriched over pointed sets consists of a set of objects, a pointed set $\mathcal{C}(B,C)$ of morphisms for each pair of objects and an (associative) composition law given by a pointed map

(2)
$$\mathcal{C}(B,C) \wedge \mathcal{C}(A,B) \xrightarrow[\mu]{} \mathcal{C}(A,C),$$

together with pointed unit maps $j_B : S^0 \to \mathcal{C}(B,B)$ that yield commutative diagrams

(3)
$$\begin{tikzcd} S^0 \wedge \mathcal{C}(A,B) \arrow[d, "j\wedge 1"'] \arrow[dr, "\nu_L"] & \\ \mathcal{C}(B,B) \wedge \mathcal{C}(A,B) \arrow[r, "\mu"'] & \mathcal{C}(A,B) \end{tikzcd}$$

and

(4)
$$\begin{tikzcd} \mathcal{C}(A,B) \wedge S^0 \arrow[d, "1\wedge j"'] \arrow[dr, "\nu_R"] & \\ \mathcal{C}(A,B) \wedge \mathcal{C}(A,A) \arrow[r, "\mu"'] & \mathcal{C}(A,B). \end{tikzcd}$$

We will say $\mathcal{C}$ is a Sets$_*$-category or an element of Cat$_*$ to indicate an enrichment over pointed sets (always leaving the choice of enrichment implicit since we will never consider more than one for any category). The image of the non-basepoint element of S^0 under the pointed unit map $j_B : S^0 \to \mathcal{C}(B, B)$ will be referred to as the identity map for B.

Example 2.8.

(1) We will let $\mathcal{S}^0$ indicate the element of Cat_* with a single object and just S^0 as the morphism set, with the unit map as composition law.
(2) Given two small categories $\mathcal{C}$ and $\mathcal{D}$ enriched over pointed sets, one can form their "smash product" $\mathcal{C} \wedge \mathcal{D}$. It's object set is the product of the two object sets, and for morphisms $\mathcal{C} \wedge \mathcal{D}((c, d), (c', d')) = \mathcal{C}(c, c') \wedge \mathcal{D}(d, d')$. The pointed composition law is then given by the smash product of the two pointed composition laws, as is the unit map. Notice $\mathcal{S}^0$ of the last example is the unit object of this operation, which we use later to combine various protocols (see section 6, especially the combinatorial auction).
(3) Any ordinary category $\mathcal{D}$ may be enriched over pointed sets by simply adding a basepoint to each morphism set. The composition law ν of $\mathcal{D}$ extends as

$$\mu : \mathcal{D}(B, C)_+ \wedge \mathcal{D}(A, B)_+ \approx (\mathcal{D}(B, C) \times \mathcal{D}(A, B))_+ \to \mathcal{D}(A, C)_+$$

where the first map is the natural bijection and the second map is ν_+. The unit map $j_B : S^0 \to \mathcal{D}(B, B)$ should be defined by sending the non-basepoint to the identity map 1_B. We will write $\mathcal{D}_+$ for this construction.
(4) The partially ordered set $\{0, 1, \ldots, n\}$ may be considered as a category $[n]$ with a unique non-identity map $k \to j$ if and only if $k < j$. If we apply the previous construction, then $[n]_+$ lies in Cat_* and will be particularly important in section 5.
(5) A category with one object enriched over pointed sets consists of an object c and a pointed set $C = \mathcal{C}(c, c)$ together with an associative pointed multiplication $\mu : C \wedge C \to C$ with pointed (two-sided) unit map $j : S^0 \to C$. This will be referred to as a pointed monoid, a concept we expand on in the next example, which sets the stage for our general approach.
(6) Suppose D is a set with a partial multiplication $\nu : M \to D$ for some subset $M \subset D \times D$, which is associative and unital where defined and M contains any ordered pair with the unit in either position. Then we can define a category $\mathcal{D}$ with one object d enriched over pointed sets by taking $\mathcal{D}(d, d) = D_+$ and the composition law should be given by

$$f \circ g = \begin{cases} \nu(f, g), & \text{if } (f, g) \in M; \\ * & \text{otherwise.} \end{cases}$$

Notice $D_+ \vee D_+$ is sent to the basepoint since it does not intersect M, and the unit condition is defined by assumption on M, so this descends to the quotient $(D_+ \times D_+)/(D_+ \vee D_+)$ as the expected composition law $D_+ \wedge D_+ \to D_+$. We view this process as associating a pointed multiplication to any partial multiplication, a precursor of our general process for handling precategories.
(7) Given any category $\mathcal{C}$ enriched over pointed sets, one can build a new category $\mathcal{C}_0$ with one object enriched over pointed sets. Identify all of the objects to a single object, hence all of the unit maps become a single unit map. The composition law is then a partial multiplication which is both

associative and unital, so we can proceed as in the previous example to define the quotient pointed monoid $\mathcal{C}_0$.

In fact, Example 2.8.6 is a special case of our general technique for dealing with undefined compositions. That is, we consider M to be the set of pairs of legal compositions, $\nu(f, g)$ to represent the composition of f and g, and $*$ to represent the illegal composition. Since any pair with an illegal morphism as either component will be treated as the basepoint of the smash product, declaring the composition law in this way enforces the expected condition that composing an illegal morphism with any other morphism also gives us an illegal morphism.

Notice we can also define an ordinary category from any element of Cat_* just by precomposing the pointed composition law by the collapse (or quotient) map

$$\mathcal{C}(B, C) \times \mathcal{C}(A, B) \to \mathcal{C}(B, C) \wedge \mathcal{C}(A, B) \to \mathcal{C}(A, C)$$

and taking the image of the non-basepoint under $j_B : S^0 \to \mathcal{C}(B, B)$ as the identity morphism of the ordinary category. This is known as the underlying category of an enriched category. In the case of Example 2.8.6, the composition law of the underlying category of $\mathcal{D}$ is simply that of M where relevant and the basepoint everywhere else.

As a consequence, we see the basepoints play two roles in our applications. First, the disjoint nature of the added basepoint makes it a convenient place to "send everything else" when defining a composition law. Second, by moving to categories enriched over pointed sets, we encode the requirement that any composition involving an illegal morphism should be illegal itself in the mathematical background, which keeps it out of the way for implementation.

2.5. **Functors Enriched over Pointed Sets.**

Our models for protocols will be arrows in Cat_* or functors enriched over pointed sets. Thus, we should define this term and discuss it.

Definition 2.9. Given $\mathcal{C}$ and $\mathcal{D} \in \mathrm{Cat}_*$, an *enriched functor* from $\mathcal{C}$ to $\mathcal{D}$ will denote an assignment of an object $F(C)$ for each object $C \in \mathcal{C}$ together with pointed maps $F_{A,B} : \mathcal{C}(A, B) \to \mathcal{D}(F(A), F(B))$ that preserve the pointed identity maps (meaning that the following commutes)

(5)
$$\begin{tikzcd} S^0 \arrow[r, "j_B"] \arrow[dr, "j_{F(B)}"'] & \mathcal{C}(B,B) \arrow[d, "F_{B,B}"] \\ & \mathcal{D}(F(B),F(B)) \end{tikzcd}$$

and the composition laws (meaning they make the following commute)

(6)
$$\begin{tikzcd} \mathcal{C}(B,C) \wedge \mathcal{C}(A,B) \arrow[r, "\mu_{\mathcal{C}}"] \arrow[d, "F_{B,C} \wedge F_{A,B}"'] & \mathcal{C}(A,C) \arrow[d, "F_{A,C}"] \\ \mathcal{D}(F(B),F(C)) \wedge \mathcal{D}(F(A),F(B)) \arrow[r, "\mu_{\mathcal{D}}"'] & \mathcal{D}(F(A),F(C)). \end{tikzcd}$$

Example 2.10. (1) An enriched functor from $\mathcal{S}^0$ to $\mathcal{C}$ is simply the choice of some object $C \in \mathcal{C}$ (and its identity map). Together with the unit property discussed in Example 2.8.2, this implies a number of important formal consequences.

(2) If $F : \mathcal{C} \to \mathcal{C}'$ and $G : \mathcal{D} \to \mathcal{D}'$ are functors enriched over pointed sets, then they yield a combined functor $F \wedge G : \mathcal{C} \wedge \mathcal{D} \to \mathcal{C}' \wedge \mathcal{D}'$ by applying the smash product operation of Example 2.8.2 (so this operation is functorial on pairs).

(3) Given an ordinary functor between ordinary categories $F : \mathcal{C} \to \mathcal{D}$, define a functor enriched over pointed sets $F_+ : \mathcal{C}_+ \to \mathcal{D}_+$ as in Example 2.8.3 by simply sending the basepoint morphism to the basepoint morphism and otherwise agreeing with F itself.

(4) Suppose $\mathcal{D} \in \mathrm{Cat}_*$. Then the set of Sets_*-functors from $[n]_+$ (defined in Example 2.8.4) to $\mathcal{D}$ is in natural bijection with the set of ordinary functors from $[n]$ to $\mathcal{D}$, since any Sets_*-functor must preserve basepoints. (See Section 5 for consequences.)

(5) Suppose $\mathcal{C}$ and $\mathcal{D}$ are each categories with one object enriched over pointed sets, called pointed monoids in Example 2.8.5. Then an enriched functor $F : \mathcal{C} \to \mathcal{D}$ is determined by a map of sets $\mathcal{C}(c, c) \to \mathcal{D}(d, d)$ which is compatible with the pointed multiplication determined by the composition law.

(6) Suppose C and D are both sets with associative partial multiplications $\nu : N \to C$ and $\mu : M \to D$, for $N \subset C \times C$ and $M \subset D \times D$ both containing all ordered pairs with the unit in either position. If we have a map $F : C \to D$, which preserves identities and makes

(7)
$$\begin{tikzcd}
N \arrow[r, "(F\times F)|_N"] \arrow[d] & M \arrow[d] \\
C \arrow[r, "F"'] & D
\end{tikzcd}$$

commute, then we can define a functor enriched over pointed sets $\mathcal{F} : \mathcal{C} \to \mathcal{D}$ as in Example 2.8.6 by taking the basepoint morphism to the basepoint morphism and otherwise agreeing with F.

(7) Given $\mathcal{C}$ and the associated "quotient" $\mathcal{C}_0$ as in Example 2.8.7 there is a canonical functor enriched over pointed sets $\mathcal{C} \to \mathcal{C}_0$. The functor identifies all objects and sends each morphism to itself, considered as an endomorphism of the unique object of $\mathcal{C}_0$. This technique has mathematical applications discussed briefly in [15].

It will be important in our model that one can (as in the last example above) extend any Sets_*-functor to one having a pointed monoid (i.e., an element of Cat_* with a single object) as the target.

