A Minimum Message Length Approach for Argument Interpretation

Ingrid Zukerman and Sarah George
School of Computer Science and Software Engineering
Monash University
Clayton, VICTORIA 3800, AUSTRALIA
email: {ingrid,sarahg}@csse.monash.edu.au

Abstract

We describe a mechanism which receives as input a segmented argument composed of NL sentences, and generates an interpretation. Our mechanism relies on the Minimum Message Length Principle for the selection of an interpretation among candidate options. This enables our mechanism to cope with noisy input in terms of wording, beliefs and argument structure; and reduces its reliance on a particular knowledge representation. The performance of our system was evaluated by distorting automatically generated arguments, and passing them to the system for interpretation. In 75% of the cases, the interpretations produced by the system matched precisely or almost-precisely the representation of the original arguments.

1 Introduction

Discourse interpretation is at the cornerstone of human-computer communication, and an essential component of any dialogue system. In order to produce an interpretation from a user's NL utterances, the concepts referenced by the user's words must be identified, the propositions built using these concepts must be understood, and the relations between these propositions must be determined. Each of these tasks is fraught with uncertainty.

In this paper, we focus on the interpretation of argumentative discourse, which is composed of implications. We present a mechanism for the interpretation of NL arguments which is based on the application of the Minimum Message Length (MML) Principle for the evaluation of candidate interpretations (Wallace and Boulton, 1968). The MML principle provides a uniform and incremental framework for combining the uncertainty arising from different stages of the interpretation process. This enables our mechanism to cope with noisy input in terms of wording, beliefs and argument structure, and to factor out the elements of an interpretation which rely on a particular knowledge representation.

Our interpretation mechanism is embedded in a web-based argumentation system called BIAS (Bayesian Interactive Argumentation System). BIAS uses Bayesian Networks (BNs) (Pearl, 1988) as its knowledge representation and reasoning formalism. It is designed to be a comprehensive argumentation system which will eventually engage in an unrestricted interaction with users. However, the current version of BIAS performs two activities: it generates its own arguments (from a BN) and interprets users' arguments (generating a Bayesian subnet as an interpretation of these arguments). In this paper we focus on the interpretation task.

Figure 1(a) shows a simple argument given by a user, and Figure 1(d) shows a subset of a BN which contains the preferred interpretation of the user's argument; the nodes corresponding to the user's input are shaded. The user's argument is obtained through a web interface (the uncertainty value of the consequent is entered using a drop-down menu). In this example, the user's input differs structurally from the system's interpretation, the belief value for the consequent differs from that in the domain BN, and the wording of the statements differs from the canonical wording of the BN nodes. Still, the system found a reasonable interpretation in the context of its domain model.

The results obtained in this informal trial are validated by our automated evaluation. This evalua-

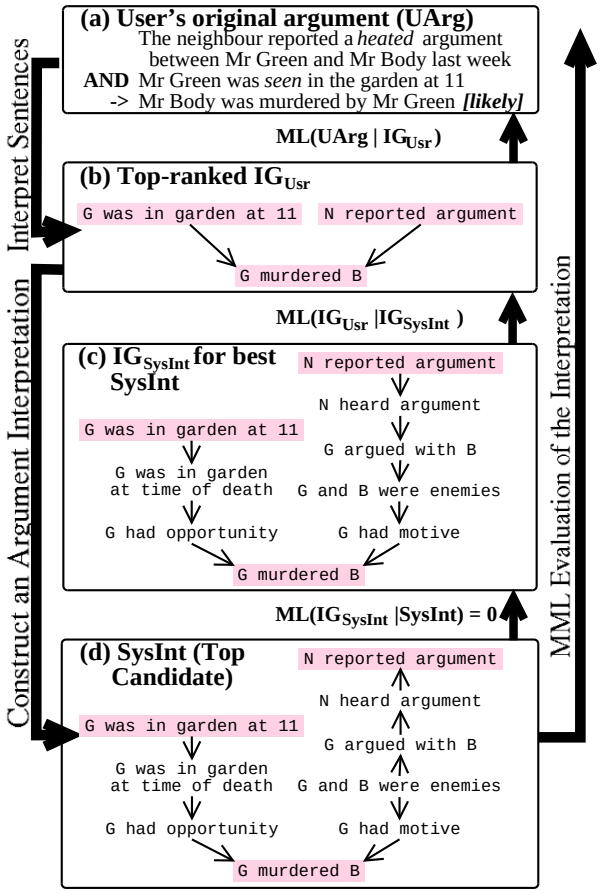

Figure 1: Interpretation and MML evaluation

tion, which assesses baseline performance, consists of passing distorted versions of the system's arguments back to the system for interpretation. In 75% of the cases, the interpretations produced by the system matched the original arguments (in BN form) precisely or almost-precisely.

In the next section, we review related research. We then describe the application of the MML criterion to the evaluation of interpretations. In Section 4, we outline the argument interpretation process. The results of our evaluation are reported in Section 5, followed by concluding remarks.

2 Related Research

Our research integrates plan recognition for discourse understanding with the application of the MML principle (Wallace and Boulton, 1968).

The system described in (Carberry and Lambert, 1999) recognized a user's intentions during expert-consultation dialogues. This system considered several knowledge sources for discourse understanding. It used plan libraries as its main knowledge representation formalism, and handled short conversational turns. In contrast, our system relies on BNs and handles unrestricted arguments.

