Probabilistic Record Type Lattices for Incremental Reference Processing[*]

Julian Hough and Matthew Purver

1 Introduction

This chapter[1] addresses the issue of incorporating probabilistic type-theoretic inference into an incremental dialogue framework, using processing of referring expressions (that is, their interpretation and generation) as a test case. Within our framework, we model reference processing in a psycholinguistically plausible way– that is, in a strictly left-to-right, word-by-word, incremental fashion. We additionally show how the model is capable of processing disfluent referring expressions while making use of the information the disfluency conveys to reflect psycholinguistic results on the effect on processing speed.

Our model incorporates probabilistic Type Theory with Records (Cooper et al., 2014) and order-theoretic models of probability and information theory (Knuth, 2005) into a formal dialogue system that reflects the psycholinguistic evidence.

In Section 2 we introduce the challenge of modelling reference processing, and overview some previous approaches. In Section 3 we describe the semantic and dialogue framework we use and how we enrich it with probabilistic record type lattices. In Section 4 we describe how our model can simulate psycholinguistic results in reference processing and we finish with a discussion and conclusion.

2 Background on reference processing models

There has been significant work on simple referential communication games in psycholinguistics and computational and formal models of communication. These reference games are usually posed as a human-human situation where an instruction giver and instruction follower have access to the same visual scene with simple objects, and the instructor produces an utterance to make the instructee select the object(s) he or she refers to. A typical referring expression (RE) produced by an instruction giver in a simple domain of coloured shapes would be "the yellow square"– these simple noun phrases are the ones we focus on here.

From a speech production perspective, Levelt (1989)'s seminal work modelled speaker strategies for producing REs in such a simple object naming game. He showed

[*]This is a pre-print of an article published in *Modern Perspectives in Type-Theoretical Semantics*, Volume 98 of the series *Studies in Linguistics and Philosophy* pp 189-222, Springer International Publishing. The final authenticated version is available online at: `https://doi.org/10.1007/978-3-319-50422-3_8`

[1]Much of the work here is drawn from Hough and Purver (2014b) and Hough and Purver (2014a).

how the production process could be split into three separate stages of *conceptualization* (choosing the elements or properties of the objects the speaker intends to vocalize), *syntactic formulation* (choosing the lexical items to convey the content, linearising their order and checking lexical agreements) and *articulation* (speaking). He also showed how speakers use informationally redundant features of the target object, or *over-specification*, violating Grice's Maxim of Quantity that speakers should say no more than is necessary to convey their communicative intention (Grice, 1975), a result that has been supported in subsequent accounts.

In the natural language generation (NLG) community, referring expression generation (REG) has been widely studied (see (Krahmer and Van Deemter, 2012) for a comprehensive survey). The incremental algorithm (IA) (Dale and Reiter, 1995), the most well-known REG algorithm, is an iterative feature selection procedure that operates on logical properties of objects in a visual scene (such as `colour=yellow` for yellow objects). The IA computes the *distractor* set of referents (i.e. those not intended for selection) which each property used in a RE could cause to be inferred. From this, the IA gives a utility value to each property based on its ability to determine the referent in a non-greedy manner– it iterates over the properties in terms of a fixed preference order (whereby say, colour properties would ranked as preferred over shape properties, and would therefore be used first), not just their ability to determine the referent uniquely, and adds them if they reduce the number of distractors. The IA stops when the combination of properties determines the referent unambiguously. This was designed to be consistent with over-specification phenomena in that certain types of properties will be selected if they have any discrimination ability, even if the final RE generated is not optimally brief. More recently Frank and Goodman (2012) investigate property preference and discriminatory power empirically in a Bayesian model of REG based on information-theoretic surprisal in terms of how much REs reduce uncertainty about their intended referent, a measure which they claim correlates strongly to human judgements of which RE best describes a given target (carried out in a multiple choice study rather than allowing open answers).

2.1 Incrementality

The aspect of reference processing we focus on here is *incremental processing*. That is to say, as opposed to modelling the selection of linguistic content in REG and the interpretation of REs on the level of complete utterances, we wish to model the inference listeners make on a strictly *left-to-right*, *word-by-word* basis. We assume the requirements of incremental semantics are to yield the maximal information from an utterance as it is processed (Hough et al., 2015).

Existing models closest to our own are the incremental REG models described by Guhe (2007) and Fernández (2013) and the incremental reference resolution model proposed by Kennington and Schlangen (2014). Guhe (2007)'s approach to REG is to model a fine-grained incremental conceptualizer which passes conceptual increments (partial logical forms) to an incremental syntactic formulator, allowing piece-meal processing. While not focussing on reference processing, the incremental conceptualization is a point of departure we use in our model below.

Fernández (2013), on the other hand, takes a more purely linguistic, syntax-level perspective. Fernández sketches a novel solution to modelling over-specification, arguing the phenomenon may be caused more by the affordances of incremental left-to-right word-by-word information processing in different languages, rather than salience of properties as proposed by Dale and Reiter (1995)'s IA.[2] She argues properties seemingly redundant when considering the RE as a whole unit may in fact be important when considering their incremental word-by-word contribution to reference resolution, that is, their *incremental informativity*. The paper gives cross-linguistic evidence from Spanish speakers based on Rubio-Fernández (2011)'s experimental results, arguing over-generation of redundant adjectives is less common in languages where such adjectives can be added post-nominally.. She exemplifies a domain where the only red lamp in a scene is the referent, and it is possible to individuate it from its object type (i.e. the property that it is a lamp) alone, where describing its colour is redundant, however still partially discriminative as there is another red object in the domain. For an English speaker "the red lamp" would be a typical over-specified description, whereas in Spanish "la lámpara roja" ("the red lamp") would be less common, and "la lámpara" would be a more common RE. This cross-linguistic difference is due to the fact the post-nominal "roja" does not add any reference information incrementally after "la lámpara", which on its own has sufficient discriminatory power, while the English "red", although not uniquely determining the referent, narrows the reference set and so is incrementally informative. Fernández uses this example to sketch a REG system that uses a variant of Dale and Reiter (1995)'s IA for content selection interleaved with a TAG-based syntactic formulator that is strictly left-to-right incremental in its tree construction. She emphasizes the importance of tightly coupling the REG procedure with an interpretation component of a dialogue system but does not give details of how this could be done.

On the interpretation side of reference, reference resolution, Kennington and Schlangen (2014)'s incremental system models the role of the hearer or instructee. The system continuously incrementally outputs a distribution of possible referents conditioning on the logical values of properties of the objects in the scene and the words used to refer to those properties spoken by the instructor. The model forms part of *situated* dialogue processing, as it continuously updates its referent distributions based on perceptual data, and not necessarily just linguistic data. The conditional probabilities are calculated using a generative model (of the speaker) and implemented using Markov Logic networks. Kennington et al. (2014)'s development of the model uses incremental semantic representations built up word-by-word by an incremental rMRS (robust Minimal Recursion Semantics) parser (Peldszus et al., 2012) as part of the property set it conditions on, boosting results from using simple n-gram models.

Motivated by incremental approaches such as those just described, this chapter presents an incremental dialogue-motivated account of reference identification which models the speaker in terms of incremental NLG and the hearer in terms of incremental interpretation of utterances in reference identification games.

In addition to modelling incremental processing of fluent utterances, the model aims to reflect the evidence from Brennan and Schober (2001)'s evidence that *self-repair* can speed up semantic processing (or at least reference identification) in such games. An incorrect RE being partly vocalized and then repaired in the instructions in conjunction with a filled pause interregnum "uh" (e.g. "the yell-, uh, purple square") yields quicker response times to select the correct object from the onset of the target ("purple") than in the case of the fluent instructions ("the purple square"), with no significant effect on accuracy– we return to this example below.

[2]It is worth noting here that the IA deals with incrementality in property selection of the avoidance of re-computation type, rather than word-by-word surface-level incrementality.

$$R_1 : \begin{bmatrix} l_1 & : & T_1 \\ l_2 & : & T_2 \\ l_3 & : & T_3(l_1) \end{bmatrix} \quad R_2 : \begin{bmatrix} l_1 & : & T_1 \\ l_2 & : & T_{2'} \end{bmatrix} \quad R_3 : []$$

$$S_1 = \begin{bmatrix} l_1 & = & a \\ l_2 & = & b \\ l_3 & = & c \end{bmatrix} \quad S_2 = \begin{bmatrix} l_1 & = & a \\ l_2 & = & b' \end{bmatrix} \quad S_3 = []$$

Figure 1: Example TTR record types (top row) and records (bottom)

3 Incorporating probability into type judgements and dialogue states

To meet the challenges outlined, we present a reversible reference processing model (i.e. one that works both in interpretation and generation with minor parametric changes), situated within the incremental dialogue framework DyLan ('Dynamics of Language', Purver et al., 2011). While DyLan is capable of incremental semantic processing, we would like a dialogue model to be able to reflect the uncertainty of reference as a referring expression progresses and allow probabilistic reasoning about the reference situation in general, which is an uncontroversial view for modern cognitive models (Chater et al., 2006). For this to become possible, the model should generate distributions over type judgements over a reference situation rather than just allow binary type judgements. In the following subsections we explain the probabilistic type theoretic and lattice theoretic mathematical tools we use to achieve incremental probabilistic type judgements in DyLan.

3.1 Probabilistic TTR

Firstly, we briefly describe the chosen semantic representation framework for our model, Type Theory with Records (TTR, Cooper, 2005, 2012).[3]

In TTR, the principal logical form of interest is the *record type* (abbreviated 'RT' largely from here), consisting of sets of *fields* of the form $[\, l : T \,]$ containing a label l, from a set of labels, and a type T, from a type ordering, representing the central type-theoretic judgement $l : T$, that an object labelled l is of type T. RTs can be witnessed (i.e. judged as inhabited) by *records* of that type, where a record is a semantic object structured isomorphically to a RT, consisting of sets of label-value pairs $[\, l = v \,]$. See Figure 1 for examples of RTs and records.

The central type judgement in TTR that a record s is of record type R, i.e. $s : R$, can be made from the component type judgements of individual fields of R, e.g. the one-field record $[\, l = v \,]$ is of record type $[\, l : T' \,]$ just in case type v is of type T'. This single-field RT check is generalisable to records and RTs with multiple fields: a record s is of RT R if s includes fields with labels matching those in the fields of R in a one-to-one relation, such that all fields in R are matched to a field in s, and all label-matched

[3]We only introduce the elements of TTR relevant to the phenomena discussed below. See Cooper (2012) and Cooper (fcmg) (Chapter XXX, this volume) for a detailed formal description.

fields in s have a value of the same type as their corresponding field in R (or, their value is a subtype of the corresponding type in R, see below). A record type check can be defined as in (1). Thus it is possible for the record s to have more fields than RT R and for $s : R$ to still hold, but not vice-versa: $s : R$ cannot hold if RT R has more fields than the record s.

(1) *Record type check*:
For a record s and and record type R, $s : R$ is true iff for every field $[\, l : T \,]$ in R there is a field $[\, l = v \,]$ in s such that $v : T$.

Fields can have values representing predicate type (*PType*) judgements, such as $l_3 : T_3(l_1)$ in Figure 1, and consequently fields can be *dependent* on fields preceding them (i.e. represented graphically higher up) in the RT, e.g. in Figure 1, l_1 is bound in the *PType* judgement field l_3, so l_3 depends on l_1.

Semantically, the boolean judgement of whether type judgements are *true* or *false* is determined by a model, or *type system* (Cooper, 2012). A type system consists of (i) a partially ordered set of types (*type hierarchy*), ordered by the subtype relation as will be explained further below, of which the supremum is the type $Type$, (ii) a set of labelled objects (*type inhabitants*) which is disjoint from the type hierarchy, and (iii) a valuation function $A(T)$ which maps each type T in the type hierarchy to a subset of the set of type inhabitants. Therefore, in terms of judgements, $l : T'$ is true iff the object labelled l in the type system is a member of the set $A(T')$; i.e. $l : T'$ iff $l \in A(T')$. In terms of record types and records, if the field $l : T'$ in a RT is true according to the type system, then the judgement in a record $l = v$ is true iff v is of type T'; i.e. given $l : T'$ is true, then $l = v$ is true iff $v : T'$ (where $v : T'$ is true iff $v \in A(T')$ is true). In addition to judgements that atomic objects are of a given type, and records are of a given record type, it is possible to make judgements that a type is of another type inductively from their position in a type hierarchy.