Unfortunately, the notion of enriched natural transformation can become quite complicated in general, as on page 25 of [18]. However, as a consequence of the fact that the underlying set functor creates the notion of commutative diagram in Sets_*, a natural transformation enriched over pointed sets is simply a natural transformation whose source and target happen to be functors enriched over pointed sets. In other words, given $F,\ G : \mathcal{C} \to \mathcal{D}$ functors enriched over pointed sets, a natural transformation η from F to G consists of maps $\eta_C : F(C) \to G(C)$ which make the diagram

(8)
$$\begin{tikzcd} F(C) \arrow[r, "F(f)"] \arrow[d, "\eta_C"'] & F(D) \arrow[d, "\eta_D"] \\ G(C) \arrow[r, "G(f)"'] & G(D) \end{tikzcd}$$

commute (in Sets$_*$) for any morphism $f : C \to D$ in $\mathcal{C}$. This is important mainly because if we were to adhere to the formal notation of enriched categories, we should instead write $\eta_C : S^0 \to \mathcal{D}(F(C), G(C))$ (with our map the image of the non-basepoint) and write the diagram in a more complex form.

Essentially all of the standard constructions of category theory are available in Cat$_*$, including limits, pushouts, etc. One possibly disturbing thing to notice is that, for general enriched categories, there is also a new type of adjoint (hence a stronger notion of limit) called an enriched adjoint for an enriched functor. However, since the underlying set functor is conservative (i.e. creates the notion of isomorphism) in our case, this is actually equivalent to the usual notion of adjunction when considering functors enriched over pointed sets (see the middle of page 50 of [18]). Hence, we don't need to distinguish between weaker or stronger types of adjoints or even to change our usual notions, which is quite convenient.

3. Dialog Game Protocols

The primary purpose of the work in this paper is for modelling dialog game protocols (see section 4). Formal dialog games are interactions between two or more players, where each player "moves" by making utterances, according to a defined set of rules. Their history in philosophy dates at least to Aristotle [4] and they were widely studied by philosophers in medieval times [34]. In modern times, formal dialog games have found application in philosophy [13], formal logic [22, 28], computational linguistics [21], and computer science [5]. Dialog games differ from the games of economic game theory [29] in that payoffs for winning or losing a game are not considered, and because there is no use of uncertainty measures, such as probabilities, to model the possible moves of opponents. They also differ from the abstract games used as a semantics for interactive computation [1], in the tradition of Hintikka's game-theoretic semantics [14]; these abstract games do not share the rich rule structure of agent dialog games, and are not themselves intended to have interpretations involving the beliefs or actions of multiple agents.

The main application of dialog games in computer science has been to the design of protocols for interaction between autonomous software agents. In [25], one of the authors presented a model of a generic formal dialog game in terms of the components of its specification, and we summarise this model here. We first assume that the topics of discussion between the agents can be represented in some logical language, whose well-formed formulae are denoted by the lower-case Roman letters, p, q, r, etc. A dialog game specification then consists of the following elements:

Commencement Rules: Rules which define the circumstances under which a dialog commences.

Locutions: Rules which indicate what utterances are permitted. Typically, legal locutions permit participants to assert propositions, permit others to question or contest prior assertions, and permit those asserting propositions which are subsequently questioned or contested to justify their assertions. Justifications may involve the presentation of a proof of the proposition or an argument for it. The dialog game rules may also permit participants to utter propositions to which they assign differing degrees of commitment, for example: one may merely *propose* a proposition, a speech act which entails less commitment than would an *assertion* of the same proposition.

Locution Combination Rules: Rules which define the dialogical contexts under which particular locutions are permitted or not, or obligatory or not. For instance, it may not be permitted for a participant to assert a proposition p and subsequently the proposition $\neg p$ in the same dialog, without in the interim having retracted the former assertion.

Commitments: Rules which define the circumstances under which participants express commitment to a proposition. Typically, the assertion of a claim p in the debate is defined as indicating to the other participants some level of commitment to, or support for, the claim. Since the work of Hamblin [13], formal dialog systems typically establish and maintain public sets of commitments, called *commitment stores*, for each participant; these stores are usually non-monotonic, in the sense that participants can also retract committed claims, although possibly only under defined circumstances.

Commitment Combination Rules: Rules which define whether and how commitments may be combined in a dialog or between dialogs. If two commitments are in conflict with one another, for example, which commitment takes precedence over the other? The answers to such questions are typically application-dependent, as discussed in [25].

Termination Rules: Rules which define the circumstances under which a dialog ends.

Dialog game protocols have now been articulated for many different types of dialogs, for example: information-seeking dialogs [35]; inquiries [24]; persuasion dialogs [2]; negotiation dialogs [3]; and deliberations [23]. It is worth noting that more than one notion of *commitment* is present in the literature on dialog games.[2] Our primary motivation is the use of dialog games as the basis for automated interactions between autonomous software agents. Because agents will typically enter into these interactions in order to achieve some wider objectives, and not just for the enjoyment of the interaction itself, we believe it is reasonable to define commitments in terms of future actions or propositions (indicative statements) about the world outside the dialog. In a commercial negotiation dialog, for instance, the utterance of an offer may express a willingness by the speaker to undertake a subsequent transaction on the terms contained in the offer. For this reason, we can

[2]For example, Hamblin [13] treats commitments in a purely dialogical sense, as an imperative to defend a statement in the dialog if requested. In contrast, Walton and Krabbe [35] treat commitments as obligations to (execute, incur or maintain) a course of action, which they term action commitments. These actions may be utterances in a dialog, as when a speaker is forced to defend a proposition he has asserted against attack from others, or they may be propositions referring to the world beyond the dialog. In contrast, Singh's social semantics [33] requires participants in an interaction to express publicly their beliefs and intentions; these expressions are called *social commitments*. These include both expressions of belief in propositions and expressions of intent to execute or incur future actions.

view commitments as semantic mappings between locutions and subsets of some set of statements concerning actions or beliefs in the world external to the dialog.

4. Modelling Dialog Game Protocols

In this section, we present a detailed overview of our framework for modelling dialog game protocols using the theory of categories enriched over pointed sets. The allowable sequences of locutions are modelled as an enriched category, and the outcome of each sequence is viewed as a morphism in an outcome category via a functor. Enriching the categories and functors over pointed sets allows us to specify illegal behaviour. We call a category that models the sequences of locutions a *locution* category, and a category that models the outcomes an *outcome* category.

4.1. **Locution Categories.** The allowable sequences of locutions of protocols are modelled using small categories enriched over Sets_*, as discussed in Section 2. In a locution category, the set of objects in the category represent placeholders (or possible states of dialogs), while the indecomposable morphisms between any two fixed objects represent locutions. Thus, if we have the locutions $f : A \to B$ and $g : B \to C$, then it is possible to compose the two together at B to bring about a new morphism. We represent this morphism as $g \circ f : A \to C$, indicating that the participating agents can utter f followed by g. It is this composition that allows us to model dialogs as sequences of locutions. Note that one could label the composition of f and g as $f; g$, or something else that resembles a sequential composition operator in computer science, but for the purpose of this paper, we use the notation commonly used in category theory.

Then arbitrary morphisms in our morphism category will correspond to sequences of locutions, which we will refer to as partial dialogs, with the adjective partial removed if the dialog cannot be legally extended by further locutions. Therefore, a locution category consists of a set of objects (the placeholders), a set of pointed sets of morphisms indexed on ordered pairs of objects (the partial dialogs), and a composition law:

$$\mathcal{D}(B, C) \times \mathcal{D}(A, B) \to \mathcal{D}(A, C).$$

The composition law of a standard category poses a problem for modelling protocols: if we have a category containing a set of morphisms representing locutions, we may wish to forbid the composition of certain combinations of these morphisms. For example, consider a case in which we have a category containing two morphisms (among others) $f : A \to B$ and $g : B \to C$. Because B is the target of f and the source of g, the composition law states that there must be a resulting composite morphism from A to C. However, it may be that we want to exclude combinations with g coming directly after f in a dialog. For example, it may be desired to forbid agents from uttering contradictory locutions in a dialog.

To get around this problem, one might work with precategories or composition graphs, where not all compositions are necessarily defined. However, that introduces the necessity to work out extensive mathematical material before proceeding with certain operations, since these objects have not been heavily studied in mathematics. We would prefer to work with a mature mathematical theory for our framework, in order to have a reliable source of technical results to call upon. Thus, in the appendix we prove that what we call (strictly associative) precategories are equivalent to categories (without zero objects) enriched over pointed sets, and one half of the equivalence is really just an extension of the process discussed in Example 2.8.6. We also present a third alternative equivalent view in the appendix (relying on categories with zero objects and functors which preserve zero objects).

To model illegal dialogs using categories enriched over pointed sets, if we want to forbid $g \circ f$, we instead model the protocol to indicate that the composite morphism $g \circ f$ is the basepoint of the relevant morphism set, which we refer to as the illegal morphism, and represent using $*$. The material in the appendix essentially details the fact that refusing to define certain compositions is logically equivalent to declaring exactly that set of possibly composable pairs to take on a single, fixed "illegal" value, provided one deals with composition carefully. In our case, the composition is dealt with by working with a category enriched over pointed sets, whose composition law is a map:

$$\mathcal{D}(B, C) \wedge \mathcal{D}(A, B) \to \mathcal{D}(A, C).$$

Defining $g \circ f = *$ is a *protocol-specific* decision. That is, this composition may be illegal in this instance, but other protocol models may allow such a composition. However, composing an illegal partial dialog with any other partial dialog should always be an illegal partial dialog. That is, the compositions $g \circ *$ and $* \circ g$ should be illegal for any morphism g in any locution category. Considering that this is not a protocol-specific decision, but generic, it seems useful to prevent this composition at the framework level to prevent protocol specifiers introducing errors, and to simplify their task.

4.2. **Outcome Categories.**

Objects in locution categories are simply used as placeholders, indicating the start-points and end-points of the partial dialogs. Modelling a protocol as a category enriched over pointed sets, with a set of placeholder objects, a set of morphisms, and composition law specifying the combination of morphisms specifies only the legal/illegal sequences of locutions of a protocol, but not the outcomes or meanings of the locutions or sequences of locutions. For example, if we were to specify an English auction, we could specify that a participant cannot bid once the auctioneer has announced the auction is finished, but we cannot specify what the outcome of a legal bid is; that is, we cannot specify that the value of the current bid is increased. By "outcome", we mean not only the conditions that hold when the protocol terminates, but also the conditions that hold during the dialog, which may or may not be in a scope outside the category itself. That is, they may be commitments to perform some action after the protocol has terminated; although such commitments may arise from utterances made at any time during the dialog.

Some people view this assignment of messages to meaning as unnecessary, and prefer to take the view that the meaning of messages can be interpreted from the messages themselves by specifying a finite set of locutions, and giving each locution a semantics. In this case, the locution category is sufficient to model a protocol, and assigning a meaning to locutions is unnecessary.

However, our framework also incorporates the assignment of meanings to dialogs, or even to locutions. One solution for an auction protocol would be to replace the set of placeholders with a set of pairs indicating the current bid and the participant who made the last bid. However, we introduce a more flexible way of modelling outcomes that uses a functor to map each (partial) dialog in a protocol to its outcome in a pointed monoid.