BNs have been used in several systems that perform plan recognition for discourse understanding, e.g., (Charniak and Goldman, 1993; Horvitz and Paek, 1999; Zukerman, 2001). Charniak and Goldman's system handled complex narratives, using a BN and marker passing for plan recognition. It automatically built and incrementally extended a BN from propositions read in a story, so that the BN represented hypotheses that became plausible as the story unfolded. Marker passing was used to restrict the nodes included in the BN. In contrast, we use domain knowledge to constrain our understanding of the propositions in a user's argument, and apply the MML principle to select a plausible interpretation.

Like Carberry and Lambert's system, both Horvitz and Paek's system and Zukerman's handled short dialogue contributions. Horvitz and Paek used BNs at different levels of an abstraction hierarchy to infer a user's goal in information-seeking interactions with a Bayesian Receptionist. In addition, they used decision-theoretic strategies to guide the progress of the dialogue. We expect to use such strategies when our system engages in a full dialogue with the user. In previous work, Zukerman used a domain model and user model represented as a BN, together with linguistic and attentional information, to infer a user's goal from a short-form rejoinder. However, the combination of these knowledge sources was based on heuristics.

The approach presented in this paper extends our previous work in that (1) it handles input of unrestricted length, (2) it offers a principled technique for selecting between alternative interpretations of a user's discourse, and (3) it handles discrepancies between the user's input and the system's expectations at all levels (wording, beliefs and inferences). Further, this approach makes no assumptions regarding the synchronization between the user's beliefs and the system's beliefs (but it assumes that the system is a domain expert). Finally, this approach may be extended to incorporate various aspects of discourse and dialogue, such as information pertaining to the dialogue history and user modeling information.

The MML principle is a model-selection technique which applies information-theoretic criteria to trade data fit against model complex-

ity (a glossary of model-selection techniques appears in `http://www-white.media.mit.edu/~tpminka/statlearn/glossary`). MML has been used in a variety of applications, e.g., in NL it was used for lexical selection in speech understanding (Thomas et al., 1997). In this paper, we demonstrate its applicability to a higher-level NL task.

3 Argument Interpretation Using MML

The MML criterion implements Occam's Razor, which may be stated as follows: "If you have two theories which both explain the observed facts, then you should use the simplest until more evidence comes along". According to the MML criterion, we imagine sending to a receiver a message that describes a user's NL argument, and we want to send the shortest possible message.[1] This message corresponds to the simplest interpretation of a user's argument. We postulate that this interpretation is likely to be a reasonable interpretation (although not necessarily the intended one).

A message that encodes an NL argument in terms of an interpretation is composed of two parts: (1) instructions for building the interpretation, and (2) instructions for rebuilding the original argument from this interpretation. These two parts balance the need for a concise interpretation (Part 1) with the need for an interpretation that matches closely the user's utterances (Part 2). For instance, the message for a concise interpretation that does not match well the original argument will have a short first part but a long second part. In contrast, a more complex interpretation which better matches the original argument may yield a message that is shorter overall, with a longer first portion, but a shorter second portion. Thus, the message describing the interpretation (BN) which best matches the user's intent will be among the messages with a short length (hopefully the shortest). Further, a message which encodes an NL argument in terms of a reasonable interpretation will be shorter than the message which transmits the words of the argument directly. This is because an interpretation which comprises the nodes and links in a Bayesian subnet (Part 1 of the message) is much more compact than a sequence of words which identifies these nodes and links. If this interpretation is reasonable (i.e., the user's argument is close to this interpretation), then the encoding of the discrepancies between the user's argument and the interpretation (Part 2 of the message) will not significantly increase the length of the message.

In order to find the interpretation with the shortest message length, we compare the message lengths of candidate interpretations. These candidates are obtained as described in Section 4.

3.1 MML Encoding

The MML criterion is derived from Bayes Theorem: $\Pr(D\&H) = \Pr(H) \times \Pr(D|H)$, where D is the data and H is a hypothesis which explains the data.

An optimal code for an event E with probability $\Pr(E)$ has message length $\mathrm{ML}(E) = -\log_2 \Pr(E)$ (measured in bits). Hence, the message length for the data and a hypothesis is:

$$\mathrm{ML}(D\&H) = \mathrm{ML}(H) + \mathrm{ML}(D|H)\,.$$

The hypothesis for which $\mathrm{ML}(D\&H)$ is minimal is considered the best hypothesis.

Now, in our context, *UArg* contains the user's argument, and *SysInt* an interpretation generated by our system. Thus, we are looking for the *SysInt* which yields the shortest message length for

$$\mathrm{ML}(\mathit{UArg}\&\mathit{SysInt}) = \mathrm{ML}(\mathit{SysInt}) + \mathrm{ML}(\mathit{UArg}|\mathit{SysInt})$$

The first part of the message describes the interpretation, and the second part describes how to reconstruct the argument from the interpretation. To calculate the second part, we rely on an intermediate representation called *Implication Graph* (*IG*). An Implication Graph is a graphical representation of an argument, which represents a basic "understanding" of the argument. It is composed of simple implications of the form $\mathit{Antecedent}_1\, \mathit{Antecedent}_2 \ldots \mathit{Antecedent}_n \Rightarrow \mathit{Consequent}$ (where $\Rightarrow$ indicates that the antecedents imply the consequent, without distinguishing between causal and evidential implications). IG_{Usr} represents an understanding of the user's argument. It contains propositions from the underlying representation, but retains the structure of the user's argument. IG_{SysInt} represents an understanding of a candidate interpretation. It is directly obtained from *SysInt*, but it differs from *SysInt* in that all its arcs point towards a goal node and head-to-head evi-

[1] It is worth noting that the sender and the receiver are theoretical constructs of the MML theory, which are internal to the system and are not to be confused with the system and the user. The concept of a receiver which is different from the sender ensures that the message constructed by the sender to represent a user's argument does not make unwarranted assumptions.

dence nodes are represented as antecedents of an implication, while *SysInt* is a general Bayesian subnet. Since both IG_{Usr} and IG_{SysInt} use domain propositions and have the same type of representation, they can be compared with relative ease.