3.1.1 The subtype relation

In line with the semantic interpretation requirement of incremental interpretation and NLG models yielding the maximal information from an utterance as it is processed (Hough et al., 2015), a strongly incremental account will require checking whether RTs under construction are consistent with the RTs representing domain concept RTs provided by a conceptualizer incrementally. To do this we make use of the $\sqsubseteq$ ('is a subtype of') relation, which is *subsumptive* in TTR, that is if RT R_1 is a subtype of RT R_2 (i.e. $R_1 \sqsubseteq R_2$) then there are no objects of type R_1 that are not of type R_2, or in the sense of the phrase from Description Logic, R_1 *is subsumed by* R_2.

Operationally, subtype relation checking can be defined for RTs in terms of fields as simply: $R_1 \sqsubseteq R_2$ iff for all fields $[\, l : T_2 \,]$ in R_2, R_1 contains $[\, l : T_1 \,]$ where $T_1 \sqsubseteq T_2$. In Figure 1, it will be the case that $R_1 \sqsubseteq R_3$, $R_2 \sqsubseteq R_3$ and $R_1 \sqsubseteq R_2$ iff $T_2 \sqsubseteq T_{2'}$. The transitive nature of this relation (i.e. iff $R_1 \sqsubseteq R_2$ and $R_2 \sqsubseteq R_3$ then $R_1 \sqsubseteq R_3$) can be used effectively for type-theoretic inference as will be described below. An operational definition for a subtype check, adapted from (Fernández, 2006, p.96), is given in (2). If R_1 has n fields and R_2 has m fields, assuming naively a uniform cost for each type check on the type hierarchy, the complexity of this check can be $O(n \times m)$ in the worst case where every field in one RT needs to be compared against every field in the other.[4] Note that the label-matching conventions for type checking are extremely

[4]The cost of the subtype check for a field may be more costly if it is dependent (i.e. a $PType$, however this is not important for the discussion here.

useful for computability here, as the complexity would be far greater if unconstrained re-labelling was permitted.

(2) *Subtype relation check*:
For record types R_1 and R_2, $R_1 \sqsubseteq R_2$ holds just in case for each field $[\, l : T_2 \,]$ in R_2 there is a field $[\, l : T_1 \,]$ in R_1 such that $T_1 \sqsubseteq T_2$. This relation is reflexive and transitive.

While we do not discuss the full stratified type hierarchy for TTR here, we note that for all types, $T_1 \sqsubseteq T_2$ implies that $T_1 : T_2$, but does not imply $T_2 : T_1$ unless $T_2 \sqsubseteq T_1$, a consistency that extends to RT judgements. There are many complexities here which we will not deal with as regards type stratification– again see Cooper (2012) for details. We do not believe these complexities affect TTR's suitability for dialogue modelling and the discussion here.

We use the notion of *manifest* (singleton) types, e.g. T_a, the type T of which only a is a witness. Here, we represent manifest RT fields such as $[\, l : T_a \,]$ where $T_a \sqsubseteq T$ by using the syntactic sugar $[\, l_{=a} : T \,]$ following Cooper (2012). The subtype relation effectively allows progressive instantiation of fields in a monotonic fashion, as the addition of fields to an RT R, and the manifestation of fields in R, leads to R' where $R' \sqsubseteq R$. This is practically useful for an incremental dialogue system in terms of meeting the strong incremental interpretation and minimization of re-computation requirements of incremental semantics (Hough et al., 2015) and for other reasons of incremental utterance processing as we will explain.

3.1.2 Meet types and the merge operation

We make use of the *meet type* of two or more RTs and an operation to yield an *equivalent* RT to the meet type.[5] As Cooper (2012) explains, the meet type of two RTs results in a type that is no longer an RT, even if the objects it witnesses are records, however, a RT extensionally equivalent to the meet type of two RTs R_1 and R_2 is the yield of a merge operation $R_1 \curlywedge R_2$ (Larsson, 2010). Operationally, in the simplest case merge can be characterized as union of fields of two RTs, for example for R_1 and R_2 in (3). In the event of label-type clashes between labels in two RTs (i.e. cases where R_1 contains $l_1 : T_1$ and R_2 contains $l_1 : T_2$), in this chapter we assume all examples like this $T1$ and $T2$ are incompatible (disjoint in the type hierarchy), in which case the resulting $R_1 \curlywedge R_2$ is $\bot$.

$$\begin{gathered}\text{if } R_1 = \begin{bmatrix} l_1 & : & T_1 \\ l_2 & : & T_2 \end{bmatrix} \text{and } R_2 = \begin{bmatrix} l_2 & : & T_2 \\ l_3 & : & T_3 \end{bmatrix} \\ R_1 \wedge R_2 \equiv R_1 \curlywedge R_2 = \begin{bmatrix} l_1 & : & T_1 \\ l_2 & : & T_2 \\ l_3 & : & T_3 \end{bmatrix}\end{gathered} \tag{3}$$

3.1.3 Join types and the minimal common supertype operation

Here we also define a dual of the merge operation, not found in the TTR literature, which is necessary for the analysis below: what we call the *minimal common supertype*

[5]In TTR two types T_1 and T_2 are equivalent iff for any object a in the domain such that iff $a : T_1$ then $a : T_2$ and vice-versa. This extends to record types.

operator $\curlyvee$. While technically the minimal common supertype of R_1 and R_2 is the *join type* $R_1 \vee R_2$, here, for reasons that will become apparent below in the discussion on type lattices, we are also interested in isolating the minimal common supertype of two RTs R_1 and R_2 which is still a non-disjunctive RT, which, when there are no clashing type judgements, amounts to field intersection as below in (4). Where there are label-type clashes between fields in two RTs, i.e. where R_1 contains $l_1 : T_1$ and R_2 contains $l_1 : T_2$, in the examples in this chapter we assume the minimal common supertype of T_1 and T_2 is the most general type $Type$, and in these cases the field is omitted in the result of $R_1 \curlyvee R_2$. Note the minimal common supertype RT of multiple RTs is generally *not* equivalent to their join type as will be explained.

$$\text{if } R_1 = \begin{bmatrix} l_1 & : & T_1 \\ l_2 & : & T_2 \end{bmatrix} \text{and } R_2 = \begin{bmatrix} l_2 & : & T_2 \\ l_3 & : & T_3 \end{bmatrix} \\ R_1 \curlyvee R_2 = \begin{bmatrix} l_2 & : & T_2 \end{bmatrix} \tag{4}$$

3.1.4 Going probabilistic

While classical type theory has been the predominant mathematical framework in natural language semantics for many years (Montague, 1974, inter alia), it is only recently that probabilistic type theory has been discussed for this purpose. Similarly, type-theoretic representations have been used within dialogue models (Ginzburg, 2012); and probabilistic modelling is common in dialogue systems (Young et al., 2013, inter alia), but combinations of the two remain scarce. In this chapter this connection is made, taking Cooper et al. (2014, 2015)'s probabilistic TTR as the principal point of departure for modelling incremental inference in dialogue as described above.

At the time of writing there had been no methods for practical integration of probabilistic type-theoretic inference into a dialogue system; here we discuss computationally efficient methods for implementation. We argue for their efficacy in simple referential communication domains, but simultaneously suggest the methods could be extended to larger domains and additionally used for real-time learning in future work.

Given that TTR has a highly flexible rich type system, variants have been considered with type judgements corresponding to real number valued perceptual data used in conjunction with linguistic context, such as those representing visual information (Larsson, 2011; Dobnik et al., 2013), demonstrating its potential for situated, embodied and multi-modal dialogue systems. The possibility of integration of perceptron learning (Larsson, 2011) and Naive Bayes learning (Cooper et al., 2014) into TTR show how linguistic processing and probabilistic conceptual inference can be treated in a uniform way within the same formal system.

Probabilistic TTR as described by Cooper et al. (2014, 2015) replaces the categorical $s : T$ judgement, the judgement that it is *true* or *false* that an object s is of type T, with the real number valued $p(s : T) = v$ where $v \in [0,1]$.[6] The authors show how standard probability theoretic and Bayesian equations can be applied to type judgements and how an agent might learn from experience in a simple classification game.

[6]Several people we have discussed this with are not convinced a type judgement can be probabilistic. We remain agnostic to the plausibility of a non-conditional judgement such as this one being real-valued, however we do think real-valued *conditional* probability judgements are realistic. We thank David Schlangen and Arash Eshghi for discussions on this. The view we set out below can be cashed out purely in terms of conditional type judgements, however the conditional judgement may at times be notationally suppressed where appropriate and in a consistent manner– these cases will be noted where they crop up.

In their example, the agent is presented with instances of a situation with associated type judgements and it learns with each round by updating its set of probabilistic type judgements to best predict the type of object in focus – in this case updating the probability judgement that something is an apple given its observed colour and shape, i.e. $p(s : T_{apple}|s : T_{Shp}, s : T_{Col})$ where $Shp \in \{shp_1, shp_2\}$ and $Col \in \{col_1, col_2\}$. From a cognitive modelling perspective, these judgements can be viewed as learning concepts from probabilistic perceptual information, and if framed as a language acquisition scenario these concepts could be associated with words. We use similar methods in the toy reference domain below, but show how complex type judgements can be constructed efficiently, and how conditional probabilistic judgements can be made incrementally without exhaustive iteration through individual type classifiers, as the mechanisms in Cooper et al. (2014, 2015) and Kennington and Schlangen (2014)'s models require.

For the exposition of probabilistic TTR, we repeat some of Cooper et al.'s calculations and show some equivalences not described by the authors. We describe our efficient order-theoretic and graphical methods for generating and incrementally retrieving probabilities in Section 3.2.

Cooper et al. (2015), under the assumption that type judgements can be real-valued, define conditional probability of an object being of type R_2 given it is of type R_1 as in (5). This is the most important judgement in probabilistic TTR, due to the framework's motivation: an agent can judge a situation s is of a given situation type, given the evidence that it is of other situation types. In this way an agent is positioned as a classifier of situations given the evidence available to it. Here we assume s can be a record, not just a basic type, and so R_1 and R_2 can be record types.

$$p(s : R_2|s : R_1) = \frac{p(s : R_1 \wedge R_2)}{p(s : R_1)} \tag{5}$$

Given classical probability theoretic equivalences, they define the probability of a situation being of a meet (conjunctive) and join (disjunctive) types of two basic types or RTs in terms of the standard *product* rule in (6) and *sum* rule (7) in probability theory:

$$p(s : R_1 \wedge R_2) = p(s : R_1)p(s : R_2|s : R_1) \tag{6}$$

$$p(s : R_1 \vee R_2) = p(s : R_1) + p(s : R_2) - p(s : R_1 \wedge R_2) \tag{7}$$

It is practically useful, as we will describe below, that the join probability can be computed in terms of the meet. Given the classical probability theoretic definitions for the meet and the join type they show it is possible to sustain the below:

$$\begin{aligned} p(s : R_1 \wedge R_2) &\leq p(s : R_1) \\ p(s : R_1 \wedge R_2) &\leq p(s : R_2) \\ p(s : R_1) &\leq p(s : R_1 \vee R_2) \\ p(s : R_2) &\leq p(s : R_1 \vee R_2) \end{aligned} \tag{8}$$

Also, there are equivalences between meet types, join types and subtypes in terms of type judgements as described above, in that assuming if $R_1 \sqsubseteq R_2$ then $p(s : R_2|s : R_1) = 1$, we have:

$$\begin{aligned}
&\text{if } R_1 \sqsubseteq R_2 \\
p(s : R_1 \wedge R_2) &= p(s : R_1) \\
p(s : R_1 \vee R_2) &= p(s : R_2) \\
p(s : R_1) &\leq p(s : R_2)
\end{aligned} \tag{9}$$

We return to an explanation for these classical probability equations holding within probabilistic TTR below in terms of record type lattices. We make a remark here that the meet type probability of two conjuncts is the same as the probability of the RT result from the merge operation of those conjuncts in (10). This is the case due to the extensional equivalence of a (non record type) meet type $R_1 \wedge R_2$ and the resulting record type from the operation $R_1 ⩓ R_2$ as shown in (1). For this reason, all the $\wedge$ conjunctions in the above equations can be replaced by $⩓$ and the equations will still hold. The same is not true of the relationship between the join $\vee$ type and the $⩔$ operation as we will explain.

$$p(s : R_1 \wedge R_2) = p(s : R_1 ⩓ R_2) \tag{10}$$

Through the subtype relation, merge operator and minimal common super type operator, we will now be able to show why these classical probability theoretic equations hold in TTR, due to the structure of record type lattices, and how these are useful objects for incremental dialogue processing.