Outcomes, like the locutions, are modelled using a category enriched over pointed sets. As with locution categories, we equip the set of morphisms $\mathcal{D}(A, B)$ with a basepoint, $*$, for every pair of objects, A and B. In order to allow us to model the interactions of outcomes, they are modelled as morphisms in a category, rather than as objects. In fact, our models of outcome categories contain only a single object, implying that all outcomes are composable, unless explicitly excluded by the composition law of the category. If we started with some $\mathcal{O}'$ with more than one object, simply apply the collapse operation of Example 2.8.7 and compose with the Sets_*-functor of Example 2.10.6. Among other advantages, this means we do not need to specify what happens to objects when describing our functors, since all objects must be sent to the unique object.

Assigning outcomes to locutions, or even to dialogs, is handled using an enriched functor. Thus, given a locution category, $\mathcal{D}$, and an outcome category, $\mathcal{O}$, a functor $F : \mathcal{D} \to \mathcal{O}$ assigns an outcome morphism, $F(f)$, in $\mathcal{O}$, for every morphism (partial dialog) f in $\mathcal{D}$. Enriched functors are basepoint-preserving by definition, therefore, $F(*) = *$, indicating that the outcome of an illegal locution or dialog is always distinguished as illegal. Again, we suggest this should not be a protocol-specific decision, so we encode it in the mathematical framework to make this clearer for the end user.

One advantage of taking this approach, rather than, for example, assigning the objects in $\mathcal{D}$ to outcomes, is that outcomes are now composable, and so we can model their interactions. However, there is a more important reason for doing this. Using functors allows us to specify a collection of dialogs using a category $\mathcal{D}$, but to leave the outcomes of these dialogs unspecified. Supplying a functor $F : \mathcal{D} \to \mathcal{O}$ allows us to use the same locution category to represent many different protocols: those with the same dialogs but with different outcomes. Therefore, one can specify the rules of a protocol without having to instantiate the outcomes of that protocol.

We can also use the inverse idea, and specify a category of outcomes $\mathcal{O}$, and combine this with different locution categories specifying the rules to achieve these outcomes. Such an approach allows us to study the relationship between categories that have the same outcome category. This is discussed in more detail in Section 5.

Consolidating the above sections, one can see that a model of a protocol is a morphism, $F : \mathcal{D} \to \mathcal{O}$, in Cat_* (the category of small categories enriched over pointed sets), in which $\mathcal{D}$ is the locution category modelling the sequences of allowable locutions, $\mathcal{O}$ the outcome category modelling the possible outcomes, and F the functor mapping dialogs to outcomes.

5. Categories of Protocols

A primary motivation behind our mathematical model is to study the relationship between different protocols. This is a key reason for using enriched category theory to represent protocols instead of other methods, such as graphs, finite automata, or precategories. In this section, we discuss how category theory can be used to study protocol relationships.

5.1. **Over- and Under-Categories.**

Definition 5.1. Take $\mathcal{C}$ to be a category, and A to be an object in $\mathcal{C}$. The *category of objects over A*, $\mathcal{C}/A$, also called the *over-category*, is a new category such that

(1) the objects of $\mathcal{C}/A$ are the morphisms $f : B \to A$, for any object B in $\mathcal{C}$; and
(2) the morphisms of $\mathcal{C}/A$ between $f : B \to A$ and $g : C \to A$ are arrows $h : B \to C$, such that $f = g \circ h$.

Informally, this states that the over category $\mathcal{C}/A$ is the category in which the objects are any morphism in $\mathcal{C}$ whose target object is A, and the morphisms from $f : B \to A$ to $g : C \to A$ are the arrows $h : B \to C$, such that the following diagram commutes:

(9)
$$\begin{tikzcd} B \arrow[rr, "h"] \arrow[dr, "f"'] & & C \arrow[dl, "g"] \\ & A & \end{tikzcd}.$$

The dual of an over category is the *category of objects under* A, written $A/\mathcal{C}$, and also known as the *under-category.* As one would expect, the objects are morphisms whose source object is A, so the above diagram describing the morphisms is simply inverted:

(10)
$$\begin{tikzcd} & A \arrow[dl, "g"'] \arrow[dr, "f"] & \\ B \arrow[rr, "h"'] & & C. \end{tikzcd}$$

Example 5.2. (1) $\mathrm{Sets}/\mathbb{N}$ is the category of $\mathbb{N}$-graded sets. Given $F : S \to \mathbb{N} \in \mathrm{Sets}/\mathbb{N}$, define S_n to be the subset $F^{-1}(n) \subseteq S$, and similarly for $G : T \to \mathbb{N}$. The fact that morphisms in the over-category are commutative triangles then corresponds to the usual condition that a map of graded sets restricts to a mapping $S_n \to T_n$.
(2) If $\mathcal{C} \in \mathrm{Cat}_*$, $\mathrm{Cat}_*/\mathcal{C}$ is the category of pointed categories over $\mathcal{C}$, containing all functors $F : \mathcal{D} \to \mathcal{C}$, in which $\mathcal{D}$ is any pointed category.

Our thesis is that over- and under-categories can be used to study the relationships between protocols. In fact, Example 5.2.2 is exactly the structure that we use.

The reader should also be warned that we do not assume all dialogs begin at the same point in this work in general, although the idea of working under (or over) a fixed object in Cat_* could be used to model that case as well.

Definition 5.3. Given an outcome category (i.e., a pointed monoid) $\mathcal{O}$, the over-category $\mathrm{Cat}_*/\mathcal{O}$ is the *category of protocols over* $\mathcal{O}$.

Notice, $\mathrm{Cat}_*/\mathcal{O}$ contains every protocol in which $\mathcal{O}$ is the outcome category, and its morphisms are, by construction, compatible with locution categories and assignments of outcomes. From Definition 5.1, we know that the morphisms of $\mathrm{Cat}_*/\mathcal{O}$ are such that the following diagram commutes:

(11)
$$\begin{tikzcd} \mathcal{D} \arrow[rr, "H"] \arrow[dr, "F"'] & & \mathcal{D}' \arrow[dl, "G"] \\ & \mathcal{O} & \end{tikzcd}$$

where $\mathcal{D}$ and $\mathcal{D}'$ are objects in Cat_*. This implies that there is a natural notion of morphism between protocols that share the same outcome category.

This definition is important in the framework, because it gives us a robust framework for studying the relationships between protocols with the same outcome categories; that is, protocols that achieve the same outcomes, but using different dialogs. For example, given the outcome category that models the results of auctions (with a fixed number of bidders and monetary system), we can study the category of auction protocols (under those assumptions), and the relationships between the different types of auctions.

Example 5.4. As an example of the use of $\mathrm{Cat}_* /\mathcal{O}$, consider the implications of defining functors over $\mathcal{O}$ for the purpose of deriving new protocol models. If one has a family of protocol models whose outcome category is $\mathcal{O}$, and wants to use these models on a similar outcome category $\mathcal{O}'$, one can define a functor $F : \mathcal{O} \to \mathcal{O}'$ between the two outcome categories, which identifies related outcomes. From there, defining a functor $H : \mathrm{Cat}_* /\mathcal{O} \to \mathrm{Cat}_* /\mathcal{O}'$, which creates a new family of protocols that have the same dialogs as the first, but with different outcomes, would be straightforward:

$$H(G) = F \circ G.$$

(This construction is often referred to as postcomposition or base change in the mathematical literature.) Therefore, for every $G : \mathcal{D} \to \mathcal{O}$, the composition $F \circ G : \mathcal{D} \to \mathcal{O}'$ is the same dialog protocol as G, but with dialog morphisms mapped to morphisms in $\mathcal{O}'$ instead of $\mathcal{O}$.

This example can be instantiated using the examples in Section 6. Many of these examples in Section 6 model negotiation protocols, and the outcome categories consist of natural numbers[3] representing a price. In these cases, the natural numbers are an abstract model of positive values of currency, whereas the whole numbers allow zero as well. If we take the functor F from above as mapping natural numbers representing dollars to rational numbers representing euros, for example, we can define $H(G)$ using F to give us the negotiation protocols with euros as the outcome categories. Not only would defining the functor F be more straightforward than defining a new functor mapping the dialogs to outcomes, but F would also be reusable within other protocols that require translation between currencies (at this exchange rate).

Under-categories can be used similarly to study the category of protocols that share the same locution category. Given a locution category, $\mathcal{D}$, the under-category $\mathcal{D}/\mathrm{Cat}_*$ is the category whose objects are morphisms in Cat_* $F : \mathcal{D} \to \mathcal{O}$ for some $\mathcal{O} \in \mathrm{Cat}_*$. The morphisms in the under-category from F to $G : \mathcal{D} \to \mathcal{O}'$ are those $H : \mathcal{O} \to \mathcal{O}'$ that make the following commute:

(12)
$$\begin{tikzcd} & \mathcal{D} \arrow[dl, "F"'] \arrow[dr, "G"] & \\ \mathcal{O} \arrow[rr, "H"'] & & \mathcal{O}' \end{tikzcd}$$

[3]In fact, the outcome categories are more than this, but for the purpose of discussion, we will assume otherwise for the moment.

Thus, as with the over-category, the under-category implies a natural notion of morphism between protocols with the same locution category. In fact, working in the arrow category of Cat_* would give a natural way to define morphisms between arbitrary protocols as commutative squares in Cat_*, which is somewhat more general than those given by either over- or under-categories.

The notion of an auction protocol being translated between different currencies would again appear as a morphism in the under-category, but less naturally. In the over-category base change construction above, the currency exchange would be viewed as defining the new protocol from the existing one. However, the under-category will indicate this transition as a correspondence, but not in the sense of defining G from F. Thus, the two views would reflect different perspectives which could both be useful in different contexts.

5.2. **Nerves of (Pointed) Categories.** Among the most important basic objects in category theory are categories denoted $[n]$ discussed in Example 2.10.4, which contain only a string of n composable morphisms aside from the required identity maps (and illegal maps if we work with $[n]_+$). For example:

(13)
$$\begin{tikzcd}
{[1]} & 0 \arrow[r] & 1 \\
{[2]} & 0 \arrow[r] & 1 \arrow[r] & 2 \\
{[3]} & 0 \arrow[r] & 1 \arrow[r] & 2 \arrow[r] & 3
\end{tikzcd}$$

Then $[0]$ simply consists of a single object and its identity (while $[0]_+$ would also have an illegal morphism, making it isomorphic to $\mathcal{S}^0$ discussed in Example 2.8.1). The main use for a category $[n]$ is that a functor $G : [n] \to \mathcal{D}$ (or a pointed functor $G : [n]_+ \to \mathcal{D}$ if $\mathcal{D} \in \mathrm{Cat}_*$) is simply a string of n composable morphisms (commonly called an n-simplex in the category). Thus, given a locution category $\mathcal{D}$, we see $G : [n]_+ \to \mathcal{D}$ picks out a specific partial dialog in that protocol which consists of n composable morphisms. One should be careful to notice that the functional length of the partial dialog may be shorter than n locutions, since some of the chosen morphisms could be identity maps, thereby falsely extending the length of the partial dialog. Similarly, the morphism $G(k) \to G(k+1)$, which is an arrow between two objects in $\mathcal{D}$, may correspond to a long string of locutions in $\mathcal{D}$, thereby falsely shortening the length of the partial dialog. Thus, one must work harder to define the length of a partial dialog in this framework, by borrowing the notion of non-degenerate simplices of the nerve construction below.