Figure 1 illustrates the interpretation of a short argument presented by a user, and the calculation of the message length of the interpretation. The interpretation process obtains IG_{Usr} from the user's input, and *SysInt* from IG_{Usr} (left-hand side of Figure 1). If a sentence in *UArg* matches more than one domain proposition, the system generates more than one IG_{Usr} from *UArg* (Section 4.1). Each IG_{Usr} may in turn yield more than one *SysInt*. This happens when the underlying representation has several ways of connecting between the nodes in IG_{Usr} (Section 4.2). The message length calculation goes from *SysInt* to *UArg* through the intermediate representations IG_{SysInt} and IG_{Usr} (right-hand side of Figure 1). This calculation takes advantage of the fact that there can be only one IG_{Usr} for each *UArg*–*SysInt* combination. Hence,

$$\begin{aligned}\Pr(\textit{UArg}\,\&\,\textit{SysInt}) &= \Pr(\textit{UArg}, IG_{Usr}, \textit{SysInt}) \\ &= \Pr(\textit{UArg}|IG_{Usr}, \textit{SysInt}) \times \\ &\quad \Pr(IG_{Usr}|\textit{SysInt}) \times \Pr(\textit{SysInt}) \\ &\overset{\text{cond. ind.}}{=} \Pr(\textit{UArg}|IG_{Usr}) \times \\ &\quad \Pr(IG_{Usr}|\textit{SysInt}) \times \Pr(\textit{SysInt})\end{aligned}$$

Thus, the length of the message required to transmit the user's argument and an interpretation is

$$\begin{aligned}\mathrm{ML}(\textit{UArg}\,\&\,\textit{SysInt}) &= \mathrm{ML}(\textit{UArg}|IG_{Usr}) + \\ &\quad \mathrm{ML}(IG_{Usr}|\textit{SysInt}) + \\ &\quad \mathrm{ML}(\textit{SysInt}) \qquad (1)\end{aligned}$$

That is, for each candidate interpretation, we calculate the *length* of the message which conveys:

- *SysInt* – the interpretation,
- $IG_{Usr}|\textit{SysInt}$ – how to obtain the belief and structure of IG_{Usr} from *SysInt*,[2] and
- $\textit{UArg}|IG_{Usr}$ – how to obtain the sentences in *UArg* from the corresponding propositions in IG_{Usr}.

The interpretation which yields *the shortest message* is selected (the message-length equations for each component are summarized in Table 1).

[2]We use IG_{SysInt} for this calculation, rather than *SysInt*. This does not affect the message length because the receiver can obtain IG_{SysInt} directly from *SysInt*.

Throughout the remainder of this section, we describe the calculation of the components of Equation 1, and illustrate this calculation using the simple example in Figure 2 (the message length calculation for our example is summarized in Table 2).

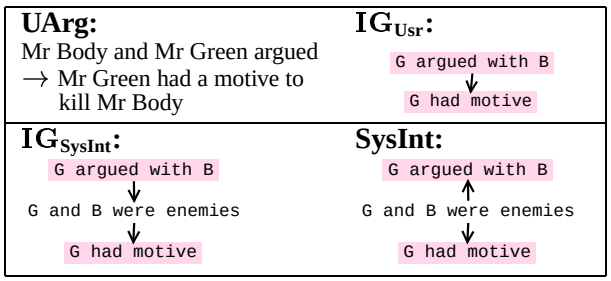

Figure 2: Simple Argument and Interpretation

3.2 Calculating ML(SysInt)

In order to transmit *SysInt*, we simply send its propositions and the relations between them. A standard MML assumption is that the sender and receiver share domain knowledge (recall that the receiver is not the user, but is a construct of the MML theory). Hence, one way to send *SysInt* consists of transmitting how *SysInt* is extracted from the domain representation. This involves selecting its propositions from those in the domain, and then choosing which of the possible relations between these propositions are included in the interpretation. In the case of a BN, the propositions are represented as nodes, and the relations between propositions as arcs. Thus the message length for *SysInt* in the context of a BN is

$$\log_2(\#\_\text{nodes}(\textit{SysInt})) + \log_2(\#\_\text{arcs}(\textit{SysInt})) + \log_2 C^{\#\_\text{nodes}(\textit{domainBN})}_{\#\_\text{nodes}(\textit{SysInt})} + \log_2 C^{\#\_\text{incident\_arcs}(\textit{SysInt})}_{\#\_\text{arcs}(\textit{SysInt})} \qquad (2)$$

For the example in Figure 2, in order to transmit *SysInt* we must choose 3 nodes from the 82 nodes in the BN which represents our murder scenario (the Bayesian subnet in Figure 1(d) is a fragment of this BN). We must then select 2 arcs from the 3 arcs that connect these nodes. This yields a message of length $\log_2 3 + \log_2 2 + \log_2 C^{82}_3 + \log_2 C^3_2 = 1.6 + 1 + 16.4 + 1.6 = 20.6$ bits.