3.2 Probabilistic Record Type lattices

To support efficient reference processing, we represent dialogue domain concepts as partially ordered sets (*posets*) of RT judgements. This is inspired by the use of RT lattices in automatic grammar learning by Eshghi et al. (2013), however here they are fleshed out in a formal way to provide an interface to a general reasoning system and probabilistic TTR.

A poset has several advantages over an unordered set of un-decomposed record types: the possibility of incremental type checking; increased speed of type checking, particularly for pairs of or multiple type judgements; immediate use of type judgements to guide system decisions; inference from negation; efficient construction of a question under discussion (QUD) structure that includes question relevance values contingent on probability; and modelling the learning of type judgements. We leave the final two challenges for future work, but discuss the others here.

From a set of RTs which are semantically disjoint (i.e. for any two RTs in this set, their record inhabitants in the type system are disjoint), it is possible to construct a valid record type lattice. As per set-theoretic lattices, RT lattices can be visualised as Hasse diagrams such as those in Figure 2, however here the ordering arrows show $\sqsubseteq$ ('is a subtype of') relations from descendant to ancestor nodes, rather than the normal set inclusion relation.

To characterize a RT lattice L ordered by $\sqsubseteq$, we adapt Knuth (2005)'s description of lattices in line with standard order theory. L is a partially ordered set of RTs closed under the *meet* and *join* operations, whereby all pairs of elements have a unique element that is their meet and a unique one that is their join. This is to say, for a pair of RT elements R_x and R_y, their lower bound is the set of all $R_z \in L$ such that $R_z \sqsubseteq R_x$ and $R_z \sqsubseteq R_y$, and their unique greatest lower bound is their meet. The meet of any two RTs

$$\begin{tikzcd}
 & R_3 = [\,] = \top & \\
R_1 = \begin{bmatrix} a : T_1 \\ b : T_2 \end{bmatrix} \arrow[ur] & & R_2 = \begin{bmatrix} a : T_2 \\ b : T_1 \end{bmatrix} \arrow[ul] \\
 & R_0 = \bot \arrow[ul] \arrow[ur] &
\end{tikzcd}$$

(a)

$$\begin{tikzcd}
 & & & R_8 = [\,] = \top & & & \\
R_4 = \begin{bmatrix} b : T_1 \end{bmatrix} \arrow[urrr] & & R_5 = \begin{bmatrix} a : T_1 \end{bmatrix} \arrow[ur] & & R_6 = \begin{bmatrix} b : T_2 \end{bmatrix} \arrow[ul] & & R_7 = \begin{bmatrix} a : T_2 \end{bmatrix} \arrow[ulll] \\
 & R_1 = \begin{bmatrix} a : T_1 \\ b : T_1 \end{bmatrix} \arrow[ul] \arrow[ur] & & R_2 = \begin{bmatrix} a : T_1 \\ b : T_2 \end{bmatrix} \arrow[ul] \arrow[ur] & & R_3 = \begin{bmatrix} a : T_2 \\ b : T_2 \end{bmatrix} \arrow[ul] \arrow[ur] & \\
 & & & R_0 = \bot \arrow[ull] \arrow[u] \arrow[urr] & & &
\end{tikzcd}$$

(b)

Figure 2: Record Type lattices ordered by the subtype relation, adapted from Eshghi et al. (2013). While (a) happens to be complemented, RT lattices are not in general, as (b) shows.

R_x and R_y in L is the RT resulting from $R_x \mathbin{\dot\wedge} R_y$, and, given (3), is also extensionally equivalent to the meet type $R_x \wedge R_y$. Dually, if the unique least upper bound exists for R_x and R_y this is their join in L and in TTR terms is the result of $R_x \mathbin{\dot\vee} R_y$, but *not* necessarily extensionally equivalent to the join type $R_x \vee R_y$. This is due to the fact that the result of $R_x \mathbin{\dot\vee} R_y$ may be extensionally equivalent to the minimal common supertype of other pairs of RTs in L (and consequently may be the type of different objects or records which are not of type R_x or R_y), so $R_x \mathbin{\dot\vee} R_y$ can be a more general type than the disjunctive type $R_x \vee R_y$. For example in Fig. 2 (b), the join element in the lattice of R_1 and R_3, consistent with the join being the $\dot\vee$ operator, is R_8, the empty record type, as they have no fields in common. However this is not equivalent to the disjunctive join type as the empty record type includes all objects of type R_2 as well, not just those of type R_1 and R_3.

The decision not to include disjunctive and conjunctive types directly on L, only using RTs and operations that yield new RTs, is motivated by limiting the size (and therefore complexity) of the lattice, and also by keeping consistency in the type hierarchy: the limitation of the lattice to types that are only of record type means this is a record type lattice. As just shown, while the extensionally equivalent RTs for meet types are included in L, elements representing join types are not in general. In summary, given the ordering relation $\sqsubseteq$, the join and meet operations under which the lattice is closed are $\dot\vee$ and $\dot\wedge$.[7]

We now introduce other relevant terminology. One element *covers* another if it is a direct successor to it in the subtype hierarchy. L has a greatest element ($\top$) and least element ($\bot$), with the *atoms* being the elements that cover $\bot$; in Figure 2 (b0 if R_0 is viewed as $\bot$, the atoms are R_1, R_2 and R_3. *Join-irreducible* elements are those which cannot be expressed as the join of two other elements– in Figure 2 (a) the only join-irreducible elements are the atoms and $\bot$, however in Figure 2 (b) they consist of the $\bot$, the atoms and R_4 and R_7.

In line with standard lattice theory, given the characterization of the meet and join operations as $\dot\vee$ and $\dot\wedge$, a RT lattice L ordered by the subtype relation obeys the following rules for any three elements x, y and z in L:

$$x \mathbin{\dot\vee} x = x;\ x \mathbin{\dot\wedge} x = x \qquad \text{(L1. Idempotency)}$$

$$x \mathbin{\dot\vee} y = y \mathbin{\dot\vee} x;\ x \mathbin{\dot\wedge} y = y \mathbin{\dot\wedge} x \qquad \text{(L2. Commutativity)}$$

$$x \mathbin{\dot\vee} (y \mathbin{\dot\vee} z) = (x \mathbin{\dot\vee} y) \mathbin{\dot\vee} z;\ x \mathbin{\dot\wedge} (y \mathbin{\dot\wedge} z) = (x \mathbin{\dot\wedge} y) \mathbin{\dot\wedge} z \qquad \text{(L3. Associativity)}$$

$$x \mathbin{\dot\vee} (x \mathbin{\dot\wedge} y) = x \mathbin{\dot\wedge} (x \mathbin{\dot\vee} y) = x \qquad \text{(L4. Absorption)}$$

Assuming RT lattices are bounded they satisfy the following identity laws:

$$x \mathbin{\dot\vee} \bot = x \text{ (I1.)}$$

$$x \mathbin{\dot\wedge} \top = x \text{ (I2.)}$$

$$x \mathbin{\dot\wedge} \bot = \bot \text{ (I3.)}$$

$$x \mathbin{\dot\vee} \top = \top \text{ (I4.)}$$

RT lattices ordered by the subtype relation are *distributive lattices* as they obey the two distributivity relations:

[7]Graphically, the join of two elements can be found by following the connecting edges upward until they first converge on a single RT, e.g. $R_1 \mathbin{\dot\vee} R_2 = R_5$ in Figure 2 (a), and the meet can be found by following the lines downward until they connect to give the result of their merge operation, e.g. $R_5 \mathbin{\dot\wedge} R_6 = R_2$.

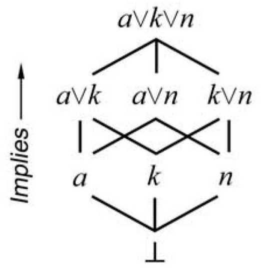

Figure 3: An assertion lattice of propositions A_3 from Knuth (2005)

$$x \mathbin{\dot\wedge} (y \mathbin{\dot\vee} z) = (x \mathbin{\dot\wedge} y) \mathbin{\dot\vee} (x \mathbin{\dot\wedge} z) \qquad \text{(D1. Distributivity of } \dot\wedge \text{ over } \dot\vee\text{)}$$

$$x \mathbin{\dot\vee} (y \mathbin{\dot\wedge} z) = (x \mathbin{\dot\vee} y) \mathbin{\dot\wedge} (x \mathbin{\dot\vee} z) \qquad \text{(D2. Distributivity of } \dot\vee \text{ over } \dot\wedge\text{)}$$

A final piece of lattice terminology is that a RT element R_x has a *complement* if there is a unique element $\neg R_x$ such that $R_x \mathbin{\dot\vee} \neg R_x = \top$ and $R_x \mathbin{\dot\wedge} \neg R_x = \bot$. The lattice Figure 2 (a) is *complemented* as this holds for every element, as R_1 complements R_2 and vice-versa. However RT lattices in general are distributive but not necessarily complemented, as shown in Figure 2 (b), where it can be seen, for example, that R_3 is complemented by R_1, R_4 and R_5.

3.2.1 Adding probability to lattices

To explain the incorporation of probabilities into RT lattices, it is necessary to draw on Knuth (2005)'s work on generalizing a Boolean algebra to the probability calculus through the use of real-valued inclusion measures on lattices. Knuth shows how a Boolean algebra of logical statements can be expressed as a distributed complemented lattice of propositions ordered by the implication ($\rightarrow$) relation, a lattice he calls the *assertion lattice* (see Figure 3). The assertion lattice is isomorphic to the power set of its atomic elements, and so it can also be seen as ordered by the subset inclusion relation $\subseteq$, with its meet being set intersection $\cap$ and its join set union $\cup$ and complement the complement set operator $\sim$. The assertion lattice is distributed and complemented, so the Boolean operators $\wedge$ and $\vee$ and $\neg$ happily coincide with $\cap$, $\cup$ and $\sim$.

Knuth's *Inquiry Calculus* extends Boolean algebra to the probability calculus by characterizing conditional probability $p(x|y)$ as the real-valued *degree* to which statement y implies x in the assertion lattice. This is calculated in terms of the inclusion function $Z(x, y)$ for distributive lattices– that is, the degree to which x includes y:

$$p(x|y) = Z(x, y) = \begin{cases} 1 & \text{if } y \rightarrow x \\ 0 & \text{if } x \wedge y = \bot \\ p & \text{otherwise, where } 0 \leq p \leq 1 \end{cases} \tag{11}$$

If (11) is viewed as a lazy evaluation function, when the first two cases do not apply, the third case p can be calculated by Bayes' theorem, which can also be formulated purely through the inclusion measure between pairs of elements in the lattice by adding $\top$ as a conjunct of the conditional. Knuth's *analog of Bayes' theorem for distributed lattices* is in (12):

$$p(x|y) = p(x|y \wedge \top) = \frac{p(x|\top)p(y|x \wedge \top)}{p(y|\top)} = \frac{Z(x, \top)Z(y, x \wedge \top)}{Z(y, \top)} \tag{12}$$

The standard probability sum and product rules for any two statements are also derivable through a similar technique, using the degree of inclusion of join and meet elements of $\top$ – see Knuth (2005) equations 7-10. When the product rule is applied to the numerator in (12), we get the standard equation for conditional probability, which can again be formulated analogously in terms of the inclusion function:

$$p(x|y) = p(x|y \wedge \top) = \frac{p(x \wedge y|\top)}{p(y|\top)} = \frac{Z(x \wedge y, \top)}{Z(y, \top)} \tag{13}$$

All these calculations are possible through using the degree of inclusion $Z(x, \top)$ initially assigned axiomatically (as a probability prior) to each join-irreducible element x of the assertion lattice (which are atoms in a Boolean lattice)– all other probabilities can be calculated in terms of these using Knuth's lattice-theoretic analogs to the standard probability equations. Knuth shows how these equations hold for any distributed lattice. More detail on this will follow when explaining probability in RT lattices.