Suppose $\mathcal{O}$ has only one object (as in the examples considered in Section 6) and $H : [n]_+ \to \mathcal{O}$ is a pointed functor. Then H corresponds to the choice of an ordered sequence of n outcomes (and hence includes the choice of their product) and this makes H an object in $\mathrm{Cat}_* / \mathcal{O}$. Thus, we can consider a morphism in $\mathrm{Cat}_* / \mathcal{O}$, which consists of a commutative triangle

(14)
$$\begin{tikzcd}
{[n]_+} \arrow[rr, "G"] \arrow[dr, "H"'] & & \mathcal{D} \arrow[dl, "F"] \\
& \mathcal{O}. &
\end{tikzcd}$$

This corresponds to a partial dialog in our protocol whose associated outcome sequence is the ordered sequence of outcomes associated to H. However, it may be that we have compressed the partial dialog by looking at a longer partial dialog and reducing it down to a shortened apparent length by composing certain portions and ignoring intermediate outcomes over the excised period. In any event, the functor G chooses a sequence of composable morphisms in $\mathcal{D}$, which corresponds to a partial dialog in our protocol (although we may be fast-forwarding certain portions of the dialog in some sense). The assumption that the triangle commutes implies the outcomes associated to the relevant portions of the partial dialog must be those associated to H. If we only want to pay attention to the final outcomes (and the initial state) rather than to an ordered sequence of intermediate outcomes, we should simply consider the case $n = 1$.

A natural object to associate to a model of a protocol is the collection of all such commutative triangles (graded by n). This is a very natural construction in category theory and corresponds to the simplicial set associated to a category, often called the (pointed) **nerve** of the category. In terms of protocols, this corresponds to looking at individual partial dialogs sorted by their ordered sequence of intermediate outcomes, possibly by ignoring some intermediate outcomes. If we restrict to what are usually called the 1-simplices, or setting only $n = 1$, this corresponds to considering partial dialogs where we keep only the final outcomes. As alluded to above, even the 1-simplices of the pointed nerve contains a substantial amount of information about the protocol model in question.

As one application, there are a large number of notions of equivalence of simplicial sets which one can import to the context of protocol models using this pointed nerve construction. We have already applied some of these to the study of protocol equivalence in [16].

6. Examples

In this section, we present some of the examples modelled using our framework.

6.1. **FIPA ACL.**

FIPA ACL, the Agent Communications Language of the IEEE Standards Committee Foundation for Intelligent Physical Agents (FIPA), defines 22 locutions which may be uttered by agents in a dialog in any order [11]. These include locutions to *inform* another agent of the truth of some proposition, or to *request* that some action be undertaken. The FIPA specification also publishes a collection of protocols that define the locutions that can be uttered for certain protocols, and one could attempt to formalise these protocols within our framework. However, one can view FIPA ACL (as opposed to its protocols) and its locutions as a free protocol representing all the possible dialogs. The ACL itself does not prevent any combinations of locutions, so an agent may utter any of the 22 locutions at any point in a dialog. This means the locution category should be (basepoints added to) the free category on the 22 possible locutions (with copies for each agent), essentially just a repeated tiling where the tile consists of 22 morphisms for each agent, with the same source and all targets are different. (See below for a tiling example with only three morphisms.) Free objects are one of the most fundamental concepts in mathematics, sometimes thought of as the most universal constructions. This connection with a free category explains the feeling that many other protocols could be modelled by imposing relations on the FIPA ACL.

For the outcome category in this example, notice that no specific outcomes are defined or associated to locutions in the FIPA ACL, hence there are no combination rules for outcomes. In our framework, this may be represented by saying the outcome category should simply be $[0]_+$ described in Section 5. That is, we should instead think of a single outcome which iterates to itself as the outcome associated to each (legal) locution.

To understand the functor $F : \mathcal{D} \to [0]_+$, it will suffice to notice that no combination of locutions is forbidden, so the basepoints are an afterthought in this case. In order to represent this, we should think of F as adding basepoints to an ordinary functor from a free category to the category with one object and only the identity morphism. There is only one such functor into such a trivial category, namely the functor which sends all objects to the unique object and all morphisms to the identity morphism. Thus, our functor F will send only the basepoint morphisms to the basepoint morphism of $[0]_+$ and every other morphism will be sent to the identity morphism in $[0]_+$. Mapping every morphism to the identity morphism in the codomain is known as a constant functor.

Of course, one can provide an alternative outcome category (and related functor) to instantiate the protocol for different usages. As an example, consider an outcome category in which the indecomposable morphisms represent atomic propositions in a logical language, and the composition of morphisms represents the conjunction of propositions. Therefore, the composition of morphisms a and b represents the proposition $a \wedge b$, in which $\wedge$ is the Boolean algebra conjunction operator, not the smash product operator. Composing further morphisms adds more constraints to the final outcome. One could define the mapping such that composing any morphism a with its Boolean negation $\neg a$ could result either in an illegal morphism, because we have a contradiction, or perhaps could represent the retraction of a, for example, in an argumentation dialog [26].

6.2. **English Auctions.** Perhaps the most widely-used formal interaction protocols are auctions. These are processes by which one or more buyers negotiates the price of some good with one or more sellers [19]. In the most common form of auction, the so-called *English* auction, multiple potential buyers of a single good bid increasingly higher prices to purchase the good from a single seller. The winning bidder is that potential buyer who makes the highest bid, and the amount paid by the buyer is the amount indicated in that highest bid. Each bid may be viewed as an utterance creating a commitment to purchase the item if agreed by the seller. We can represent this process by viewing our category as a tiling, where a single tile is defined by the number of locutions and the set of parameters allowable for each.

For example, suppose a basic auction protocol (for two bidders) consists of three possible utterances:

"Agent a increments the current bid by amount n";

"Agent b increments the current bid by amount m"; and

"The clock ticks with no bid".
When a then increments the current bid, this is an locution, while n is a parameter which would generally be a natural number. However, zero is always the parameter for the clock. (The clock is only included so that the end of the auction is detectable by three consecutive clock ticks.) The basic "tile" would then consist of four objects (one more than the number of locutions), which we will label S, A, B, and C. Then $\mathcal{D}(S,A)$ would be the (pointed) natural numbers (the possible parameters) corresponding to the first locution where a increments the current bid. Similarly, $\mathcal{D}(S,B)$ would be the (pointed) natural numbers corresponding to the locution where b increments the current bid. Finally, $\mathcal{D}(S,C)$ would be two points, one corresponding to the clock tick and the other to the illegal locution. There would be no other morphisms aside from the required identities.

A diagram of this tile would be as follows:

(15)
$$\begin{tikzcd}
 & C & \\
 & S \arrow[u, "tick"] \arrow[r, "{a,\mathbb{N}_*}"] \arrow[dl, "{b,\mathbb{N}_*}"'] & A \\
B. & &
\end{tikzcd}$$

Now the point of the tiling idea is that we would think of each of A, B or C as a new location for 0 (with a bit of care at C). One iteration of this process might yield the following diagram (where new objects are de-emphasised):

(16)
$$\begin{tikzcd}
 & & & \bullet & & & \\
 & & & C \arrow[u, "tick"] \arrow[r, "{a,\mathbb{N}_*}"] \arrow[dl, "{b,\mathbb{N}_*}"'] & \bullet & \bullet & \\
 & & \bullet & S \arrow[u, "tick"'] \arrow[rr, "{a,\mathbb{N}_*}"] \arrow[ddll, "{b,\mathbb{N}_*}"'] & & A \arrow[u, "tick"] \arrow[r, "{a,\mathbb{N}_*}"] \arrow[dl, "{b,\mathbb{N}_*}"] & \bullet \\
 & \bullet & & & \bullet & & \\
 & B \arrow[u, "tick"] \arrow[r, "{a,\mathbb{N}_*}"'] \arrow[dl, "{b,\mathbb{N}_*}"'] & \bullet & & & & \\
\bullet & & & & & &
\end{tikzcd}$$

Iterating this procedure but allowing tiles only "three high" to establish the ending condition from three consecutive ticks yields something like a 3-dimensional lattice, which can be reasonably described as a partially free category (enriched over pointed sets). Notice we also avoid much of the state-space explosion problem of the FIPA ACL in this case, since any three consecutive ticks ends the dialog, allowing us to impose a height restriction in this diagram.

Notice the fact that bids must (strictly) increase is modelled by the fact that the parameter is either a natural number (the stated increment to the previously existing bid) or is illegal. It is also important to notice the resulting composition is associative, which is required in all cases.

For the outcome category, we first build a small pointed monoid consisting of 1, a, b and the basepoint, which we'll call the "last bidder" pointed monoid. This keeps track of the last real bidder (either a or b, 1 if neither a nor b appeared yet, or the basepoint if it appears anywhere), which is associative but definitely not commutative. Now take the smash product operation of Example 2.8.2 with the pointed whole numbers under addition considered as a pointed monoid (so zero is now the identity in this second factor) to form the outcome category.

The functor would simply take "a increments the bid by n" to the pair (a, n) in this notation (any tick of the clock would be sent to $(1, 0)$). At any time, incrementing the bid amounts to committing the bidder to pay the new total if there are no more bids. The auction ends after the clock ticks three consecutive times (which is as high as the tiles allow one to go).

Also note the current bid is not explicitly mentioned in the locution category, but the addition operation (as composition) in the outcome category keeps track of the current total bid as part of the obligation incurred should the auction end. Compatibility with the addition in the outcome category, so the assignment of outcomes is functorial, is the reason the locution category must focus on the increment rather than the current bid.

In the situation where there are only two bids by different parties, the locution sequence would be a increments the bid by x, followed by b increments the bid by y. The outcome associated to this string would be $(b, y) \circ (a, x) = (b, x + y)$. In particular, the outcome category sees this as equivalent to a single action by b to increment the bid by $x + y$, although the locution category does not see these as equivalent.

Notice this description allows a bidder to insert two increments in sequence, without any other bidder being involved, although one could specifically exclude that choice as well (by removing one branch of certain tiles). One reason to leave this possibility available at this level is that one could view the final round of betting in a poker hand as an example of an auction. In that case, the idea of a "string raise", iterating two consecutive bid increments without another acting, is important (although generally forbidden in that context for psychological as well as coherent action reasons). Also our flexibility in allowing dialogs to start at different points is useful in this example. The novice poker player might tend to ignore "the pot" when considering the final round of betting, but the incurred obligation for "raises" will not. In fact, our focus on the increment rather than the current total bid is more natural in the context of this poker example.

6.3. **Dutch Auctions.** A *Dutch* auction is something of the reverse of the English auction. Rather than bidders monotonically increasing the bid for the item, a Dutch auction starts with the auctioneer announcing a price (much higher than the item is expected to sell for), and the auctioneer monotonically decrements the price of the item until one bidder announces that they will buy the item at the current price. The auction terminates at this point, and the buyer must pay the last stated price.