3.3 Calculating ML($\mathbf{IG_{Usr}}$|SysInt)

The message which describes IG_{Usr} in terms of *SysInt* (or rather in terms of IG_{SysInt}) conveys how IG_{Usr} differs from the system's interpretation in two respects: (1) belief, and (2) argument structure.

3.3.1 Belief differences

For each proposition N in both IG_{SysInt} and IG_{Usr}, we transmit any discrepancy between the belief stated by the user and the system's belief in this proposition (propositions that appear in only one IG are handled by the message component which describes structural differences). The length of the message required to convey this information is

$$\sum_{N \in IG_{Usr} \cap IG_{SysInt}} \text{ML}(Bel(N, IG_{Usr}) | Bel(N, IG_{SysInt}))$$

where $Bel(N, IG_x)$ is the belief in proposition N in IG_x. Assuming an optimal message encoding, we obtain

$$\sum_{N \in IG_{Usr} \cap IG_{SysInt}} -\log_2 \Pr(Bel(N, IG_{Usr}) | Bel(N, IG_{SysInt})) \quad (3)$$

which expresses discrepancies in belief as a probability that the user will hold a particular belief in a proposition, given the belief held by the system in this proposition.

Since our system interacts with people, we use linguistic categories of probability that people find acceptable (similar to those used in Elsaesser, 1987) instead of precise probabilities. Our 7 categories are: {VeryUnlikely, Unlikely, ALittleUnlikely, EvenChance, ALittleLikely, Likely, VeryLikely}. This yields the following approximation of Equation 3:

$$\sum_{N \in IG_{Usr} \cap IG_{SysInt}} -\log_2 \Pr(BlCt(N, IG_{Usr}) | BlCt(N, IG_{SysInt})) \quad (4)$$

where $BlCt(N, IG_x)$ is the category for the belief in node N in IG_x.

In the absence of statistical information about discrepancies between user beliefs and system beliefs, we have devised a probability function as follows:

$$\Pr(BlCt(N, IG_{Usr}) | BlCt(N, IG_{SysInt})) = \gamma \times 2^{NumCt-1-|BlCt(N,IG_{Usr}) - BlCt(N,IG_{SysInt})|} \quad (5)$$

where γ is a normalizing constant, and *NumCt* is the number of belief categories (=7). This function yields a maximum probability when the user's belief in node N agrees with the system's belief. This probability gets halved (adding 1 bit to the length of the message) for each increment or decrement in belief category. For instance, if both the user and the system believe that node N is Likely, Equation 5 will yield a probability of $\gamma \times 2^{7-1-0} = 64\gamma$. In contrast, if the user believed that this node has only an EvenChance, then the probability of this belief given the system's belief would be $\gamma \times 2^{7-1-2} = 16\gamma$.

3.3.2 Structural differences

The message which transmits the structural discrepancies between IG_{SysInt} and IG_{Usr} describes the structural operations required to transform IG_{SysInt} into IG_{Usr}. These operations are: node insertions and deletions, and arc insertions and deletions. A node is inserted in IG_{SysInt} when the system cannot reconcile a proposition in the user's argument with any proposition in its domain representation. In this case, the system proposes a special *Escape* (wild card) node. Note that the system does not presume to understand this proposition, but still hopes to achieve some understanding of the argument as a whole. Similarly, an arc is inserted when the user mentions a relationship which does not appear in IG_{SysInt}. An arc (node) is deleted when the corresponding relation (proposition) appears in IG_{SysInt}, but is omitted from IG_{Usr}. When a node is deleted, all the arcs incident upon it are rerouted to connect its antecedents directly to its consequent. This operation, which models a small inferential leap, preserves the structure of the implication around the deleted node. If the arcs so rerouted are inconsistent with IG_{Usr} they will be deleted separately.

For each of these operations, the message announces how many times the operation was performed (e.g., how many nodes were deleted) and then provides sufficient information to enable the message receiver to identify the targets of the operation (e.g., which nodes were deleted). Thus, the length of the message which describes the structural operations required to transform IG_{SysInt} into IG_{Usr} comprises the following components:

$$\begin{aligned}\text{ML}(IG_{Usr} | IG_{SysInt}) = \\ \text{ML}(\text{node insertions}) + \text{ML}(\text{node deletions}) + \\ \text{ML}(\text{arc insertions}) + \text{ML}(\text{arc deletions})\end{aligned} \quad (6)$$

- **Node insertions =** number of inserted nodes plus the penalty for each insertion. Since a node is inserted when no proposition in the domain matches a user's statement, we use an insertion penalty equal to T_M – the probability-like score of the worst acceptable word-match between the user's statement and a proposition (Section 4.1). Thus the message length for node insertions is

$$\log_2(\#\_\text{nodes\_ins}) + \#\_\text{nodes\_ins} \times (-\log_2 T_M) \quad (7)$$

- **Node deletions =** number of deleted nodes plus their designations. To designate the nodes to be deleted, we select them from the nodes in *SysInt* (or IG_{SysInt}):

$$\log_2(\#\_\text{nodes\_del}) + \log_2 C^{\#\_\text{nodes}(IG_{SysInt})}_{\#\_\text{nodes\_del}} \quad (8)$$

- **Arc insertions =** number of inserted arcs plus their designations plus the direction of each arc. (This component also describes the arcs incident upon newly inserted nodes.) To designate an arc, we need a pair of nodes (head and tail). However, some nodes in IG_{SysInt} are already connected by arcs, which must be subtracted from the total number of arcs that can be inserted, yielding

$$\#\_\text{poss\_arc\_ins} = C_2^{\#\_\text{nodes}(IG_{SysInt})+\#\_\text{nodes\_ins}} - \#\_\text{arcs}(IG_{SysInt})$$