3.2.2 Probabilistic RT lattice construction and inference

Having established RT lattices as distributive, we can use Knuth (2005)'s insights to imbue them with probability values for each element. As Knuth did for sets of statements ordered by the relation 'implies', we show how a consistent probability calculus for RT lattices ordered by the relation 'is a subtype of ' falls out naturally from their structure, showing how Cooper et al. (2014, 2015)'s equations for probabilistic TTR shown in Section 3.1 can be derived in terms of a real-valued inclusion function on lattices.

To introduce probabilistic RT lattices, we show one graphically in Figure 4 and use this as a guide for explanation. It shows a well-known probability theoretic example of the possible outcomes of two consecutive coin tosses. The sample space for the possible outcomes here, where H = *heads tossed* and T = *tails tossed*, is {*HH,HT,TH,TT*}. In the spirit of probabilistic TTR, we take each of these outcomes to be a judgement that a situation is of a given record type with a probability value. These four situation types are modelled as the atoms of Figure 4, as their meet types are physically impossible situations stipulated a priori as the least element $\perp$ with probability 0, consistent with assigning prior values to join-irreducible elements (Knuth, 2005, 2006). Each atom is a pair of an RT (shown on the left), and a probability value of the situation s being of that type (on the right of each element).

For each atom R_x we assign a prior probability judgement of a situation being judged of its record type with probability 1, simulating 4 random trials, as we use a uniform distribution for this disjunction of situation types, assuming a fair coin. Following Cooper et al. (2014, 2015), the prior judgement of $s : R_x$ is stored in a set R_x, whose sum of probability judgements we notate $||R_x||$. The prior assignments are in fact probability values when normalized over the sum of the values of all the atomic probability judgements, a set we will call L, which in terms of a probability sample space

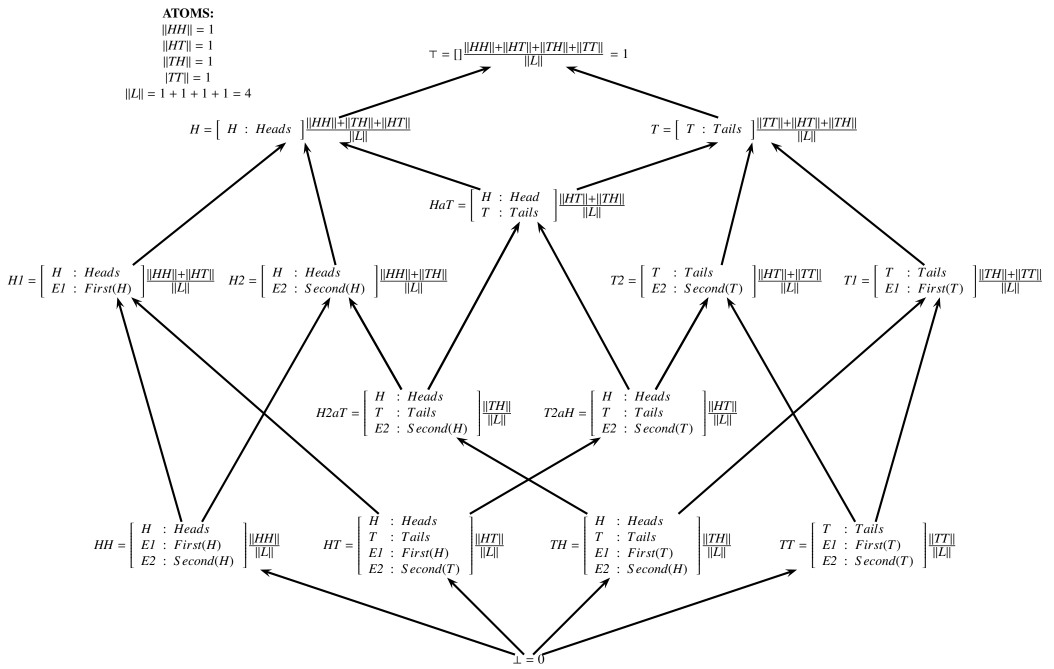
ATOMS:
$\|HH\| = 1$
$\|HT\| = 1$
$\|TH\| = 1$
$\|TT\| = 1$
$\|L\| = 1 + 1 + 1 + 1 = 4$
$\top = []^{\frac{\|HH\|+\|HT\|+\|TH\|+\|TT\|}{\|L\|}} = 1$
$H = \left[\, H : Heads \,\right]^{\frac{\|HH\|+\|TH\|+\|HT\|}{\|L\|}}$
$T = \left[\, T : Tails \,\right]^{\frac{\|TT\|+\|HT\|+\|TH\|}{\|L\|}}$
$HaT = \begin{bmatrix} H : Head \\ T : Tails \end{bmatrix}^{\frac{\|HT\|+\|TH\|}{\|L\|}}$
$H1 = \begin{bmatrix} H : Heads \\ E1 : First(H) \end{bmatrix}^{\frac{\|HH\|+\|HT\|}{\|L\|}}$
$H2 = \begin{bmatrix} H : Heads \\ E2 : Second(H) \end{bmatrix}^{\frac{\|HH\|+\|TH\|}{\|L\|}}$
$T2 = \begin{bmatrix} T : Tails \\ E2 : Second(T) \end{bmatrix}^{\frac{\|HT\|+\|TT\|}{\|L\|}}$
$T1 = \begin{bmatrix} T : Tails \\ E1 : First(T) \end{bmatrix}^{\frac{\|TH\|+\|TT\|}{\|L\|}}$
$H2aT = \begin{bmatrix} H : Heads \\ T : Tails \\ E2 : Second(H) \end{bmatrix}^{\frac{\|TH\|}{\|L\|}}$
$T2aH = \begin{bmatrix} H : Heads \\ T : Tails \\ E2 : Second(T) \end{bmatrix}^{\frac{\|HT\|}{\|L\|}}$
$HH = \begin{bmatrix} H : Heads \\ E1 : First(H) \\ E2 : Second(H) \end{bmatrix}^{\frac{\|HH\|}{\|L\|}}$
$HT = \begin{bmatrix} H : Heads \\ T : Tails \\ E1 : First(H) \\ E2 : Second(T) \end{bmatrix}^{\frac{\|HT\|}{\|L\|}}$
$TH = \begin{bmatrix} H : Heads \\ T : Tails \\ E1 : First(T) \\ E2 : Second(H) \end{bmatrix}^{\frac{\|TH\|}{\|L\|}}$
$TT = \begin{bmatrix} T : Tails \\ E1 : First(T) \\ E2 : Second(T) \end{bmatrix}^{\frac{\|TT\|}{\|L\|}}$
$\perp = 0$

Figure 4: Probabilistic record type lattice L with uniform atomic probabilities for 4 possible outcome situations for two tosses of a coin

is equivalent to the *certain event*. $\|L\|$ normalizes the sum of probability judgements for each record type judgement $\|R_x\|$ to give its prior probability $\frac{\|R_x\|}{\|L\|} = p(s : R_x)$ in line with standard probability assumptions– consequently the real valued judgements initially assigned to the atoms need not in fact sum to unity (Knuth, 2005). In Knuth's terms, these initial assignments are the inclusion values $Z(R_x, \top)$, also equivalent to the unconditional probability $p(s : R_x)$.

The role of TTR here would be trivial if the atoms were simple, non-decomposible type judgements– the only knowledge of the situation available to an agent reasoning with type judgements would be a single probability value attached to each atom– for instance, given the uniform assignment of priors one could calculate the probability of tossing two heads as $p(s : HH) = \frac{1}{4}$. However, in reality, an agent might like to know the probability of other events, such as the event that heads will be tossed at least once, or the probability that the second toss is tails, or a conditional probability of a heads on the second toss given a heads on the first toss. One could use a σ-algebra and generate all possible subsets of outcomes, however this does not capture what the types of interest for an agent might be. For this purpose, the possible outcomes, rather than being atomic, can be structured with relevant type judgements on the situation, for which record types provide a natural representation. For example, the event of two heads tosses HH can be represented through a record type including the heads toss outcome type judgement $[\, H : Heads \,]$ and two fields with $PType$ judgements which are dependent on H, representing the outcome of both the first and second tossing events being of type $Heads$, as in (14).

$$\begin{bmatrix} H & : & Heads \\ E1 & : & First(H) \\ E2 & : & Second(H) \end{bmatrix} \tag{14}$$

The choice of representation for the atoms' RT situation types determines which type judgements can be made and represents the agent's 'take' on the entire situation. The atoms in Figure 4 all have the same amount of structured information for each outcome as (14), which we hope is intuitively relevant for a coin-tossing situation. A model of how an agent decides to a frame a situation is beyond the scope of this chapter.

From the atomic situation types, one can build a RT lattice which includes all possible minimal common supertype judgements of the situation, and the type judgements required to ensure the merge operation characterizes the meet, all with the unconditional probability $p(s : R_x)$ 'stored' at each element R_x. This is achieved through a simple bottom-up graph-based construction procedure which consumes the atoms incrementally, running in time polynomial in the number of atoms, detailed fully in Hough and Purver (fcmg). It terminates when all minimal common supertypes have been generated by the $\curlyvee$ operation defined in simple cases by field intersection (4), and added to the lattice, leaving the maximally common supertype of the whole lattice as an element labelled $\top$ (possibly the empty type [ ]). The unconditional probability for each type judgement is calculated, in the spirit of Knuth's inquiry calculus, purely in terms of the unconditional probability assigned to the atoms– upon generating the minimum common supertype of two elements the algorithm 'stores' the atomic judgements the two elements contain at that new element in a set, so that no judgements are unnecessarily counted twice within the elements, consistent with the inclusion-exclusion principle of lattice joins (Knuth, 2005). Consequently every element's probability is given in terms of the atomic type judgements it contains, normalized by the sum of all atomic judgements $||L||$.

The resulting lattice in our example is as in Figure 4. The labels for each record type such as H for $[\, H : Heads \,]$ are merely for explanation, and in reality the algorithm simply labels the nodes $R_0..R_n$ for a lattice with $n + 1$ elements.[8] We will refer to RT judgements below with respect to these labels. If one checks Figure 4, all the expected prior probabilities given on the right side of each record type make sense– for example, the probability of a heads event, i.e. $p(s : H)$, is $\frac{||HH||+||TH||+||HT||}{||L||} = \frac{3}{4}$, the probability of tossing heads first, $p(s : H1)$, is $\frac{||HH||+||HT||}{||L||} = \frac{1}{2}$, and the probability of tossing tails first, $p(s : T1)$, is also $\frac{1}{2}$, calculated from $\frac{||TH||+||TT||}{||L||}$.

3.2.3 Conditional probability, meets, joins and negative types

A RT lattice L such as Figure 4 can be used as a reasoning system to make inferences in light of partial information becoming available from an ongoing situation– in our case of modelling incremental reference processing this is semantic information from an utterance in progress. We model the principal inference task as predicting the likelihood of relevant type judgements $R_x \in L$ of a situation s, given judgements of the form $s : R_y$ we have so far. To do this we use conditional probability judgements from Knuth's work on distributive lattices described above, but here using the $\sqsubseteq$ relation in place of $\rightarrow$ to give (15).

[8]The labelling convention is hopefully intuitive for discussion purposes here– *H1* is the label for the RT judgement that the first throw is heads, $T2$ the judgement that the second throw is tails, and so on. a is used in the labels to denote 'and', so *HaT* stands for the judgement that there is a heads and tails event in this situation, and so on.