Modelling a Dutch auction is similar to that of an English auction. The basic tile is not the same as the English auction, because the rules of encounter are different, but our approach is similar. The basic tile consist of four objects, $\bullet$ (the initial placeholder), S (the auctioneer), with A and B representing the bidders. Essentially, the bidders can only act by accepting the current bid, so the only morphisms from $\bullet$ to A are either $*$ or a (corresponding to "yes, bidder A will take it at that price") and similarly for B. The auctioneer also has limited responses (one virtue of this type of auction) which we represent as a morphism from $\bullet$ to S, either d to decrement the current bid by one monetary unit (possibly a large quantity when stated in a real currency) or *. A portion of the locution category $\mathcal{D}$ would then look like:

(17)
$$\begin{tikzcd}
& S & \\
& \bullet \arrow[u, "d"] \arrow[r, "a"] \arrow[dl, "b"'] & A \\
B & \bullet \arrow[u, "d"] \arrow[r, "a"] \arrow[dl, "b"'] & A \\
B & \bullet \arrow[u, "d"] \arrow[r, "a"] \arrow[dl, "b"'] & A \\
B & &
\end{tikzcd}$$

This is essentially free in one direction, with no need to forbid anything. A minimum bid is chosen by deciding how high to stack the tiles (as with the clock in the English auction example), so the auction ends either by leaving the vertical axis along a bid branch, or by reaching the top of the vertical axis, indicating that the reserve price has not been met. This general premise of making the ending conditions structural in the indexing category is important, as it allows us to avoid imposing relations which might make our desired assignment of outcomes become incompatible with the composition (thereby losing functoriality).

The outcome category will again involve the "last bidder" pointed monoid introduced in the case of the English auction to determine the winner. Taking the smash product operation of Example 2.8.2 with the pointed whole numbers considered as a pointed monoid then yields the outcome category $\mathcal{O}$. The functor $F : \mathcal{D} \to \mathcal{O}$ will satisfy $F(d) = (1,1)$, $F(a) = (a,0)$ and $F(b) = (b,0)$. Thus, the first coordinate will indicate the winner, while the second will indicate the sum of the decrements of the bid, from which the actual price may be recovered. Notice, this is an instance where we see the artificial nature of the choice of initial price indicated by the fact that it is easier to model the protocol as avoiding explicit mention of that choice.

It is interesting to note that our model generalises beyond monetary auctions. For example, a Dutch auction is easy to implement in a barter economy, but this model would then violate the functorial nature of the assignment of outcomes, unless one artificially unitises the possible bids ahead of time (nearly a currency exchange operation). However, the English auction stays roughly the same in a barter economy, with the pointed natural numbers replaced by a copy of them for each type of item considered of value for the purposes of that auction house (3 pigs, 2 chickens and a nice gold watch) is now a price. Essentially, the relationship between the Dutch and English auctions depends upon the monetary unit device; while we know how to construct an inverse to adding one unit, it is hard to obtain a common agreement on the value of a certain cow minus a pig. (What if either the cow or the pig is particularly sickly looking?).

6.4. **Monotonic Concession Protocol.** The monotonic concession protocol [36] is a formalism of the type of interaction that many people think about when they think of negotiation. The protocol is a (typically) two participant protocol that proceeds in rounds. In the initial round, each participant puts forward a proposal, for example, participant A proposes that A gives participant B an amount of money, and B will provide A with an item, and at the same time, B makes its own proposal. If either participant's proposal is acceptable to the other participant, then they have reached an agreement, and the protocol terminates. Otherwise, the participants enter another round, each putting forward a proposal *at least as good to the other participant* as the proposal it put forward in the previous round. This requirement means that the proposals are *monotonically* converging towards each other at each round. If both participants put forward the same proposal as they each did in the previous round, the protocol terminates, and no agreement is reached.

There are many different versions and abstractions of the monotonic concession protocol. As is common in economic models of the protocol, we consider that the two participants know each other's preferences for the deal, and that each participant represents its preference as a real number in the interval $[0, 1]$, with 0 being the least preferred, and 1 being the most. These models also assume that one participant's most preferred option is the other's least preferred. The moves in the protocol are represented as a value in this range, representing the amount that the participant has conceded since the initial proposal. Therefore, if both participants initially propose their most preferred options, both initial moves will be 0. At each subsequent round of the protocol, each participant then submits a proposal which is no less preferred (by the other participant) than its current proposal. Due to the monotonic nature of the protocol, these values will converge towards each other during the running of the protocol, and termination is reached if the total of the two values is greater than or equal to 1, or if neither participant concedes on any round. Clearly, the negotiation could continue infinitely as both agents could put forward proposals whose totals converge towards 1, but never reach it. However, this could not occur if the negotiation space was finite, for example, if the rules were to declare that the preferences must be vulgar fractions with denominators no larger than a fixed amount.

Alternatively, one could introduce a small positive parameter $1 > \epsilon > 0$, and say the negotiation terminates in the ϵ-protocol if $x+y > 1-\epsilon$. Thus, either the parties remain at least this fixed ϵ apart (symbolically $\epsilon \leq 1-(x+y)$), or they have crossed this negotiation threshold and declare the situation acceptable in the ϵ-protocol. Notice, it is important that we do not use the termination condition $x + y \geq 1 - \epsilon$, since all terms of infinite sequences x_n and y_n may fail to satisfy such a condition but their limits could still satisfy the condition. This is not possible with the stated strict inequality because of an analytic property of the real numbers.

We will describe the objects of the locution category as a subset of 3-dimensional space $\mathbb{R}^3$, which is abstractly represented in Figure 1. The objects will be tuples (x, y, z) where x and y are real numbers in the unit interval, representing the respective value of the preferences, while z is either 0 or 1, representing whether at least one agent has conceded from the previous round ($z = 0$), or neither agent has conceded ($z = 1$). In the case where $z = 1$, we also want to exclude any tuples with $x + y \geq 1$, so we have a triangle sitting above a square in Figure 1, which covers the same area on the xy-plane as the lower triangle in the square, but is 1 up on the z axis. For morphisms, we will only have identities and $*$ morphisms if the source satisfies either $z = 1$ or $x + y \geq 1$ (with $z = 0$). In addition to these "trivial" morphisms, there will be a unique non-trivial map $(x, y, 0)$ to (x', y', z) if either $x' = x, y' = y, z = 1$ or $x' \geq x, y' \geq y, z = 0$. More geometrically, there are non-trivial arrows in the vertical direction, and within the xy-plane along a first quadrant vector from any point "below" the line $y = 1 - x$ (that is, in the shadow of the upper triangle). The objects which are the source of only trivial maps (the upper triangle and anything not in its shadow down in the square) represent possible ending states of the negotiation, the vertical arrow represents both parties refusing to change their offer from that point, and the arrows in the xy-plane represent the possible changes of current offers.

Form the outcome category by taking the product of two copies of $[0, 1]$, given a partial "multiplication" by addition in each coordinate (where that remains within the unit square) and then produce a pointed monoid as in Example 2.8.6. The functor then takes a morphism of the form $(x, y, 0) \to (x', y', 0)$ to the ordered pair $(x' - x, y' - y)$ (the components of the vector in question) and $(x, y, 0) \to (x, y, 1)$ to the identity map $(0, 0)$. Since the composition law in the locution category is really just vector addition, this assignment will be functorial.

As a consequence, any negotiation which reached a termination point would have an associated outcome of the form (x, y) where x was the total concession of the first party, and y was the total concession of the second party. If $x + y < 1$ then the parties failed to agree by both repeating those offers. If $x + y = 1$, then a deal has been reached and the details of the offers are recorded. Finally, if $x + y > 1$, then a deal is possible but the details still need to be worked out, since the parties have offered more concessions than were strictly necessary. This suggests perhaps allowing another stage of negotiation where monotonicity might be reversed, modelled by allowing maps back toward the line $y = 1 - x$ from points $(x, y, 0)$ with $x + y > 1$.

This model generalises to the analogous concept where n parties agree when $\Sigma_{i=1}^{n} x_i = 1$, by a similar construction from the points in $\mathbb{R}^n$ "below" the hyperplane given by this equation. However, this brings up the far more flexible prospect of studying a similar question with more general surfaces than this basic hyperplane, or even just using hyperplanes with different slopes along different axes. Since there seems to be no agreement on the general termination conditions with more than two parties [9], this added flexibility would allow one to study global properties without first choosing a termination condition, or even how the protocol changes by varying the termination condition.

6.5. **Combinatorial Auctions.** A combinatorial auction [8] is an auction in which bidders are competing for more than one item, and single bids can be placed on sets of items, rather than just individual items. These auctions are used in situations in which the value of a set of items is worth more than the summed value of each of these items. For example, take the case of a federal communications commission auctioning off licenses to supply mobile phone services [12]. The country is divided into geographical areas, and companies bid to win the license of a particular area.

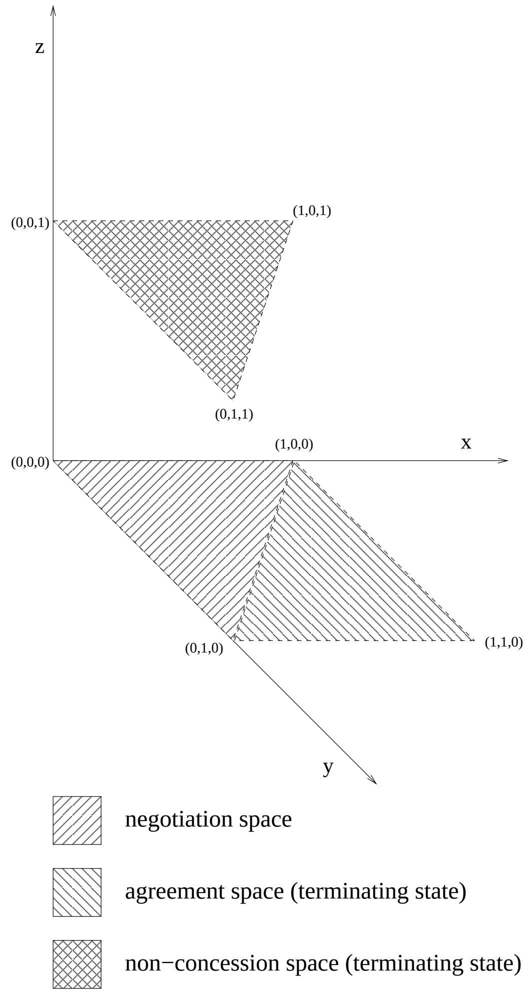
z
(0,0,1)
(1,0,1)
(0,1,1)
(0,0,0)
(1,0,0)
x
(0,1,0)
(1,1,0)
y
negotiation space
agreement space (terminating state)
non−concession space (terminating state)

Figure 1. Model for the 2-party Monotonic Concession Protocol

Companies would rather own the licenses of a cluster of areas that border each other, because there are cost savings and synergies from owning the license for adjoining geographical areas; for example, servicing one larger area is more efficient than servicing several isolated areas, and commuters that travel over more than one area can be offered special services.

Combinatorial auctions are well suited to such a situation — one can bid on clusters of areas, rather than having to take their chance on bidding on each of them separately and losing one. Such an approach encourages bidders to place higher bids — the value of the set of items is greater that the value of their individual sums. This is beneficial to the sellers, who earn more from the auction, and the buyers, who get a product that will generate more income for them.