We also need to send 1 extra bit per inserted arc to convey its direction. Hence, the length of the message that conveys arc insertions is:

$$\log_2(\#\_\text{arcs\_ins}) + \log_2 C^{\#\_\text{poss\_arc\_ins}}_{\#\_\text{arcs\_ins}} + \#\_\text{arcs\_ins} \quad (9)$$

- **Arc deletions =** number of deleted arcs plus their designations.

$$\log_2(\#\_\text{arcs\_del}) + \log_2 C^{\#\_\text{arcs}(IG_{SysInt})}_{\#\_\text{arcs\_del}} \quad (10)$$

For the example in Figure 2, IG_{SysInt} and IG_{Usr} differ in the node `[B and G were enemies]` and the arcs incident upon it. In order to transmit that this node should be deleted from IG_{SysInt}, we must select it from the 3 nodes comprising IG_{SysInt}. The length of the message that conveys this information is: $\log_2 1 + \log_2 C^3_1 = 1.6$ bits (the automatic rerouting of the arcs incident upon the deleted node yields IG_{Usr} at no additional cost).

3.4 Calculating ML(UArg|IG$_{Usr}$)

The user's argument is structurally equivalent to IG_{Usr}. Hence, in order to transmit *UArg* in terms of IG_{Usr} we only need to transmit how each statement in *UArg* differs from the canonical statement generated for the matching node in IG_{Usr} (Section 4.1). The length of the message which conveys this information is

$$\sum_{N \in IG_{Usr}} \text{ML}(\text{Sentence}_N \text{ in } \textit{UArg} | N)$$

Table 1: Summary of Message Length Calculation

ML(*UArg* & *SysInt*)	Equation 1
ML(*SysInt*)	Equation 2
ML(IG_{Usr}|*SysInt*)	
belief operations	Equations 4, 5
structural operations	Equations 6, 7, 8, 9, 10
ML(*UArg*|IG_{Usr})	Equation 11

Table 2: Summary of Message Length Calculation for the Simple Argument

ML(*SysInt*)	20.6 bits
ML(IG_{Usr}|*SysInt*)	
belief operations (no beliefs stated)	0.0 bits
structural operations	1.6 bits
ML(*UArg*|IG_{Usr})	65.6 bits
ML(*UArg* & *SysInt*)	87.8 bits

where Sentence$_N$ in *UArg* is the user's sentence which matches the proposition for node N in IG_{Usr}. Assuming an optimal message encoding, we obtain

$$\sum_{N \in IG_{Usr}} -\log_2 \Pr(\text{Sentence}_N \text{ in } \textit{UArg} | N) \quad (11)$$

We approximate Pr(Sentence$_N$ in *UArg*$|N$) using the score returned by the comparison function described in Section 4.1. For the example in Figure 2, the discrepancy between the canonical sentences "Mr Body argued with Mr Green" and "Mr Green had a motive to murder Mr Body" and the corresponding user sentences yields a message of length 33.6 bits + 32 bits respectively (=65.6 bits).

4 Interpreting Arguments

Our system generates candidate interpretations for a user's argument by first postulating propositions that match the user's sentences, and then finding different ways to connect these propositions – each variant is a candidate interpretation.

4.1 Postulating propositions

We currently use a naive approach for postulating propositions. For each user sentence S_{Usr} we generate candidate propositions as follows. For each node N in the domain, the system proposes one or more canonical sentences S_N (produced by a simple English generator). This sentence is compared to S_{Usr}, yielding a match-score for the pair (S_{Usr}, N). When a match-score is above a threshold T_M, we

have found a candidate interpretation for S_{Usr}.[3] For example, the proposition [G was in garden at 11] in Figure 1(b) is a plausible interpretation of the input sentence "Mr Green was seen in the garden at 11" in Figure 1(a). Some sentences may have no propositions with match-scores above T_M. This does not automatically invalidate the user's argument, as it may still be possible to interpret the argument as a whole, even if a few sentences are not understood (Section 3.3).

The match-score for a user sentence S_{Usr} and a proposition N – a number in the [0,1] range – is scaled from a weighted sum of individual word-match scores that relate words in S_{Usr} with words in S_N. Inserted or deleted words are given a fixed penalty.

The goodness of a word-match depends on the following factors: (1) level of synonymy – the percentage of synonyms the words have in common (according to WordNet, Miller *et al.*, 1990); (2) position in sentence (expressed as a fraction, e.g., "1/3 of the way through the sentence"); and (3) relation tags – SUBJ/OBJ tags as well as parts-of-speech such as NOUN, VERB, etc (obtained using the MINIPAR parser, Lin 1998). That is, the ith word in sentence S_N, W_{i,S_N}, matches perfectly the jth word in the user's sentence, $W_{j,S_{Usr}}$, if both words are exactly the same, they are in the same sentence position, and they have the same relation tag. The match-score between W_{i,S_N} and $W_{j,S_{Usr}}$ is reduced if their level of synonymy is less than 100%, or if there are discrepancies in their relation tags or their sentence positions. For instance, consider the canonical sentence "Mr Green murdered Mr Body" and the user sentences "Mr Body was murdered by Mr Green" and "Mr Green murdered Ms Scarlet". The first user sentence has a higher score than the second one. This is because the mismatch between the canonical sentence and the first user sentence is merely due to non-content words and word positions, while the mismatch between the canonical sentence and the second user sentence is due to the discrepancy between the objects of the sentences.