The fact that conditional probability in probabilistic TTR can be formulated in terms of the Z inclusion measure on distributed lattices gives rise to an interesting formulation of type judgements: the likelihood of a situation being of type R_x given it is of type R_y is the degree to which $R_y \sqsubseteq R_x$. We note, as per Knuth's work, that unconditional probabilities are the degree to which the element includes $\top$, so that $Z(R_x, \top) = p(s : R_x)$ and $Z(R_x \curlywedge R_y, \top) = p(s : R_x \curlywedge R_y)$ for all elements.

$$p(s : R_x | s : R_y) = Z(R_x, R_y) = \begin{cases} 1 & \text{if } R_y \sqsubseteq R_x \\ 0 & \text{if } R_x \curlywedge R_y = \bot \\ p & \text{otherwise, where } 0 \le p \le 1 \end{cases} \tag{15}$$

If treated as a lazy evaluation function, in cases where the first two cases do not apply, the third case, the real-valued degree of $R_y \sqsubseteq R_x$, can be calculated using the TTR analog rules of Knuth's inquiry calculus, and also using Cooper et al.'s conditional probability calculation (5) in Section 3.1, replacing the $\wedge$ with $\curlywedge$ to be in line with the meet operation of the RT lattice. This gives (16), which is equivalent to Cooper et al.'s equation due to Remark (10).

$$p(s : R_x | s : R_y) = \frac{p(s : R_x \curlywedge R_y)}{p(s : R_y)} \tag{16}$$

A conditional probability analog for record types can also be formulated as in (17), adapting Knuth's equation for distributive lattices (13), where the Z function again functions as in (15). We show, given $Z(R_x, \top) = p(s : R_x)$ and $Z(R_x \curlywedge R_y, \top) = p(s : R_x \curlywedge R_y)$ for all elements and the equivalence $\curlywedge \equiv \wedge$, Cooper et al.'s equation can be derived from the inclusion measures:

$$\begin{aligned} p(s : R_x | s : R_y) = p(s : R_x | s : R_y \wedge \top) &= \frac{p(s : R_x \wedge R_y | s : \top)}{p(s : R_y | s : \top)} && \text{(Knuth, 2005)} \\ &= \frac{p(s : R_x \curlywedge R_y | s : \top)}{p(s : R_y | s : \top)} \\ &= \frac{Z(R_x \curlywedge R_y, \top)}{Z(R_y, \top)} \\ &= \frac{p(s : R_x \curlywedge R_y)}{p(s : R_y)} \\ &= \frac{p(s : R_x \wedge R_y)}{p(s : R_y)} && \text{(Cooper et al., 2014, 2015)} \end{aligned} \tag{17}$$

If each atom R_x is assigned an initial probability value $Z(R_x, \top)$, Knuth's inclusion measure analog will work. To illustrate, returning to our coin-tossing example, given the first toss is a heads, an agent might like to know the probability of the second toss being heads, i.e. $p(s : \mathit{H2} \,|\, s : \mathit{H1})$. Through Knuth's formulation, we need the numerator $Z(\mathit{H1} \curlywedge \mathit{H2}, \top)$, which can be found on the lattice by $Z(HH, \top)$, which is $\frac{1}{4}$, and the denominator $Z(H1, \top)$, which is $\frac{1}{2}$, giving the expected overall result $p(s : \mathit{H2} \,|\, s : \mathit{H1}) = \frac{1}{2}$. As shown at the bottom of (17), Cooper et al.'s equation is equivalent to Knuth's measure, and for this example $Z(HH, \top) = p(s : HH) = \frac{1}{4}$ and $Z(H1, \top) = p(s : H1) = \frac{1}{2}$.

Similarly, Knuth's product and sum rule analogs will work with this formulation for RT lattices to find the probability of meet and join types. One can derive Cooper et al.'s equation again from Knuth's, given $Z(R_x, \top) = p(s : R_x)$ and the identity law

$R_x \curlywedge \top = R_x$. The below holds for all the meets in the lattice $\curlywedge$ as well as the meet types, due to their equivalence.

$$\begin{aligned}
p(s : R_x \wedge R_y) &= p(s : R_x | s : \top)p(s : R_y | s : R_x \wedge \top) && \text{(Knuth, 2005)}\\
&= p(s : R_x | s : \top)p(s : R_y | s : R_x \curlywedge \top)\\
&= Z(R_x, \top)Z(R_y, R_x \curlywedge \top)\\
&= Z(R_x, \top)Z(R_y, R_x)\\
&= p(s : R_x)p(s : R_y | s : R_x) && \text{(Cooper et al., 2014, 2015)}\\
&= p(s : R_x \curlywedge R_y) && \text{(equivalent element on RT lattice)} \qquad (18)
\end{aligned}$$

In our adaptation of Knuth's formulation, given this is equivalent to $Z(R_x, \top)Z(R_y, R_x)$, it can be said the probability of a situation being of type R_x and of type R_y is the degree to which they have subtypes in common. To illustrate in our running example, an agent may want to know the probability of the situation containing a heads toss and a tails toss, i.e. $p(s : H \wedge T)$. In Knuth's formulation we would need $Z(H, \top)$, which is $\frac{3}{4}$ and $Z(T, H)$, which can be calculated through conditional probability given in (17), giving $\frac{Z(T \curlywedge H, \top)}{Z(H, \top)} = \frac{Z(HaT, \top)}{Z(H, \top)} = \frac{2}{3}$, and so, given $\frac{3}{4} \times \frac{2}{3} = \frac{1}{2}$ we get the expected result. More straightforwardly in practice however, this value could be stored on the lattice as $H \wedge T \equiv H \curlywedge T$, and the probability can be found directly on an element $H \curlywedge T = HaT$, where $p(s : HaT) = \frac{1}{2}$.

The probability of the situation being of the disjunctive join type can be derived in a similar manner from Knuth's inclusion measure of probability, and via the equivalence $\wedge \equiv \curlywedge$. It is possible to derive Cooper et al.'s standard join probability definition, given $Z(R_x, \top) = p(s : R_x)$ and $Z(R_x \curlywedge R_y, \top) = p(s : R_x \curlywedge R_y)$ for all elements:

$$\begin{aligned}
p(s : R_x \vee R_y) &= p(s : R_x | s : \top) + p(s : R_y | s : \top) - p(s : R_x \wedge R_y | s : \top) && \text{(Knuth, 2005)}\\
&= p(s : R_x | s : \top) + p(s : R_y | s : \top) - p(s : R_x \curlywedge R_y | s : \top)\\
&= Z(R_x, \top) + Z(R_y, \top) - Z(R_x \curlywedge R_y, \top)\\
&= p(s : R_x) + p(s : R_y) - p(s : R_x \curlywedge R_y)\\
&= p(s : R_x) + p(s : R_y) - p(s : R_x \wedge R_y) && \text{(Cooper et al., 2014, 2015)} \quad (19)
\end{aligned}$$

In our running example, the agent may want to know the probability that there will be a heads tossed first or a tails tossed second, i.e. $p(s : H1 \vee T2)$. Before deriving this calculation, it is worth noting that there is no element on the lattice which represents an appropriate type judgement of this event. If one were to assume the equivalence of $\curlyvee$ and $\vee$ in TTR, as Figure 4 shows, the result of $H1 \curlyvee T2$ is $\top$, meaning $p(s : H1 \curlyvee T2) = 1$, which is not the probability of $p(s : H1 \vee T2)$, as there is an outcome TH which should not be included in this type judgement. Disjunctive probabilities are available through both Knuth and Cooper et al's equations in terms of the $\curlyvee$ operator. In our example, we can calculate $p(s : H1)$, $p(s : T2)$ and $p(s : H1 \curlywedge T2)$ through simply taking their probability value directly on a pre-computed lattice, or through Knuth's inquiry calculus. These probabilities are $p(s : H1) = \frac{1}{2}$, $p(s : T2) = \frac{1}{2}$ and $p(s : H1 \curlywedge T2) = p(s : HT) = \frac{1}{4}$, which, when plugged into Cooper et al.'s equation in 19 give the expected $\frac{3}{4}$.

It is worth noting that Knuth's inquiry calculus equations still hold for the lattice join $\curlyvee$, only if each element is expressed as the atoms of which it is a join, i.e. $H1 = HH \curlyvee HT$, as the inclusion-exclusion principle for the generalized sum rule calculation requires all the disjuncts as in (20), where we generalize a join operation as $\sqcup$ and meet operation $\sqcap$ on any lattice. If we take $\sqcup$ to be $\dot{\vee}$, then $R_1 ... R_n$ must be atoms for this to give the correct value, however, if $\sqcup$ is $\vee$, then this calculation will work for any lattice element without having to represent its constituent atoms. Given the equivalence $\dot{\wedge} \equiv \wedge$, $\sqcap$ can be either of these in the below.

$$\begin{aligned} p(R_1 \sqcup R_2 \sqcup \cdots \sqcup R_n | \top) = &\sum_{i=1}^{n} p(R_i | \top) \\ &- \sum_{i<j} p(R_i \sqcap R_j | \top) \\ &+ \sum_{i<j<k} p(R_i \sqcap R_j \sqcap R_k | \top) \\ &- \cdots \end{aligned} \tag{20}$$

where if $\sqcup$ is $\dot{\vee}$, $R_1 ... R_n$ must be atoms in the RT lattice.
otherwise if $\sqcup$ is $\vee$, $R_1 ... R_n$ can be any record types.
$\sqcap$ can be $\wedge$ or $\dot{\wedge}$

While the conditional equations above condition on positive type judgements, an agent may want to condition on negative RT judgements, that is to obtain the probability a situation is of a RT in light of evidence that it is not of a given RT. As shown above, an RT lattice is distributive but not guaranteed to be complemented, so we cannot be guaranteed to find a unique complement element on L as was the case for Knuth's Boolean lattices, however we can still calculate $p(s : R_y | s : \neg R_x)$ by obtaining $p(s : R_y)$ in L modulo the probability mass of R_x and that of its subtypes as in (21).

$$p(s : R_y | s : \neg R_x) = \begin{cases} 0 & \text{if } R_y \sqsubseteq R_x \\ \frac{p(s:R_y) - p(s:R_x \dot{\wedge} R_y)}{p(s:\top) - p(s:R_x)} & \text{otherwise} \end{cases} \tag{21}$$

In our running example, an agent may know the first toss is not heads, and given this information wants to calculate the probability that there will be a heads, i.e. $p(s : H | s : \neg H1)$. Through (21) the probability is $\frac{p(s:H) - p(s:H \dot{\wedge} H1)}{p(s:\top) - p(s:H1)} = (\frac{1}{2} - \frac{1}{4}) \div (1 - \frac{1}{2}) = \frac{1}{2}$.

3.2.4 Efficiency gains through graphical search

While all calculations can be done algebraically in terms of the atoms' probabilities, the computational advantage of a pre-constructed finite lattice is that the subtype relation judgements and atomic, meet and join probabilities required for (15) - (21) can be found efficiently through graphical search algorithms by characterising L as a Directed Acyclic Graph (DAG). In Figure 4, the elements can be seen as nodes, and the subtype relation ordering arrows can be viewed as reachability edges which make $\bot$ the source and $\top$ the sink. With this characterisation, if R_y is reachable from R_x then $R_x \sqsubseteq R_y$.

In DAG terms, the meet of two RTs R_x and R_y, $R_x \dot{\wedge} R_y$, can be found at their lowest common ancestor (LCA) node – e.g. $p(s : H1 \dot{\wedge} H2)$ in Figure 4 can be found as $\frac{1}{4}$ directly at node HH. Note if R_y is reachable from R_x, i.e. $R_x \sqsubseteq R_y$, then due to the equivalences listed in (9) and by Remark (10), $p(s : R_x \dot{\wedge} R_y)$ can be found directly at R_x. If the meet of two nodes is $\perp$ (e.g. *HH* and *HT* in Figure 4), then their meet probability is 0 as $p(s : \perp)=0$.