We will model the locutions along the lines of the English auction, with the parameter being a non-zero vector (so not all of the components can be zero at once) whose coordinates are either natural numbers or zero (or *). The components of the vector should be indexed by non-empty subsets of the collection of items under consideration (often called $\mathcal{P}_1$, as a variant of the power set construction). In particular, if there are n items at auction, then there will be $2^n - 1$ components of the bidding vectors. Alternatively, we could add a sum component to the vectors. This would allow us to specify that the vector must be non-zero more naturally by declaring the sum of the increments must be a (non-zero) natural number and give bid vectors with 2^n components.

For the outcome category, we must have a more complex vector, to deal with the inherent complexity of deciding the optimal sale from the point of view of the auction house. In this model, we assume that the earnings of the auction house are a percentage of the total amount made, so they earn the most by selling every one of the items exactly once, and therefore focus on the total of the bids associated to any partition of the items available.

First, consider the collection of partitions of the set $\{1, \ldots, n\}$. Given any partition $A_1, \ldots A_k$ (ordered by smallest element), we need the smash product of k copies of the "last bidder" pointed monoid with the pointed whole numbers. Thus, an element is a tuple $(a_1, \ldots, a_k, m)$ where a_i represents a bidder (or unit or *), and m is a whole number or *. The smash product construction simply ensures that if any component is *, then the entire vector is defined to be *. Unfortunately, this is with just one partition, so we must now take the smash product of these pointed monoids, indexed over the set of all partitions of $\{1, \ldots, n\}$.

For the functor, we first describe what should happen given a single choice of partition. If the locution parameter vector had a zero component for each of the subsets A_i, or the locution was a clock tick, then it should be sent to $(1,0)\ldots,(1,0),0$, with the unit bidder in each component and zero values. If the parameter vector is non-zero for an A_i, then the chosen bidder a_j and the increment for that group should appear in the slot i, and the sum of the increments should appear in the last slot. For example, if a_3 increments the bids by m_1 on group $\{0,2\}$ and by m_2 on $\{3\}$, then for the partition $\{0,2\},\{1\},\{3\}$ this would be sent to $(a_3, m_1), (1,0), (a_3, m_2), m_1 + m_2$. To construct the entire functor, now perform this type of operation with each choice of partition, although the smash product has the property of making even this bid illegal if, in some other part of the auction, some portion not visible here was illegal.

This entire large vector should be viewed as the outcome for any dialog, precisely because of the complexity of the decision making process from the point of view of the auction house. If there is a single instance of a maximum value according to the sum entries for the various partitions, then that partition should be chosen and the individuals listed as the remainder of that vector have bought the subsets which carry their name at the indicated prices. However, dealing with cases where there are ties between different partitions will be much more complex, presumably involving small bias choices as are common in sports.

After reviewing the model for the sealed bid auction, one might want to instead assign only the bid vector for the partition with the maximum total bid, but that would not be functorial here. Unfortunately, the sum of the maximum entries is not the same as the maximum value of the sum of ordered entries, since the maximal values may occur in different components of the two vectors, although the former is always the larger of the two.

6.6. **Sealed-bid Auctions.** Sealed-bid auctions are auctions in which bidders do not reveal the value of their bid to other participants, only to the auctioneer. Typically, each participant submits exactly one bid, so these are also *one-time* auctions.

Commonly used sealed-bid auctions are *first-price* and *second-price* auctions. In a first-price auction, each participant submits one sealed bid, and the auction is won by the participant that submitted the highest bid. The winner must pay the amount that they bid. A downside of a first-price auction is the *winner's curse*: the winner has paid more for the item than anyone else has valued it, so may have overpaid for the item. This typically means that bidders will often not bid their true valuation of the item [19].

Second-price auctions attempt to overcome this problem by operating in the same way as the first-price auction, but instead of the winner paying the price they bid, they pay the price bid by the second-highest bidder. This encourages bidders to bid their true valuation of the item, given that the winner knows that at least one other bidder values the item at the price they paid.

This example is discussed in this paper because this is an example of two protocol models that have the same locution category, given that the bidding process is the same in each, and the same outcome category, given that it contains the winning bidder and the price they pay, but the functor between the two categories is different.

Because the bids are sealed, locutions represent the auctioneer opening all of the bids. Thus, the locution category contains only two objects S and T (for source and target), while a morphism from S to T consists of a vector $(m_1, \dots, m_n)$ representing the bids of the n bidders (or *). It is useful to note that there are no non-trivial compositions in this locution category, so functoriality of any assignment out of this locution category becomes nearly automatic. For the outcome category, take the smash product of the "last bidder" pointed monoid with the pointed whole numbers considered as a pointed monoid. Thus, an outcome consists of a pair (a_j, m) where a_j represents a bidder (or the unit) and m a price.

For the first price auction, the assignment should send $(m_1, \dots, m_n)$ to (a_j, m_j) where m_j represents the (first instance, ordered by j, of the) maximal price and a_j the associated bidder. Simply mandating that identities and basepoints are preserved then makes this a functor. For the second price auction, the functor should send $(m_1, \dots, m_n)$ to (a_j, m_k), with a_j chosen as before and m_k the maximal remaining price upon ignoring m_j.

In fact, one could similarly consider the third price auction, or a median price auction, or the average price auction (assuming real number prices are allowed in the outcome category) or whatever other criteria one may want to enforce[4]. This suggests one might want to consider the entire ordered vector of pairs (a_j, m_j) as the outcome, since it contains all of the relevant information.

These two (or more) auctions provide excellent motivation for why we model protocols as morphisms in Cat_*, rather than as categories with the objects representing the outcomes. If we were to model the first-price and second-price auctions

[4]Byde [7] has applied evolutionary/computational game-theoretic techniques to find an optimal ω-price auction between the first-price and second-price auctions under certain assumptions regarding the preferences of buyers and sellers.

as single categories, the possibility of reusing protocol models and comparing different protocols with the same outcomes would be more difficult. The authors believe that the separation between the legal locutions and the outcomes is great enough that modelling them using morphisms between categories is beneficial.

7. Related Work

To the authors' knowledge, the work in this paper, and its predecessor workshop presentation [17], is one of only two attempts to provide a mathematical model of agent interaction protocols. Fernandez and Endriss [10] present a hierarchy of abstract models for protocols. At the base of the hierarchy are deterministic finite automata. The states in the automata represent states between locutions, which are the transitions in the automata. Fernandez and Endriss then restrict and extend these in different ways to bring about new classes of dialog, for example, by adding memory to the automata to keep track of the utterances that have occurred in the dialog, which can influence the transitions in the automata. This allows one to model protocols in which the rules permit the same state to occur more than once in a dialog, but from which different transitions are available.

Our approach is clearly different to Fernandez and Endriss, considering that we use enriched category theory, and that we model the dialogs separately from their outcomes. Extending the models with data structures such as stacks is unnecessary because using categories to model dialogs explicitly represents the allowable sequences of locutions.

Other authors have used deterministic finite automata and graphs to model multi-agent protocols [30, 31], as well as approaches such as Petri Nets [20] and process algebras [27]. However, such work is with a view to specifying and implementing multi-agent protocols, whereas we aim to provide a mature mathematical framework for studying protocols, their properties, and their relationships, not for protocol implementation.

8. Conclusions and Future Work

In this paper, we have presented in detail a mathematical framework for agent interaction protocols using category theory. In our model, the legal sequences of locutions are modelled as morphisms in a category enriched over pointed sets, in which the objects represent placeholders between locutions, and arbitrary morphisms represent strings of locutions. The composition law of the category provides us with a way to build up dialogs by composing together locutions. The outcomes of dialogs are specified by using a functor to map between dialogs and an additional category: the outcome category.

The theory of categories assume that if the target of one morphism, f, is the same as the source of another morphism, g, then there must exist another morphism $g \circ f$ from the source of f to the target of g. This would seem to imply that any two locutions that can be composed are legal dialogs. To represent that certain compositions are illegal by the rules of the protocol, the composition law enriched over pointed sets indicates that their composition is equivalent to an added morphism, $*$, called the basepoint. The theory of enriched categories is a mature mathematical theory, and the enriched categories over pointed sets behave very much as one expects from ordinary categories.

We present a number of examples of protocol models using our framework, and we also discuss the implications of using category theory for protocol modelling. One such implication is that it allows us to study the relationships between protocols using category theory because each protocol is a morphism in the category of small categories enriched over pointed sets. In fact, this correspondence was used in a weaker form to assist in constructing various notions of equivalence between two protocols in [16].

We see no restriction in our framework being generalised to model more general types of actions and their outcomes, and in fact, we believe this would be quite straightforward. Locutions are generally treated as certain types of action, and modelling the outcomes of actions would be similar to modelling the outcomes of locutions. However, this is not the motivation of our work, so we have not investigated this topic any further.

In future work, we plan to use our framework to model protocols and analyse their properties. Of particular interest to us are economic models of negotiation, and dialog protocols for argumentation; however, we believe the framework could be applied successfully in many other areas. Additional future work is planned to study *protocol combination*: the derivation of protocols from other protocols. Our interests lie in studying the situations in which certain types of combinations can be performed, and the models that result from these combinations. There are also several promising technical variations on some of the protocol models discussed in Section 6 which we would like to pursue, such as multi-party versions of the monotonic concession protocol.

Acknowledgements. The authors are grateful for financial support from the EC PIPS project (EC-FP6-IST-507019), website: `http://www.pips.eu.org`, the EC ASPIC project (IST-FPC-002307), website: `http://www.argumentation.org/`, and the EPSRC MBC project (GR/T10657/01), website: `http://www.marketbasedcontrol.com/`.

Appendix A. Replacing Strictly Associative Precategories

A precategory (or in some references a composition graph) is essentially a generalisation of a category where not all compositions are necessarily defined. We will focus on strictly associative precategories, where the collection of compositions which are not defined satisfies an associative law, a fairly technical restriction. We note that protocols can be modelled as strictly associative precategories, and then be replaced by categories enriched over pointed sets for the purpose of study. The advantage of this replacement is the ability to appeal to the mature mathematical theory of enriched categories. As a consequence, we would like to make precise the method of replacing strictly associative precategories with categories enriched over pointed sets, which is embodied in Theorem A.7 below. To this end, we need to define strictly associative precategories, since the literature is somewhat muddled concerning various types of precategories, which is our primary reason to avoid them.

A.1. **Strictly Associative Precategories.**

Definition A.1. (1) A small strictly associative precategory, often just a precategory in what follows, $\mathcal{C}$ will consist of two sets, a set of objects and a set of morphisms, together with an assignment of source and target object for each morphism. We will also require the existence of a restricted composition law which is associative in the strict sense. That is, there should be a map of sets $M_{B,C,D} \to \mathcal{C}(B,D)$ for some subset

$$M_{B,C,D} \subset \mathcal{C}(C,D) \times \mathcal{C}(B,C)$$

for each triple of objects B, C, D which is associative in the strict sense that $(f \circ g) \circ h = f \circ (g \circ h)$ whenever both compositions exist and both must fail to exist if either fails to exist. In addition, there must exist two-sided identity maps so that $M_{B,C,C}$ contains all pairs of the form $(1_C, f)$ (whose composition must be f) and $M_{C,C,D}$ contains all pairs of the form $(g, 1_C)$ (whose composition must be g). The categories are those precategories where $M_{B,C,D}$ is all of $\mathcal{C}(C,D) \times \mathcal{C}(B,C)$.