[3]This step of the matching process is concerned only with identifying the nodes that best match a user's sentences. Words indicating negation provide further (heuristic-based) information about whether the user intended the positive version of a node (e.g., "Mr Green murdered Mr Body") or the negative version (e.g., "Mr Green *didn't* murder Mr Body"). This information is used when calculating the user's belief in a node.

Upon completion of this process, the match-scores between a user sentence and its candidate propositions are normalized, and the result used to approximate $\Pr(S_{Usr}|N)$, which is required for the MML evaluation (Section 3.4).[4]

At first glance, this process may appear unwieldy, as it compares each of the user's sentences with each proposition in the knowledge base. However, since the complexity of this process is linear for each input sentence, and our informal trials indicate that most user arguments have less than 10 propositions, response time will not be compromised even for large BNs. Specifically, the response time on our 82-node BN is perceived as instantaneous.

4.2 Connecting the propositions

The above process may match more than one node to each of the user's sentences. Hence, we first generate the IG_{Usr}s which are consistent with the user's argument. For instance, the sentence "Mr Green was seen in the garden at 11" in Figure 1(a) matches both [G was in garden at 11] and [N saw G in garden] (but the former has a higher probability). If each of the other input sentences in Figure 1(a) matches only one proposition, two IGs which match the user's input will be generated – one for each of the above alternatives.

Figure 3 illustrates the remainder of the interpretation-generation process with respect to one IG_{Usr}. This process consists of finding connections within the BN between the nodes in IG_{Usr}; eliminating superfluous BN nodes; and generating sub-graphs of the resulting graph, such that all the nodes in IG_{Usr} are connected (Figures 3(b), 3(c) and 3(d), respectively). The connections between the nodes in IG_{Usr} are found by applying a small number of inferences from these nodes (spreading outward in the BN). Currently, we apply two rounds of inferences, as they enable the system to produce "sensible" interpretations for arguments with small inferential leaps. These are arguments whose nodes are separated by at most four nodes in the system's BN, e.g., nodes b and c in Figure 3(d).[5] If upon completion of this process, some nodes are still

[4]We are currently implementing a more principled model for sentence comparison which yields more accurate probabilities.

[5]Intuitively, one round of inferences would miss out on plausible interpretations, while three rounds of inferences would allow too many alternative interpretations. Our choice of two rounds of inferences will be validated during trials with users.

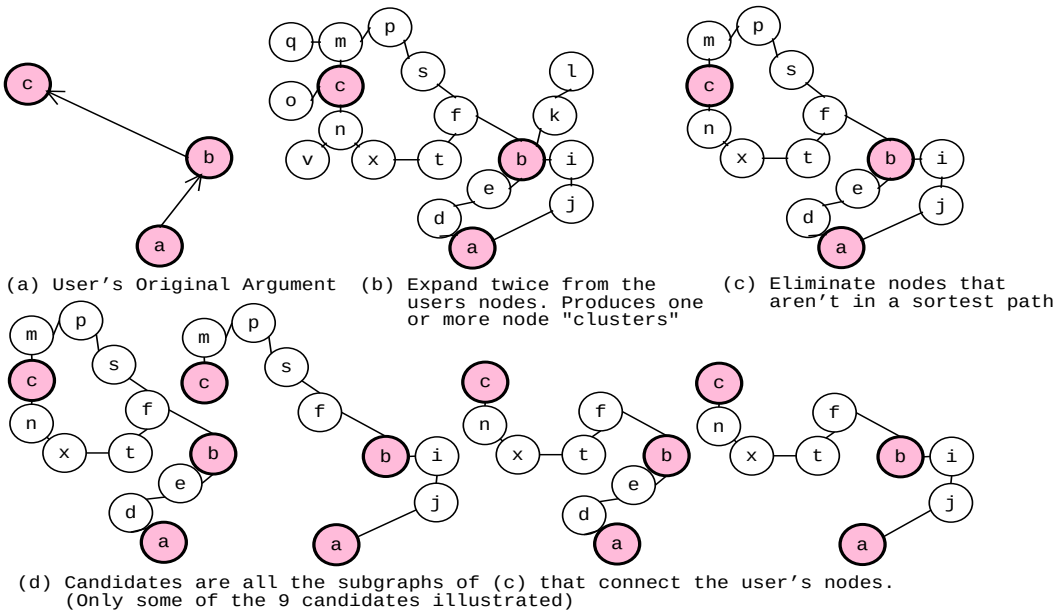

Figure 3: Argument interpretation process

unconnected, the system rejects the current IG_{Usr}. This process is currently implemented in the context of a BN. However, any representation that supports the generation of a connected argument involving a given set of propositions would be appropriate.

5 Evaluation

Our evaluation consisted of an automated experiment where the system interpreted noisy versions of its own arguments. These arguments were generated from different sub-nets of its domain BN, and they were distorted at the BN level and at the NL level. At the BN level, we changed the beliefs in the nodes, and we inserted and deleted nodes and arcs. At the NL level, we distorted the wording of the propositions in the resultant arguments. All these distortions were performed for BNs of different sizes (3, 5, 7 and 9 arcs). Our measure of performance is the edit-distance between the original BN used to generate an argument, and the BN produced as the interpretation of this argument. That is, we counted the number of differences between the source BN and the interpretation. For instance, two BNs that differ by one arc have an edit-distance of 2 (one addition and one deletion), while a perfect match has an edit-distance of 0.