As for the minimal common supertype (join element) of two RTs R_x and R_y, $R_x \curlyvee R_y$, this can be found at their highest common descendent (HCD) node – e.g. $p(s : HH \curlyvee HT)$ in Figure 4 can be found as $\frac{1}{2}$ directly at node *H1*. Note if R_y is reachable from R_x, i.e. if $R_x \sqsubseteq R_y$, then due to the equivalence available for this ordering situation $R_y \curlyvee R_x = R_y$, then $p(s : R_x \curlyvee R_y)$ can be found directly at node R_y. If the join of two nodes is $\top$ (e.g. *H* and *T* in Figure 4), then their minimal common supertype probability is 1 as $p(s : \top)=1$.

Finding the lattice meet or join of two nodes, due to the symmetrical equivalence of finding the LCA node and finding the HCD node (with just a reverse in the reachability relation for the HCD case), is a LCA search problem for a DAG (Aho et al., 1976), for which there are widely developed and efficient algorithms.

3.3 DS-TTR and the DyLan dialogue framework

Moving towards a dialogue application, if we consider the atoms in Fig. 4 to be domain concepts, or possible *information states* (Traum and Larsson, 2003; Ginzburg, 2012), for a dialogue system, it is easy to see graphically how the RT lattice L can be used for incremental inference in terms of a downward search from the initial underspecified $\top$ state. The DyLan framework (Purver et al., 2011) we use here, reasons with RTs incrementally as information states in this way– as incrementally specified RTs become available from the interpretation process they are matched to those in L to determine how far down towards the final states our current state allows us to be. In terms of linguistic processing, different sequences of words or utterances lead to different paths to these atoms, and one can make probability judgements about the likelihood of the final states, or indeed any other states encoded in L as shown above.

To achieve this we need a semantic construction process of record types, which, in line with our motivation of modelling incremental reference processing, should be word-by-word incremental. For this purpose we use TTR combined with the grammar formalism Dynamic Syntax (DS, Kempson et al., 2001; Cann et al., 2005, *inter alia*) in DS-TTR (Purver et al., 2011; Eshghi et al., 2012, 2013) which integrates TTR representations into inherently incremental DS parsing.

While we do not go into detail here, and refer the reader to (Purver et al., 2011; Eshghi et al., 2012, 2013), DS-TTR yields incremental type judgements as words are processed strongly incrementally (left-to-right, word-by-word). We show example DS-TTR record type output for the utterance “the yellow square” in Figure 5. As in this chapter we are concerned with reference processing, we only consider the embedded record type labelled r in these record types, which represents the restrictor of an iota term, representing the proof type of a unique referent of type e (Cann et al., 2005). The utterances we deal with here are definite referring expressions which refer to unique objects in a scene.

3.3.1 Extending DyLan with probability

Given DyLan’s DS-TTR parser provides RTs incrementally, and the context of a situation represented by an RT lattice L is available, it becomes possible to make probabilistic judgements on a word-by-word basis about a set of RTs of interest, such as the possible final states of the dialogue. In line with Knuth (2005), we will call the set of RTs of interest the *central issue*, or I. Here we assume all $R_y \in I$ are disjoint

$$\text{the} \quad \left[\begin{array}{lll} r & : & \left[\begin{array}{lll} x & : & e \end{array}\right] \\ x_{=\iota(r.x,r)} & : & e \end{array}\right]$$

$$\text{yellow} \quad \left[\begin{array}{lll} r & : & \left[\begin{array}{lll} x & : & e \\ col & : & yellow(x) \end{array}\right] \\ x_{=\iota(r.x,r)} & : & e \end{array}\right]$$

$$\text{square} \quad \left[\begin{array}{lll} r & : & \left[\begin{array}{lll} x & : & e \\ col & : & yellow(x) \\ shp & : & square(x) \end{array}\right] \\ x_{=\iota(r.x,r)} & : & e \end{array}\right]$$

Figure 5: Incremental semantic construction by DS-TTR

in L, such as the atoms, ensuring a valid probability distribution. To make inference about the degree to which I is *resolved*, that is, given current evidence R_x whether the agent can predict $p(s : R_y \mid s : R_x) = 1$ for some $R_y \in I$, or the confidence it has in its best prediction and its competitor hypotheses, the interpretation process only need output a conditional probability distribution $P_{R_y \in I}(s : R_y | s : R_x)$. It is straightforward to characterize a standard Maximum Likelihood (ML) multi-class probabilistic classifier for a central issue I and conditioning type judgement $s : R_x$ in these terms, outputting the best 'hard' prediction and its probability (or confidence in its prediction) by the standard *arg max* and *max* functions in (22) and (23), respectively.[9]

$$\hat{R}_y = \underset{R_y \in I}{\arg\max}\, p(s : R_y | s : R_x) \tag{22}$$

$$\hat{p} = p(s : \hat{R}_y | s : R_x) = \max_{R_y \in I} p(s : R_y | s : R_x) \tag{23}$$

All conditional probabilities $p(s : R_y | s : R_x)$ can be found on L using the equations in Section 3.2 and direct look-up on L when used graphically.

Here we assume the conditioning evidence $s : R_x$ comes from a DS-TTR parse. Mapping between DS-TTR's natural language semantics and information states (which may contain non-linguistic context), is not trivial (see Eshghi and Lemon, 2014, for discussion), however here we assume a simple type check which checks the RT yielded by the latest word, R_w, against each RT $R_x \in L$ in a top-down graph search from $\top$, until type matched, such that $R_w \sqsubseteq R_x$ and $R_x \sqsubseteq R_w$.

Given the probabilistic prediction of dialogue states is possible, probabilistic DyLan interpretation can now be defined. Purver et al. (2011) show the standard DyLan interpretation process is DAG-based, whereby parsing consists of adding new edges from the current right frontier vertex of the parse graph and a new vertex, and linking it to a RT concept. A probabilistic extension of this is the function DylanInterpret in

[9]Technically the *arg max* function returns a prediction set which may have multiple elements, if two or more type judgements have the same highest probability value.

(24), taking arguments of the interpretation graph vertex S_i it is interpreting from, the current word being consumed w and the maximal RT compiled so far $maxRT$. DyLan is initialized by setting edge S_i as S_0 (the source of the DAG) and $maxRT$ as the empty record type [] before the first word is consumed. A functional call to the DS-TTR parser $DSTTRparse(S_i, w)$ outputs a tuple $\langle R_w, p\rangle$ of the output R_w and probability p of the parse.[10] We call the conditional probability distribution over RTs in the central issue I the variable *Hypotheses*, which is continuously updated word-by-word.

(24)
```
function DylanInterpret($S_i$, $w$, $maxRT$)
    $\langle R_w, p\rangle = DSTTRparse(S_i, w)$    ▷ Maximal semantics with probability from parsing word.
    $Hypotheses = P_{R_y \in I}(s : R_y | s : R_w \wedge maxRT)$    ▷ Use parse output as conditioning evidence.
    $addEdge(S_i, newVertex(), R_w \wedge maxRT)$    ▷ Add new edge with new RT judgement.
end function
```

3.3.2 Modelling self-repairs

As Hough and Purver (2012); Eshghi et al. (2015) show, interpreting repaired speech is naturally modelled in DyLan through backtracking over interpretation edges in light of an unlikely DS-TTR parse. We can learn or stipulate a real-valued threshold *grammatical* for an acceptable level of grammaticality, and when a parse probability from $DSTTRparse(S_i, w)$ falls below this, backtracking along the DAG is initiated. With the probabilistic interpretation function in (24), it is now also possible to detect irrelevant content arising from interpretation in terms of the maximal probability in the distribution from $P_{R_y \in I}(s : R_y \,|\, s : R_w \wedge maxRT)$ being lower than a real-valued threshold *relevant*.[11] Here we also allow low-relevance judgements to initiate backtracking and then allow the negation of RTs linked to the edges backtracked over as conditioning negative type judgements for the state classifier of the form $P_{R_y \in I}(s : R_y | s : \neg R_w)$. This follows evidence that dialogue agents parse self-repairs efficiently and that repaired dialogue content (reparanda) is given special status but not removed from the discourse context (Hough and Purver, 2012; Ginzburg et al., 2014). This gives a modified definition for DylanInterpret in (25).

(25)
```
function DylanInterpret($S_i$, $w$, $maxRT$)
    $\langle R_w, p\rangle = DSTTRparse(S_i, w)$    ▷ Maximal semantics with probability from parsing word.
    $Hypotheses = P_{R_y \in I}(s : R_y | s : R_w \wedge maxRT)$    ▷ Use parse output as conditioning evidence.
    if $p < grammatical$ or $\max_{R_y \in I} Hypotheses < relevant$ then    ▷ Repair detected?
        $Hypotheses = P_{R_y \in I}(s : R_y | s : \neg R_w)$    ▷ Update hyps based on negative evidence.
        $DylanIntepret(S_{i-1}, w, \neg R_w)$    ▷ Backtrack through recursion.
    $addEdge(S_i, newVertex(), R_w \wedge maxRT)$    ▷ Successful, add new edge with new RT judgement.
end function
```

4 Simulating incremental reference processing

With the RT lattice based classification and prediction and the DyLan dialogue framework at hand it becomes possible to model incremental reference processing consistent with over-specification phenomena and Brennan and Schober (2001)'s experimental results on repaired referring expressions.

We model a simple reference identification task where an instructor produces utterances describing an object which an instructee comprehends and reacts to by selecting

[10] In future work, the probability value p will be used for reasoning within RT lattices.

[11] This version of relevance is simplified here, but Hough and Purver (2014a) and Hough and Purver (fcmg) characterize this information-theoretically.

the object they think best fits the description as quickly as possible. The visual stimulus available to both parties is as in Figure 6.

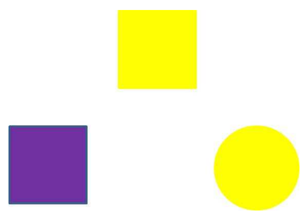

Figure 6: Visual scene for instructor and instructee in the reference identification game

In this game we characterize the referent set of a purple square, yellow square and yellow circle as mutually exclusive referent situation types (record types), which we will label *PSq*, *YSq* and *YC* respectively for convenience, and will characterize as the central issue *referents*. On the interpretation side, the challenge is to predict the final reference situation type judgement $s : R_y$, that the situation s is of record type R_y, given currently available evidence in the form of current type judgement of the situation $s : R_x$. So, as instructions are heard word-by-word the hearer tries to predict the maximally likely referent as in (26), which we formulate as an incremental prediction task for a probabilistic TTR classifier as explained above.

$$\underset{R_y \in referents}{\arg\max}\; p(s : R_y | s : R_x) \tag{26}$$

Initially, upon scanning the scene, the listener entertains a disjunction of possible referent situations as in Figure 7. We assume before the game has begun the atomic situations will all have equal probability ($\frac{1}{3}$), effecting a uniform distribution. Then, bottom-up the lattice L is built, resulting in that in Figure 8. Again, the labels for RTs are for convenience in the discussion here.

After the construction, when DylanInterpret consumes the words of referring expressions incrementally, the probabilities available word-by-word follow the expected pattern– we show the conditional probability distribution over *referents* generated by the model just described at each word in Figure 9 for three referring expressions. The second row in each table shows the incremental type judgement on L by which the classifier conditions its output, which, as described above are from an incremental DS-TTR parse and the maximal semantics from DylanInterpret.

In Figure 9, we model over-specification in "the yellow circle", which is not optimally brief, as "the circle" would be sufficient to resolve the referent. However this still follows Fernández (2013)'s principle of incremental informativity, as "yellow" is more relevant than "the" as it reduces the entropy of the central issue. "The purple square" is also over-specified by the speaker, as "the purple one", or "the purple" would be sufficient for resolution– DyLan resolves the referent upon processing "purple" due to the fact that in this referent situation $p(s : PSQ|s : P) = 1$.

For the test case utterance containing self-repair, "the yell-, uh, purple square" we show a similar probability table in Figure 11. We model Brennan and Schober (2001)'s finding of disfluent spoken instructions speeding up object recognition, with reference to the stages in Figure 10. We describe the stages T0-T5 in terms of the judgements made by the DylanInterpret function as each word is consumed.