(2) A prefunctor will denote an assignment $F : \mathcal{C} \to \mathcal{D}$ between precategories which sends the objects of $\mathcal{C}$ to objects of $\mathcal{D}$ and is defined on some portion of the morphisms. In addition, it must satisfy $F(1) = 1$ (hence be defined on all identities, as expected) and $F(f) \circ F(g) = F(f \circ g)$ in the strict sense that they are equal when both sides exist and both sides fail to exist if either fails to exist.

(3) Define $Pre\,\mathrm{Cat}$ to be the category of (strictly associative) precategories and prefunctors. By the composition of two prefunctors, we mean the assignment defined by $F(G(f))$ whenever $G(f)$ is a morphism where F is defined. Notice, this is a category in the usual sense, since composition of prefunctors is always defined by construction.

Example A.2. (1) Consider the operation of addition on the set $\{0, 1\}$ as a precategory with one object $\bullet$. In this case, 0 is the required identity object, but we do not want to define $1 + 1$. Hence

$$M_{\bullet,\bullet,\bullet} = \{(0,0),(0,1),(1,0)\} \subsetneq \{(0,0),(0,1),(1,0),(1,1)\}$$

and notice there is no smaller non-trivial example, since all compositions involving an identity are required to be defined.

(2) Consider Example 2.8.6, in which a set D has a partial multiplication $\nu : M \to D$ for some subset $M \subset D \times D$. Define a precategory by defining the composition $g \circ f$ only in the cases that $(f, g) \in M$.

(3) Given a set S, let S_1 denote the precategory with one object $\bullet$, whose morphism set is S with a unit 1 added, and the set $M_{\bullet,\bullet,\bullet}$ only contains those pairs of the form $(1, f)$ or $(g, 1)$. Given another set T, a prefunctor $F : S_1 \to T_1$ simply corresponds to a set map from some subset of S to T. This follows from the fact that $F(1_S) = 1_T$ is forced. In particular, the (minimal) case where F is only defined for the identity corresponds to the inclusion of the empty subset of S into T. Given a third such set U and $G : T_1 \to U_1$ corresponding to a map of sets from a subset of T to U, the composite of prefunctors $G \circ F : S_1 \to U_1$ corresponds to the expected composite map of sets from a subset of S (possibly empty) to U. Thus, $?_1$ is a functor, even an equivalence onto its image, from the category of sets and partial maps into the category of precategories.

One goal of this appendix is to detail a correspondence between $Pre\,\mathrm{Cat}$ and Cat_* by direct construction of the relevant functors, whereupon the correspondence theorem is essentially reduced to an observation. Since the constructions are the main part of the results, the proofs and statements of the intermediate results will be informal.

Lemma A.3. *There is a functor $\Phi : Pre\,\mathrm{Cat} \to \mathrm{Cat}_*$ defined by adding basepoints to the morphisms sets.*

Given $\mathcal{C} \in Pre\,\mathrm{Cat}$, let the object set of $\Phi(\mathcal{C})$ simply be the object set of $\mathcal{C}$ itself. For each morphism set of $\Phi(\mathcal{C})$, we choose to add a basepoint $*$ and we denote the resulting set by $\mathcal{C}(B, C)_+$. Now, for each ordered pair in the complement of $M_{B,C,D}$ we define the composite to be $*$ in $\mathcal{C}(B, D)_+$. Since any pair with $*$ in either coordinate is not in $M_{B,C,D}$, we see the restricted composition law

$$M_{B,C,D} \to \mathcal{C}(B, D) \to \mathcal{C}(B, D)_+$$

induces a map from the smash product

$$\mathcal{C}(C, D)_+ \wedge \mathcal{C}(B, C)_+ \to \mathcal{C}(B, D)_+.$$

This new composition rule is now associative with identities by the definition of a strictly associative precategory.

For a prefunctor $F : \mathcal{C} \to \mathcal{C}'$, we define $\Phi(F) : \Phi(\mathcal{C}) \to \Phi(\mathcal{C}')$ as F on objects and any morphisms where F is defined and $*$ for any morphism where F is not defined. Since the identities were available and well-behaved in the precategories, this remains the case.

To see that $\Phi(F)$ is a functor we must show it commutes with any composition (as well as being well-defined). This splits into two observations: that F preserved compositions anywhere all of the pieces were defined and that $\Phi(F)$ sends everything else to $*$. Finally, the fact that $\Phi(F)$ preserves $*$ by definition suffices to imply it is an enriched functor.

It is straightforward to verify that $\Phi : Pre\,\mathrm{Cat} \to \mathrm{Cat}_*$ is then itself a functor, since the notions of composition of (pre)functors are consistent in the categories $Pre\,\mathrm{Cat}$ and Cat_*.

A.2. **Zero Objects.**

We need one more technical definition before we proceed to construct the inverse functor.

Definition A.4. In any category $\mathcal{C}$, a zero object $0 \in \mathcal{C}$ is one with the property that for any $B \in \mathcal{C}$ there exists a unique map $B \to 0$ and a unique map $0 \to B$.

The category of groups has the trivial group (containing only an identity element) as zero object, and the category of pointed sets has any singleton as a zero object.

Remark A.5.

(1) Any category with a zero object is enriched over pointed sets. Choose as the basepoint of each morphism set $\mathcal{C}(B, C)$ the unique map $B \to 0 \to C$. Uniqueness will suffice to imply any composite with this "zero map" on either side is another zero map. Thus, the composition law will descend to the quotient defining the smash product of morphism sets. For the unit map, choose the map $S^0 \to \mathcal{C}(B, B)$ which sends the basepoint to the zero map, and the non-basepoint to the identity map 1_B.

(2) All zero objects in any category are isomorphic. To see this, notice $0 \to 0' \to 0$ and $1_0 : 0 \to 0$ are both morphisms, so by uniqueness (using 0 a zero object) they must coincide. Similarly, using $0'$ a zero object the other composite is also the identity. In fact, this shows they are isomorphic via a unique isomorphism, as in the familiar example of pointed sets mentioned above.

(3) Choosing $B = 0$ in the definition, it follows that the identity and zero maps $0 \to 0$ coincide. Thus, the unit map $S^0 \to \mathcal{C}(0,0)$ is constant on the basepoint. In fact, in any category enriched over pointed sets, this property characterises zero objects. If the identity map for an object C is also the basepoint map, then $f = f \circ 1 = f \circ * = *$ for any $f : C \to B$ so uniqueness of this map follows and C is initial. The dual argument gives the condition that C is final as well, hence a zero object.

Lemma A.6. *There is a functor $\Psi : \mathrm{Cat}_* \to Pre\,\mathrm{Cat}$ defined by removing basepoints from morphism sets (as well as zero objects) and leaving $*$ compositions undefined.*

Define the functor $\Psi : \mathrm{Cat}_* \to Pre\,\mathrm{Cat}$ on enriched categories as follows. For the objects of $\Psi(\mathcal{D})$ take the non-zero objects of $\mathcal{D}$, and for morphisms of $\Psi(\mathcal{D})$ take the complement of the basepoint in each morphism set. The claim is that $\Psi(\mathcal{D})$ is then a precategory, with the restriction of the composition law of $\mathcal{D}$. In other words, we define $M_{B,C,D}$ to be the complement of the preimage of the basepoint under the composition map of $\mathcal{D}$

$$\mathcal{D}(C,D) \wedge \mathcal{D}(B,C) \to \mathcal{D}(B,D)$$

so that pairs in M have a composition in the complement of the basepoint in $\mathcal{D}(B,D)$, i.e. in $\Psi(\mathcal{D})(B,D)$. Since the composition law in $\mathcal{D}$ is associative, we see the result has the required property that $(f \circ g) \circ h = f \circ (g \circ h)$ whenever both compositions are defined and if one fails to exist, it must be the basepoint in $\mathcal{D}$ so that the other must also fail to exist because it also represented the basepoint in $\mathcal{D}$.

For an enriched functor $F : \mathcal{C} \to \mathcal{D}$ we take $\Psi(F)$ to be the restriction of the functor F to the non-basepoint morphisms. Notice that for any f with $F(f)$ the $*$ in $\mathcal{D}$, we must then leave $\Psi(F)(f)$ undefined. Also, the unit map to the smash product for a non-zero object implies the identity map is not $*$. Since the functor F sends identities to identities (and we have removed any zero objects), this means $\Psi(F)$ will also be defined on all identities as required.

Once again, the consistency of composition implies $\Psi : \mathrm{Cat}_* \to Pre\,\mathrm{Cat}$ is then a functor as well. Notice Ψ will be particularly well-behaved when restricted to the full subcategory $\widetilde{\mathrm{Cat}_*} \subsetneq \mathrm{Cat}_*$ consisting of elements which do not contain zero objects.

A.3. The Replacement Results.

Theorem A.7. *There is an (adjoint) equivalence of categories between $Pre\,\mathrm{Cat}$ and $\widetilde{\mathrm{Cat}_*}$.*

Proof. It should be clear that $\Psi\Phi$ is the identity functor from $Pre\,\mathrm{Cat}$ to itself by construction (since there can be no zero objects in the image of Φ).

To see $\Phi\Psi$ is equivalent to the identity on $\widetilde{\mathrm{Cat}_*}$, we simply notice that Ψ only forgets about compositions which go to the basepoint and the basepoint map itself. Since everything about the basepoint can be deduced from the enrichment, this does not actually lose any information. In other words, $\Phi\Psi$ is simply the process of renaming the basepoint as the illegal point $*$ which is (enriched) isomorphic to the original category. □

The theorem is actually quite satisfying as it says that any question about the category of precategories may be rephrased in terms of Cat_*. Thus, the functor Φ may be used to bring precategories into the world of established category theory without losing any important information visible in $Pre\,\mathrm{Cat}$, at the mild price of working with the enriched notions.

The relevance of precategories to our framework is straightforward: our dialog models could be chosen as precategories, since representing legal sequences of locutions using a partial composition law, and then assuming any undefined compositions are illegal would be a suitable way to represent them as well. In philosophical terms, we can take the morphisms whose composition we refuse to define and instead define them as an illegal morphism. Regardless, we have refused to give the same information in either description.

There is also an enriched functor $\Theta : \mathrm{Cat}_* \to \widehat{\mathrm{Cat}_*}$ which adds a zero object to each category, where $\widehat{\mathrm{Cat}_*} \subsetneq \mathrm{Cat}_*$ denotes the full subcategory whose objects contain at least one zero object. Notice that, by A.5.2 the inclusion $\mathcal{C} \to \Theta\mathcal{C}$ will be an enriched equivalence in cases where $\mathcal{C} \in \widehat{\mathrm{Cat}_*}$. This actually strengthens the following related equivalence result as well.