Overall, our results were as follows. Our system produced an interpretation in 86% of the 5400 trials. In 75% of the 5400 cases, the generated interpretations had an edit-distance of 3 or less from the original BN, and in 50% of the cases, the interpretations matched perfectly the original BN. Figure 4 depicts the frequency of edit distances for the different BN sizes under all noise conditions. We plotted edit-distances of 0, ..., 9 and > 9, plus the category NI, which stands for "No Interpretation". As shown in Figure 4, the 0 edit-distance has the highest frequency, and performance deteriorates as BN size increases. Nonetheless, for BNs of 7 arcs or less, the vast majority of the interpretations have an edit distance of 3 or less. Only for BNs of 9 arcs the number of NIs exceeds the number of perfect matches. Figure 5 provides a different view of these results. It displays edit-distance as a percentage of the possible changes for a BN of a particular size (the x-axis is divided into buckets of 10%). For example, if a selected interpretation differs from its source-BN by the insertion of one arc, the percent-edit-distance will be $100 \times \frac{1}{2N+1}$, where N is the number of arcs in the source-BN.[6] The results shown in Figure 5 are consistent with the previous results, with the vast majority of the edits being in the [0,10)% bucket. That is, most of the interpretations are within 10% of their source-BNs.

We also tested each kind of noise separately,

[6] A BN of N arcs has a maximum of N+1 nodes, yielding a maximum of $2N + 1$ edits to create the BN.

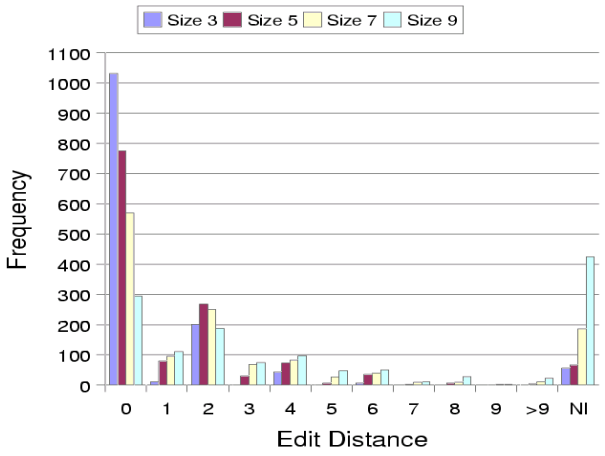

Figure 4: Frequency of edit-distances for all noise conditions (5400 trials)

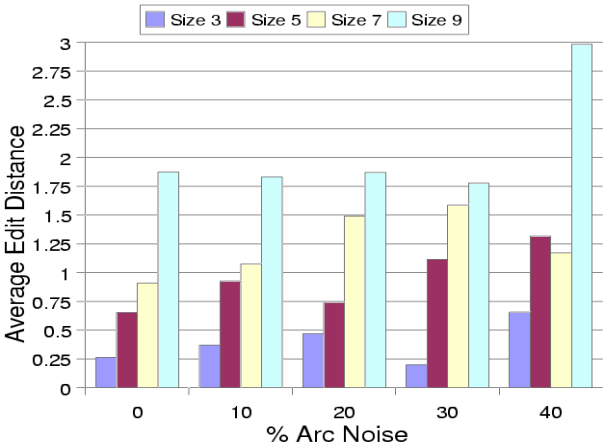

Figure 6: Effect of arc noise on performance (1560 trials)

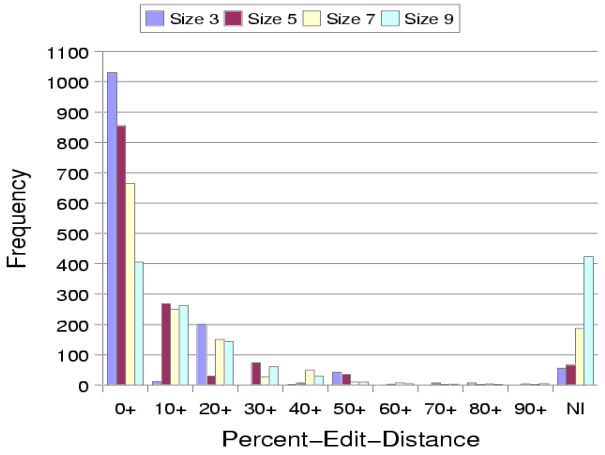

Figure 5: Frequency of edit-distances as percent of maximum edits for all noise conditions (5400 trials)

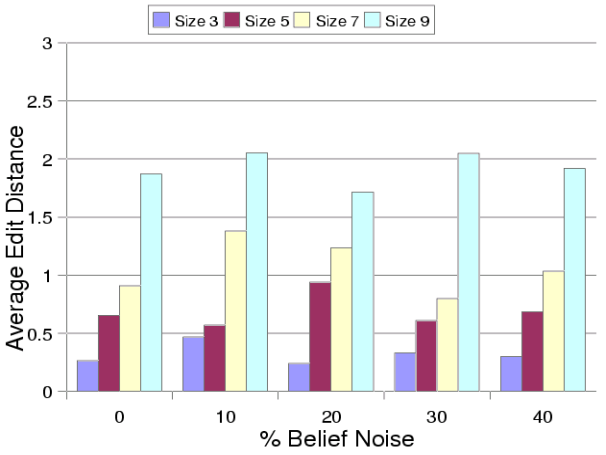

Figure 7: Effect of belief noise on performance (2040 trials)