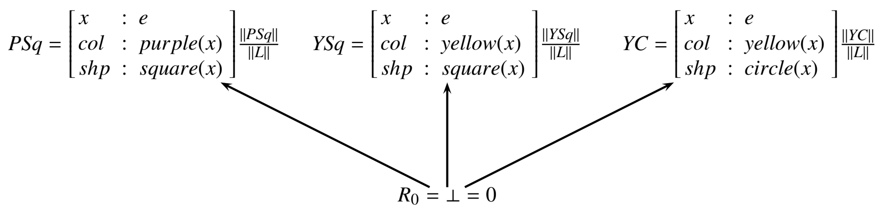
$PSq = \begin{bmatrix} x & : & e \\ col & : & purple(x) \\ shp & : & square(x) \end{bmatrix}^{\frac{\|PSq\|}{\|L\|}}$
$YSq = \begin{bmatrix} x & : & e \\ col & : & yellow(x) \\ shp & : & square(x) \end{bmatrix}^{\frac{\|YSq\|}{\|L\|}}$
$YC = \begin{bmatrix} x & : & e \\ col & : & yellow(x) \\ shp & : & circle(x) \end{bmatrix}^{\frac{\|YC\|}{\|L\|}}$
$R_0 = \perp = 0$

Figure 7: The disjunction of three types of reference situation encoded as record types

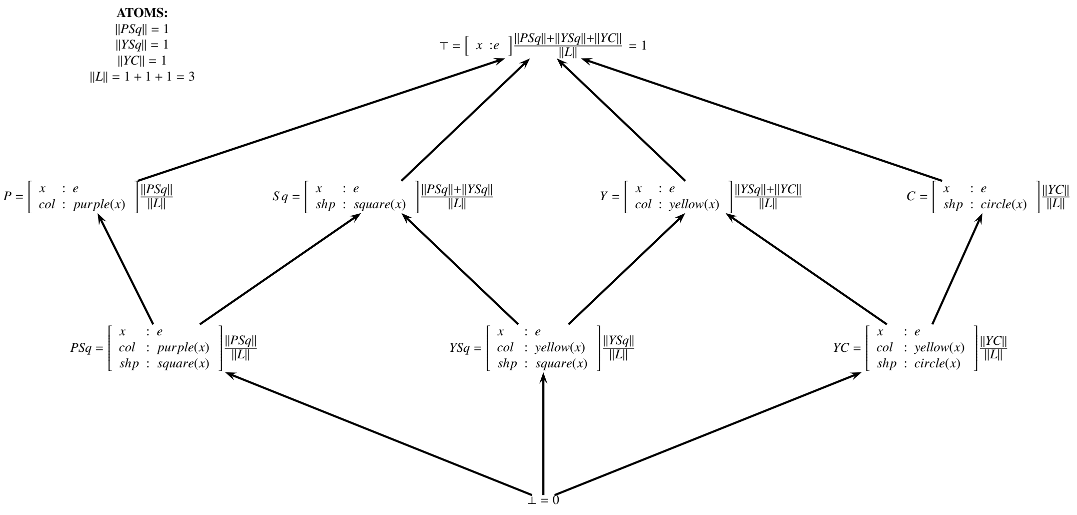
ATOMS:
$\|PSq\| = 1$
$\|YSq\| = 1$
$\|YC\| = 1$
$\|L\| = 1 + 1 + 1 = 3$
$\top = \begin{bmatrix} x : e \end{bmatrix}^{\frac{\|PSq\|+\|YSq\|+\|YC\|}{\|L\|}} = 1$
$P = \begin{bmatrix} x & : & e \\ col & : & purple(x) \end{bmatrix}^{\frac{\|PSq\|}{\|L\|}}$
$Sq = \begin{bmatrix} x & : & e \\ shp & : & square(x) \end{bmatrix}^{\frac{\|PSq\|+\|YSq\|}{\|L\|}}$
$Y = \begin{bmatrix} x & : & e \\ col & : & yellow(x) \end{bmatrix}^{\frac{\|YSq\|+\|YC\|}{\|L\|}}$
$C = \begin{bmatrix} x & : & e \\ shp & : & circle(x) \end{bmatrix}^{\frac{\|YC\|}{\|L\|}}$
$PSq = \begin{bmatrix} x & : & e \\ col & : & purple(x) \\ shp & : & square(x) \end{bmatrix}^{\frac{\|PSq\|}{\|L\|}}$
$YSq = \begin{bmatrix} x & : & e \\ col & : & yellow(x) \\ shp & : & square(x) \end{bmatrix}^{\frac{\|YSq\|}{\|L\|}}$
$YC = \begin{bmatrix} x & : & e \\ col & : & yellow(x) \\ shp & : & circle(x) \end{bmatrix}^{\frac{\|YC\|}{\|L\|}}$
$\perp = 0$

Figure 8: Record type lattice L with uniform atomic probabilities for a reference situation.

At **T0: 'the'** (not in Figure 10, but in Figure 11) the interpreter will only output $[\, x : e \,] = \top$, giving a uniform $p(s : x|s : \top) = \frac{1}{3}$ for $x \in \{PSq, YSq, YC\}$, equivalent to the atomic priors. At **T1: 'yell-'**, the best partial word hypothesis is now "yellow";[12] the interpreter therefore outputs a RT which matches the type judgement $s : Y$ (i.e. that the referent is a yellow object). Taking this judgement as the conditioning evidence, the classifier calculates the conditional distribution $p(s : PSq|s : Y) = 0$, $p(s : YSq|s : Y) = 0.5$ and $p(s : YC|s : Y) = 0.5$ (see the schematic probability distribution at stage T1 in Figure 10 for the three objects). The meet element probabilities required for the conditional probabilities can be found graphically as described above.

T2: 'uh' does not add any information to the referent situation, and can be considered a forward-looking disfluency signal (Ginzburg et al., 2014), however at **T3: 'purple'** low probability in the DS-TTR parse causes a self-repair to be recognised, enforcing backtracking on the parse graph which operates as per the definition for Dy-

[12]Although our current system does not have this capability, we assume a speech recognition module which produces word hypotheses from partial words, progress on which has been made in recent years (see Schlangen and Skantze, 2009).

	the	yellow	square
conditioning type judgement s :	$\top$	Y	YSq
$p(s : PSq)$ (purple square)	$\frac{1}{3}$	0	0
$p(s : YSq)$ (yellow square)	$\frac{1}{3}$	$\frac{1}{2}$	1
$p(s : YC)$ (yellow circle)	$\frac{1}{3}$	$\frac{1}{2}$	0

	the	yellow	circle
conditioning type judgement s :	$\top$	Y	YC
$p(s : PSq)$ (purple square)	$\frac{1}{3}$	0	0
$p(s : YSq)$ (yellow square)	$\frac{1}{3}$	$\frac{1}{2}$	0
$p(s : YC)$ (yellow circle)	$\frac{1}{3}$	$\frac{1}{2}$	1

	the	purple	square
conditioning type judgement s :	$\top$	P	PSq
$p(s : PSq)$ (purple square)	$\frac{1}{3}$	1	1
$p(s : YSq)$ (yellow square)	$\frac{1}{3}$	0	0
$p(s : YC)$ (yellow circle)	$\frac{1}{3}$	0	0

Figure 9: Probability distributions for the objects given maximal incremental semantic information

lanInterpret in (25). The detection of a self-repair repairs the edge $s : Y$, so according to DylanInterpret, the type judgement $s : \neg Y$, i.e. that this is not a yellow object, is available as soon as the repair has been recognised (T3), without having to wait until the parsing process has fully integrated the new semantic information (T4). Using the negative conditioning type judgement, using (21), at T3 the classifier now shifts the distribution using (21) to $p(s : PSq|s : \neg Y) = 1$, $p(s : YSq|s : \neg Y) = 0$ and $p(s : YC|s : \neg Y) = 0$ before the judgement $s : P \wedge \neg Y$ is made at **T4**. This early use of the negative type judgement provides a model for increased subsequent processing speed, however it does not stipulate exact timing at which the negative type inference is made in terms of phonetic form– Brennan and Schober (2001)'s results suggest this information becomes available very quickly upon detection of the substitution repair onset. Finally at **T5: 'square'** given $p(s : PSq | s : PSq) = 1$, the distribution remains unchanged. The last word could be taken as an instance of over-specification again here, due to the lack of information gain in the conditional distribution of *referents*, however, we follow Fernández (2013)'s idea that this is a likely completion of the referring expression, due to the fact syntactic completeness is generally preferred.

5 Discussion

We have presented a novel way of using Knuth (2005, 2006)'s work on probabilistic lattices which has some nice predictions for small reference domains. DS-TTR, whilst currently not fully implemented probabilistically, has potential for fully probabilistic

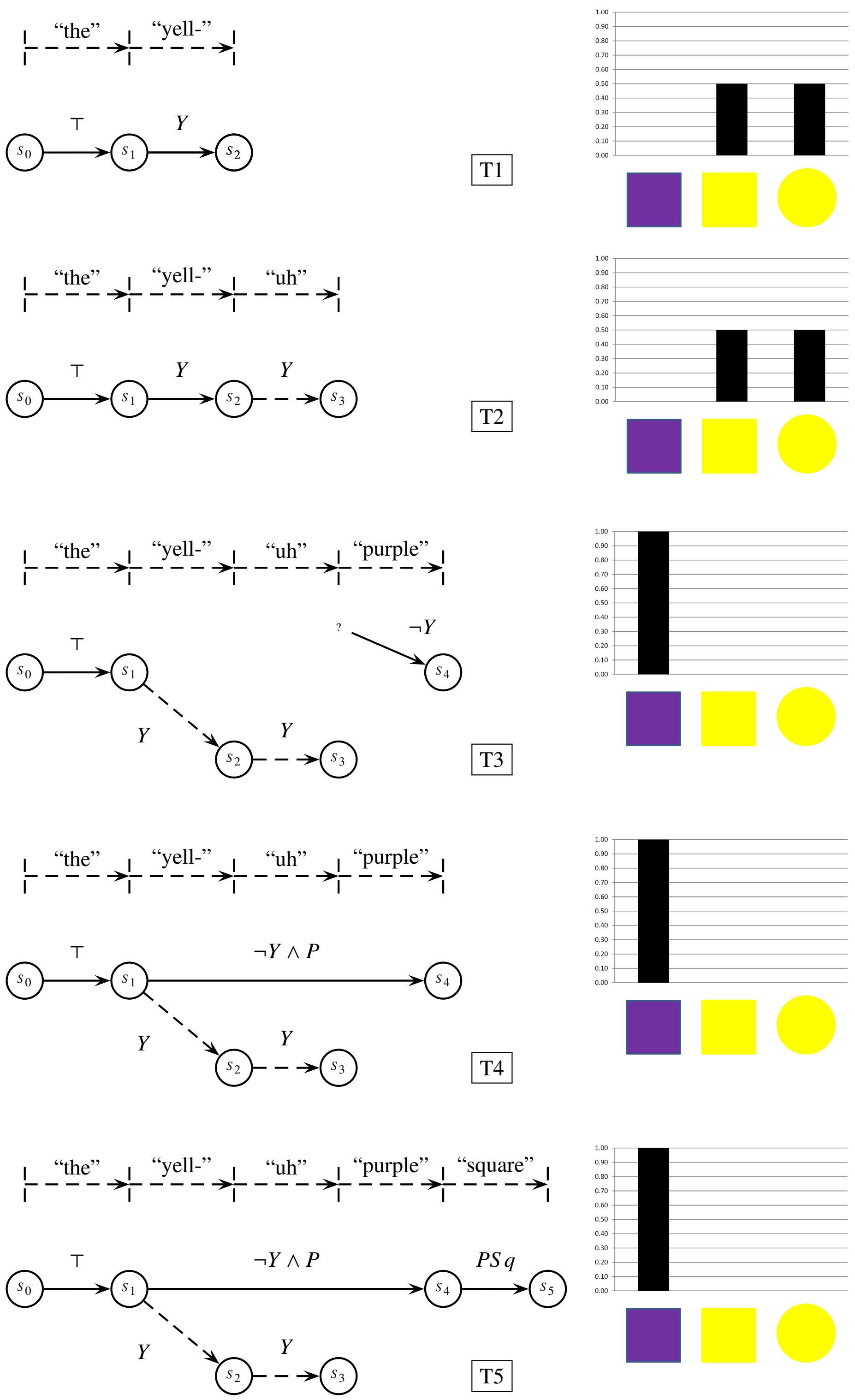

Figure 10: Incremental interpretation of a repaired referring expression in DyLan. The distribution over referents is shown in the bar graphs on the right. The conditioning type judgements label the edges. Repaired edges are dashed.