Theorem A.8. *The restricted functor* $\Theta : \widetilde{\mathrm{Cat}_*} \to \widehat{\mathrm{Cat}_*}$ *is an equivalence of categories onto its essential image.*

Proof. Remark A.5.3 implies that any enriched functor will send a zero object to another zero object, since the composite $S^0 \to \mathcal{C}(0,0) \to \mathcal{D}(F(0), F(0))$ must remain a constant map. Thus, any $F : \mathcal{C} \to \mathcal{D}$ in $\widetilde{\mathrm{Cat}_*}$ extends immediately to $\Theta(F)$, but the extension is unique since $\Theta\mathcal{D}$ contains a single zero object (as $\mathcal{D}$ contains none). It follows that Θ is an equivalence onto its essential image. □

By the remarks prior to the theorem, every object in $\widehat{\mathrm{Cat}_*}$ is equivalent (but generally not isomorphic) to one in the image of Θ (even in the image of the restriction to $\widetilde{\mathrm{Cat}_*}$).

By combining the two theorems, one could just as well replace precategories with categories with zero objects (leaving the enrichment implicit as in Remark A.5.1) and functors which preserve zero objects. While such objects may appear simpler than categories enriched over pointed sets, we return to the goal of working with a mature mathematical theory. The current authors are aware of no references systematically detailing constructions such as (co)limits for categories with zero objects and functors which preserve them.

A more subtle question is the mechanics of this replacement. When one starts out with a precategory, the composite says rather than leaving $g \circ f$ undefined, we now require the square

(18)
$$\begin{tikzcd} A \arrow[r, "f"] \arrow[d] & B \arrow[d, "g"] \\ 0 \arrow[r] & C \end{tikzcd}$$

to commute in the associated category with zero object. While the theorems imply this transition is well-behaved, it seems somehow less appealing. Since all three categories are equivalent, one can choose based on convenience how to translate any formal statement. Thus, one could choose to model protocols as arrows in either $\widehat{\mathrm{Cat}_*}$ or $\mathit{Pre}\,\mathrm{Cat}$ and translate all of our statements to that context.

We should point out this combined replacement process was understood at least by [32, Example 2.8], but those authors sought to work out the most general theory of precategories they could in that article. In order to provide a theory with the most stable underpinnings, we instead set out to work with the best-known context we could find.

A.4. **Detecting Categories and Functors.** Since the category of all small categories (and ordinary functors) is (via Φ) a subcategory of Cat_*, one might want to be able to distinguish categories from precategories within the image of Φ. More importantly, one might have a prefunctor between ordinary categories and want to know if it is a functor. In fact, Φ restricts to just the usual "adding a basepoint to the morphisms" operation considered in Example 2.8.3. More abstractly, (Φ, Ψ) is the change of base pair between ordinary categories and those enriched over pointed sets associated to the symmetric monoidal functor given by adding a basepoint.

Both identification questions above can be determined by looking at preimages of basepoints, via the following simple notion.

Definition A.9.
(1) Call a map of pointed sets non-collapsing if the preimage of the basepoint is precisely the basepoint.
(2) Call an enriched functor $F : \mathcal{C} \to \mathcal{D}$ in Cat_* non-collapsing if the map
$$F_{A,B} : \mathcal{C}(A, B) \to \mathcal{D}(F(A), F(B))$$
is non-collapsing for all pairs of objects A and B of $\mathcal{C}$.
(3) Call a $\mathcal{C} \in \mathrm{Cat}_*$ non-collapsing if the composition law
$$\mathcal{C}(B, C) \wedge \mathcal{C}(A, B) \to \mathcal{C}(A, C)$$
is a non-collapsing map for each triple of objects A, B, and C in $\mathcal{C}$.

We can now describe the ordinary categories as those elements of Cat_* which are non-collapsing, and similarly for ordinary functors.

Lemma A.10. *The category of small categories and functors is equivalent to the category of non-collapsing* $\mathcal{C} \in \mathrm{Cat}_*$ *and non-collapsing enriched functors in* Cat_*.

The equivalence is still given by (Φ, Ψ) keeping in mind that Φ is just adding a basepoint to every morphism set for an ordinary category.

In fact, the non-collapsing enriched functors in Cat_* in general correspond precisely to the functors between strong precategories in the sense of [32].

References

[1] S. Abramsky. Semantics of interaction: an introduction to game semantics. In A. M. Pitts and P. Dybjer, editors, *Semantics and Logics of Computation*, pages 1–31. Cambridge University Press, Cambridge, UK, 1997.

[2] L. Amgoud, N. Maudet, and S. Parsons. Modelling dialogues using argumentation. In E. Durfee, editor, *Proceedings of the International Conference on Multi-Agent Systems (ICMAS 2000)*, pages 31–38, Boston, MA, USA, 2000. IEEE Press.

[3] L. Amgoud, S. Parsons, and N. Maudet. Arguments, dialogue, and negotiation. In W. Horn, editor, *Proceedings of the European Conference on Artificial Intelligence (ECAI 2000)*, pages 338–342, Berlin, Germany, 2000. IOS Press.

[4] Aristotle. *Topics*. Clarendon Press, Oxford, UK, 1928. (W. D. Ross, Editor).

[5] T. J. M. Bench-Capon, P. E. Dunne, and P. H. Leng. Interacting with knowledge-based systems through dialogue games. In *Proceedings of the Eleventh International Conference on Expert Systems and Applications*, pages 123–140, Avignon, France, 1991.

[6] F. Borceux. *Handbook of Categorical Algebra*, volume 2. Cambridge University Press, 1994.

[7] A. Byde. Applying evolutionary game theory to auction mechanism design. In *Proceedings of the 4th ACM Conference on Electronic Commerce*, pages 192–193. ACM Press, 2003.

[8] P. Cramton, Y. Shoham, and R. Steinberg. *Combinatorial Auctions*. MIT Press, 2006.

[9] U. Endriss. Monotonic concession protocols for multilateral negotiation. In P. Stone and G. Weiss, editors, *Proceedings of the 5th International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS-2006)*, pages 392–399. ACM Press, May 2006.

[10] R. Fernández and U. Endriss. Abstract models for dialogue protocols. *Journal of Logic, Language and Information*, 16(2):121–140, 2007.

[11] FIPA. Communicative Act Library Specification. Standard SC00037J, Foundation for Intelligent Physical Agents, 3 December 2002.

[12] F. Guala. Building economic machines: The FCC Auctions. *Studies in the History and Philosophy of Science*, 32(3):453–477, 2001.

[13] C. L. Hamblin. *Fallacies*. Methuen, London, UK, 1970.

[14] J. Hintikka. *The Game of Language: Studies in Game-Theoretical Semantics and Its Applications*, volume 22 of *Synthese Language Library*. D. Reidel, Dordrecht, The Netherlands, 1983.

[15] M. W. Johnson. On pointed enrichments and illegal compositions. Technical Report ULCS-03-010, Department of Computer Science, University of Liverpool, Liverpool, UK, 2003.

[16] M. W. Johnson, P. McBurney, and S. Parsons. When are two protocols the same? In M-P. Huget, editor, *Communication in Multi-Agent Systems: Agent Communication Languages and Conversation Policies*, Lecture Notes in Artificial Intelligence 2650, pages 253–268. Springer, Berlin, 2003.

[17] M. W. Johnson, P. McBurney, and S. Parsons. A mathematical model of dialog. volume 141 of *Electronic Notes in Theoretical Computer Science*, pages 33–48, 2005.

[18] G. M. Kelly. *Basic Concepts of Enriched Category Theory*. London Mathematical Society Lecture Notes 64. Cambridge University Press, Cambridge, UK, 1982.

[19] V. Krishna. *Auction Theory*. Academic Press, San Diego, CA, USA, 2002.

[20] K. Lehmann and D. Moldt. Modelling and analysis of agent protocols with Petri Nets. In G. Lindemann, J. Denzinger, I. J. Timm, and R. Unland, editors, *Multiagent System Technologies*, volume 3187 of *Lecture Notes in Computer Science*, pages 85–98. Springer, Erfurt, Germany, 2004.

[21] J. A. Levin and J. A. Moore. Dialogue-games: metacommunications structures for natural language interaction. *Cognitive Science*, 1(4):395–420, 1978.

[22] P. Lorenzen and K. Lorenz. *Dialogische Logik*. Wissenschaftliche Buchgesellschaft, Darmstadt, Germany, 1978.

[23] P. McBurney, D. Hitchcock, and S. Parsons. The eightfold way of deliberation dialogue. *International Journal of Intelligent Systems*, 22(1):95–132, 2007.

[24] P. McBurney and S. Parsons. Representing epistemic uncertainty by means of dialectical argumentation. *Annals of Mathematics and Artificial Intelligence*, 32(1–4):125–169, 2001.

[25] P. McBurney and S. Parsons. Games that agents play: A formal framework for dialogues between autonomous agents. *Journal of Logic, Language and Information*, 11(3):315–334, 2002.

[26] P. McBurney and S. Parsons. Retraction and revocation in agent deliberation dialogs. *Argumentation*, 21(4), 2007. *In press.*

[27] T. Miller and P. McBurney. Using constraints and process algebra for specification of first-class agent interaction protocols. In G. O'Hare, M. O'Grady, O. Dikenelli, and A. Ricci, editors, *Post-proceedings of the Seventh International Workshop on Engineering Societies in the Agents World*, number 4457 in LNAI, pages 245–264, 2007.

[28] P. Mittelstaedt. *Quantum Logic.* D. Reidel, Dordrecht, The Netherlands, 1979.

[29] M. J. Osborne and A. Rubinstein. *A Course in Game Theory.* MIT Press, Cambridge, MA, USA, 1994.

[30] S. Parsons, C. Sierra, and N. R. Jennings. Agents that reason and negotiate by arguing. *Journal of Logic and Computation*, 8(3):261–292, 1998.

[31] J. Pitt and A. Mamdani. Communication protocols in multi-agent systems. In *Proceedings of the Agents-1999 Workshop on Specifying and Implementing Conversation Policies*, Seattle, Washington, 1999.

[32] L. Schröder and P. Mateus. Universal aspects of probabilistic automata. *Mathematical Structures in Computer Science*, 12(4):481–512, 2002.

[33] M. P. Singh. A social semantics for agent communications languages. In F. Dignum, B. Chaib-draa, and H. Weigand, editors, *Proceedings of the Workshop on Agent Communication Languages, International Joint Conference on Artificial Intelligence (IJCAI-99)*, Berlin, Germany, 2000. Springer.

[34] P. V. Spade. Recent research on medieval logic. *Synthese*, 40:3–18, 1979.

[35] D. N. Walton and E. C. W. Krabbe. *Commitment in Dialogue: Basic Concepts of Interpersonal Reasoning.* State University of New York Press, Albany, NY, USA, 1995.

[36] F. Zeuthen. *Problems of Monopoly and Economic Warfare.* G. Routledge & Sons, London, UK, 1930.

Department of Mathematics, Penn State Altoona, Altoona, PA 16601-3760
E-mail address: mwj3@psu.edu

Department of Computer Science, University of Liverpool, Liverpool, UK, L69 3BX
E-mail address: tim@csc.liv.ac.uk, p.j.mcburney@csc.liv.ac.uk