maintaining the other kinds of noise at 0%. All the distortions were between 0 and 40%. We performed 1560 trials for word noise, arc noise and node insertions, and 2040 trials for belief noise, which warranted additional observations. Figures 6, 7 and 8 show the recognition accuracy of our system (in terms of average edit distance) as a function of arc noise, belief noise and word noise percentages, respectively. The performance for the different BN sizes (in arcs) is also shown. Our system's performance for node insertions is similar to that obtained for belief noise (the graph was not included owing to space limitations). Our results show that the two main factors that affect recognition performance are BN size and word noise, while the average edit distance remains stable for belief and arc noise, as well as for node insertions (the only exception occurs for 40% arc noise and size 9 BNs). Specifically, for arc noise, belief noise and node insertions, the average edit distance was 3 or less for all noise percentages, while for word noise, the average edit distance was higher for several word-noise and BN-size combinations. Further, performance deteriorated as the percentage of word noise increased.

The impact of word noise on performance reinforces our intention to implement a more principled sentence comparison procedure (Section 4.1), with the expectation that it will improve this aspect of our system's performance.

6 Conclusion

We have offered a mechanism which produces interpretations of segmented NL arguments. Our application of the MML principle enables our system to handle noisy conditions in terms of wording, beliefs and argument structure, and allows us to isolate

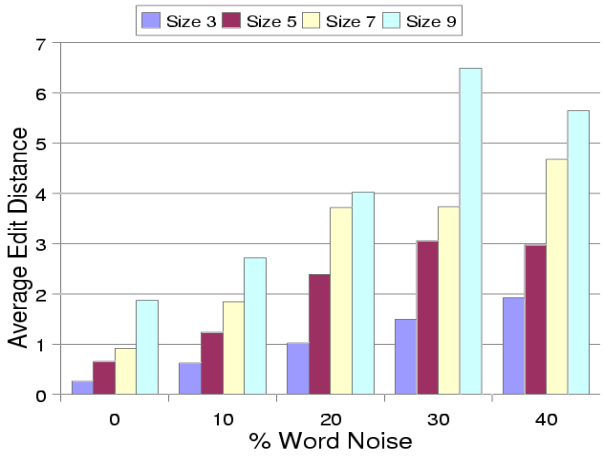

Figure 8: Effect of word noise on performance (1560 trials)

the effect of the underlying knowledge representation on the interpretation process. The results of our automated evaluation were encouraging, with interpretations that match perfectly or almost-perfectly the source-BN being generated in 75% of the cases under all noise conditions.

Our system has the following limitations:

- The interpretations generated by our system are in terms of the propositions and relations known by the system. However, the MML Principle itself addresses this limitation (at least partially), as the length of a message is a quantitative measure for determining whether an interpretation is likely to reflect the user's intentions.
- Our mechanism does not infer an implicit goal proposition, nor does it infer discourse relations from free-form discourse. At present, this limitation is circumvented by forcing the user to state the goal proposition of the argument, and to indicate clearly the antecedents and consequents of the implications in his/her argument (this is done by means of a web-based interface).
- Our argument-interpretation mechanism has been tested on one knowledge representation only – BNs.
- It is unclear whether arguments produced by automatically distorting our system's arguments are representative of arguments generated by people. Further trials with real users will be conducted to ascertain this fact.
- The system's performance deteriorates for large BNs (9 nodes). However, it is unclear whether this will affect the use of the system in practice.

Despite these limitations, we are hopeful about the potential of this approach to address the discourse interpretation challenge.

Acknowledgments

This research was supported in part by Australian Research Council grant A49927212.

References

Sandra Carberry and Lynn Lambert. 1999. A process model for recognizing communicative acts and modeling negotiation subdialogues. *Computational Linguistics*, 25(1):1–53.

Eugene Charniak and Robert P. Goldman. 1993. A Bayesian model of plan recognition. *Artificial Intelligence*, 64(1):50–56.

Christopher Elsaesser. 1987. Explanation of probabilistic inference for decision support systems. In *Proceedings of the AAAI-87 Workshop on Uncertainty in Artificial Intelligence*, pages 394–403, Seattle, Washington.

Eric Horvitz and Tim Paek. 1999. A computational architecture for conversation. In *UM99 – Proceedings of the Seventh International Conference on User Modeling*, pages 201–210, Banff, Canada.

Dekang Lin. 1998. Dependency-based evaluation of MINIPAR. In *Workshop on the Evaluation of Parsing Systems*, Granada, Spain.

George Miller, Richard Beckwith, Christiane Fellbaum, Derek Gross, and Katherine Miller. 1990. Introduction to WordNet: An on-line lexical database. *Journal of Lexicography*, 3(4):235–244.

Judea Pearl. 1988. *Probabilistic Reasoning in Intelligent Systems*. Morgan Kaufmann Publishers, San Mateo, California.

Ian Thomas, Ingrid Zukerman, Jonathan Oliver, David W. Albrecht, and Bhavani Raskutti. 1997. Lexical access for speech understanding using Minimum Message Length encoding. In *UAI97 – Proceedings of the Thirteenth Conference on Uncertainty in Artificial Intelligence*, pages 464–471. Morgan Kaufmann.

C.S. Wallace and D.M. Boulton. 1968. An information measure for classification. *The Computer Journal*, 11:185–194.

Ingrid Zukerman. 2001. An integrated approach for generating arguments and rebuttals and understanding rejoinders. In *UM01 – Proceedings of the Eighth International Conference on User Modeling*, pages 84–94, Sonthofen, Germany.