	the	yell-	uh	purple	square
conditioning type judgement s :	$\top$	Y	Y	$\neg Y, P$	PSq
$p(s : PSq)$ (purple square)	$\frac{1}{3}$	0	0	1,1	1
$p(s : YSq)$ (yellow square)	$\frac{1}{3}$	$\frac{1}{2}$	$\frac{1}{2}$	0,0	0
$p(s : YC)$ (yellow circle)	$\frac{1}{3}$	$\frac{1}{2}$	$\frac{1}{2}$	0,0	0

Figure 11: Probability distributions for the objects given maximal incremental semantic information in a repaired utterance

parsing and generation in practice.[13]

RT lattices show a nice derivation of the standard probability axioms of probabilistic TTR, in line with the characterization of them as a set of types partially ordered by $\sqsubseteq$. This means, given prior assignment of values to the join-irreducible elements, all other probabilities are derivable in terms of the degree to which types include each other. We showed the equivalence of $\curlywedge$ and $\wedge$, noted by Cooper (2012) holds in terms of probability, while the natural join in RT lattices $\curlyvee$ is not equivalent to disjunction $\vee$ in type judgements, due to type lattices generally not being complemented. There are many possible paths for research in probabilistic TTR, but hopefully this lattice characterization is useful for them.

One of the potential draw-backs of the approach is complexity blow-up and scalability. There is exponentiation of the size of the lattices in the size of the disjoint atoms, however not necessarily in their construction time. The other obvious difficulty when scaling to bigger domains is defining the domain of type judgements. However the motivation of TTR is a good one: an agent should only reason with the relevant types to a situation, rather than regarding the whole universe and all the type judgements therein, and using a Questions-Under-Discussion model (Ginzburg, 2012) for relevant issues could help in this regard.

As for reference processing, our model captures over-specification phenomena in REG in terms of probability, but not directly in terms of its decision process the way Dale and Reiter (1995)'s Incremental Algorithm does. However given the cross-linguistic evidence (Rubio-Fernández, 2011) this may not be a weakness– given over-specification may be tied to specific syntactic constructions in specific situations for a given language, it may not be appropriate to model it in the conceptualization stage only, but rather as a side-effect of incremental informativity (Fernández, 2013), which our model captures in its incremental reference resolution. In addition, our framework's processing models how listeners process self-repairs realistically, reasoning about the revocation of a type judgement itself rather than predicting the outcome through positive evidence alone, in line with Brennan and Schober (2001)'s results.

6 Conclusion

We have discussed a dialogue model which incorporates incremental probabilistic inference and efficient methods for constructing probabilistic RT lattices ordered by

[13]See `https://bitbucket.org/dylandialoguesystem/dsttr` for the latest implementation.

the subtype relation, demonstrating their efficacy for realistic reference processing. The model helps explain the experimental results on repaired referring expressions (Brennan and Schober, 2001), and also has a probabilistic characterization of over-specification in terms of incremental relevance. While we model a simple reference domain here, this is intended to be a general interpretation and generation model for dialogue. For this more general purpose, an order-based probabilistic semantics is more suitable than a model conditioning on pre-defined properties of objects as is the tendency for reference resolution and REG algorithms in the literature. We wish to explore the scalability of RT lattices to other domains and their learning capacity in future work.

References

Aho, A. V., J. E. Hopcroft, and J. D. Ullman (1976). On finding lowest common ancestors in trees. *SIAM Journal on computing 5*(1), 115–132.

Brennan, S. and M. Schober (2001). How listeners compensate for disfluencies in spontaneous speech. *Journal of Memory and Language 44*(2), 274–296.

Cann, R., T. Kaplan, and R. Kempson (2005). Data at the grammar-pragmatics interface: the case of resumptive pronouns in English. *Lingua 115*(11), 1475–1665. Special Issue: On the Nature of Linguistic Data.

Chater, N., J. B. Tenenbaum, and A. Yuille (2006). Probabilistic models of cognition: Conceptual foundations. *Trends in cognitive sciences 10*(7), 287–291.

Cooper, R. (2005). Records and record types in semantic theory. *Journal of Logic and Computation 15*(2), 99–112.

Cooper, R. (2012). Type theory and semantics in flux. In R. Kempson, N. Asher, and T. Fernando (Eds.), *Handbook of the Philosophy of Science*, Volume 14: Philosophy of Linguistics, pp. 271–323. North Holland.

Cooper, R. (fcmg). Title TBA. In *Type-Theoretical Semantics: Current Perspectives*. Springer.

Cooper, R., S. Dobnik, S. Lappin, and S. Larsson (2014, April). A probabilistic rich type theory for semantic interpretation. In *Proceedings of the EACL Workshop on Type Theory and Natural Language Semantics (TTNLS)*, Gothenburg, Sweden. Association for Computational Linguistics.

Cooper, R., S. Dobnik, S. Larsson, and S. Lappin (2015). Probabilistic type theory and natural language semantics. *LiLT (Linguistic Issues in Language Technology) 10*.

Dale, R. and E. Reiter (1995). Computational interpretations of the Gricean maxims in the generation of referring expressions. *Cognitive Science 19*(2), 233–263.

Dobnik, S., R. Cooper, and S. Larsson (2013). Modelling language, action, and perception in type theory with records. In *Constraint Solving and Language Processing*, pp. 70–91. Springer.

Eshghi, A., J. Hough, and M. Purver (2013, August). Incremental grammar induction from child-directed dialogue utterances. In *Proceedings of the Fourth Annual Workshop on Cognitive Modeling and Computational Linguistics (CMCL)*, Sofia, Bulgaria, pp. 94–103. Association for Computational Linguistics.

Eshghi, A., J. Hough, M. Purver, R. Kempson, and E. Gregoromichelaki (2012). Conversational interactions: Capturing dialogue dynamics. In S. Larsson and L. Borin (Eds.), *From Quantification to Conversation: Festschrift for Robin Cooper on the occasion of his 65th birthday*, Volume 19 of *Tributes*, pp. 325–349. London: College Publications.

Eshghi, A., C. Howes, E. Gregoromichelaki, J. Hough, and M. Purver (2015, April). Feedback in conversation as incremental semantic update. In *Proceedings of the 11th International Conference on Computational Semantics*, London, UK, pp. 261–271. Association for Computational Linguistics.

Eshghi, A. and O. Lemon (2014). How domain-general can we be? learning incremental dialogue systems without dialogue acts. In *Proceedings of the 18th SemDial Workshop on the Semantics and Pragmatics of Dialogue (DialWatt)*, Herriot Watt University, Edinburgh, pp. 53–61.

Fernández, R. (2006). *Non-Sentential Utterances in Dialogue: Classification, Resolution and Use*. Ph. D. thesis, King's College London, University of London.

Fernández, R. (2013). Rethinking overspecification in terms of incremental processing. In *Proceedings of the PRE-CogSci 2013 Workshop on the Production of Referring Expressions*, Berlin,Germany.

Frank, M. C. and N. D. Goodman (2012). Predicting pragmatic reasoning in language games. *Science 336*(6084), 998–998.

Ginzburg, J. (2012). *The Interactive Stance: Meaning for Conversation*. Oxford University Press.

Ginzburg, J., R. Fernández, and D. Schlangen (2014, June). Disfluencies as intra-utterance dialogue moves. *Semantics and Pragmatics 7*(9), 1–64.

Grice, H. (1975). Logic and Conversation. *Syntax and Semantics 3*(S 41), 58.

Guhe, M. (2007). *Incremental Conceptualization for Language Production*. NJ: Lawrence Erlbaum Associates.

Hough, J., C. Kennington, D. Schlangen, and J. Ginzburg (2015, April). Incremental semantics for dialogue processing: Requirements, and a comparison of two approaches. In *Proceedings of the 11th International Conference on Computational Semantics*, London, UK, pp. 206–216. Association for Computational Linguistics.

Hough, J. and M. Purver (2012, September). Processing self-repairs in an incremental type-theoretic dialogue system. In *Proceedings of the 16th SemDial Workshop on the Semantics and Pragmatics of Dialogue (SeineDial)*, Paris, France, pp. 136–144.

Hough, J. and M. Purver (2014a, September). Lattice theoretic relevance for incremental reference processing. Edinburgh. Poster presentation at REFNET Workshop on Computational and Psychological Models of Reference Comprehension and Production.

Hough, J. and M. Purver (2014b, April). Probabilistic type theory for incremental dialogue processing. In *Proceedings of the EACL 2014 Workshop on Type Theory and Natural Language Semantics (TTNLS)*, Gothenburg, Sweden, pp. 80–88. Association for Computational Linguistics.

Hough, J. and M. Purver (fcmg). A Lattice Theoretic Model of Relevance for Dialogue.

Kempson, R., W. Meyer-Viol, and D. Gabbay (2001). *Dynamic Syntax: The Flow of Language Understanding*. Oxford: Blackwell.

Kennington, C., S. Kousidis, and D. Schlangen (2014, August). Situated incremental natural language understanding using a multimodal, linguistically-driven update model. In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*, Dublin, Ireland, pp. 1803–1812. Dublin City University and Association for Computational Linguistics.

Kennington, C. and D. Schlangen (2014). Situated incremental natural language understanding using markov logic networks. *Computer Speech & Language 28*(1), 240–255.

Knuth, K. H. (2005). Lattice duality: The origin of probability and entropy. *Neurocomputing 67*, 245–274.

Knuth, K. H. (2006). Valuations on lattices and their application to information theory. In *Fuzzy Systems, 2006 IEEE International Conference on*, pp. 217–224. IEEE.

Krahmer, E. and K. Van Deemter (2012). Computational generation of referring expressions: A survey. *Computational Linguistics 38*(1), 173–218.

Larsson, S. (2010). Accommodating innovative meaning in dialogue. pp. 83–90.

Larsson, S. (2011, September). The TTR perceptron: Dynamic perceptual meanings and semantic coordination. In *Proceedings of the 15th Workshop on the Semantics and Pragmatics of Dialogue (SemDial 2011 - Los Angelogue)*, pp. 140–148.

Levelt, W. (1989). *Speaking: From intention to articulation*. MIT Press.

Montague, R. (1974). *Formal Philosophy: Selected Papers of Richard Montague*. Yale University Press.

Peldszus, A., O. Buß, T. Baumann, and D. Schlangen (2012, April). Joint satisfaction of syntactic and pragmatic constraints improves incremental spoken language understanding. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*, Avignon, France, pp. 514–523. Association for Computational Linguistics.

Purver, M., A. Eshghi, and J. Hough (2011, January). Incremental semantic construction in a dialogue system. In J. Bos and S. Pulman (Eds.), *Proceedings of the 9th International Conference on Computational Semantics*, Oxford, UK, pp. 365–369.

Rubio-Fernández, P. (2011). Colours & colores. Invited talk at the 4th Biennial Experimental Pragmatics Conference.

Schlangen, D. and G. Skantze (2009, March). A general, abstract model of incremental dialogue processing. In *Proceedings of the 12th Conference of the European Chapter of the ACL (EACL 2009)*, Athens, Greece, pp. 710–718. Association for Computational Linguistics.

Traum, D. and S. Larsson (2003). The information state approach to dialogue management. In Smith and Kuppevelt (Eds.), *Current and New Directions in Discourse & Dialogue*, pp. 325–353. Kluwer Academic Publishers.

Young, S., M. Gasic, B. Thomson, and J. D. Williams (2013). POMDP-Based Statistical Spoken Dialog Systems: A Review. *Proceedings of the IEEE 101*(5), 1160–1179.