CHAPTER 8

Making grounding decisions

Given a speech recognition hypothesis, a dialogue system has the choice of accepting or rejecting this hypothesis, but can also choose to provide evidence of understanding, such as a clarification request, or display its understanding. In 3.3.2.7, this choice was referred to as the *grounding decision problem*. In the previous chapters, a static model with hand-crafted thresholds was used. In this chapter, we will use a data-driven decision-theoretic model for the grounding decision problem. Based on task analysis of the HIGGINS navigation domain, dialogue cost functions will be derived, which take dialogue efficiency, consequence of task failure and information gain into account. The dialogue data presented in the previous chapter will then be used to estimate parameters for these cost functions, so that the grounding decision may be based on both confidence and dialogue context.

8.1 The grounding decision problem

The approach to grounding decisions used in the previous chapters (and which is used in many other dialogue systems) is to simply accept a speech recognition hypothesis when the confidence score is high, display understanding for middle-high scores, make a clarification request for middle-low scores and reject the hypothesis for low scores. The problem is that the confidence thresholds for these decisions are most often (as in the previous chapters) based on intuition and not on any theoretically sound and empirically based principle.

In 3.1.2 three important factors for making this decision were discussed, which we may summarise as follows:

1. The result of the early error detection: how confident the system is in its understanding.
2. Task consequences: the cost of falsely accepting an hypothesis (i.e., a misunderstanding), as well as the cost of a false rejection (i.e., a non-understanding).
3. The cost of realising the grounding action and possible reactions to it.

In the simplest case, the choice is between accept and reject, and only Factor 1 above (confidence of understanding) is considered, by comparing the confidence score against a static confidence threshold. This threshold may be optimised to minimise the sum of false acceptances and false rejections, as described in 3.3.1.2, assuming that these errors have the same costs associated with them.

In order to take Factor 2 (task-related costs and utility) into account, Bohus & Rudnicky (2001) use a data-driven technique to derive actual costs in data from the CMU Communicator system, which showed that false acceptances were more costly than false rejections.

Another aspect is that the task costs often vary depending on dialogue state. To incorporate this aspect, Bohus & Rudnicky (2005c) present a method where binary logistic regression is used to determine the costs (in terms of task success) of various types of understanding errors involved in the rejection trade-off. Different regressions may then be calculated in different dialogue states, resulting in dynamic thresholds. Surprisingly, for many dialogue states, the optimal threshold was 0 (i.e., accept everything).

However, none of these methods consider other grounding options than accept and reject and Factor 3 above (cost of grounding actions) is not considered. In some machine-learning approaches to early error detection or n-best list reordering, the machine-learner has been trained to not only consider accept and reject, but also grounding acts such as clarification (Gabsdil & Lemon, 2004; Jonson, 2006). The problem here is how to annotate the training material. When should the system ideally make a clarification? If we know that the hypothesis is correct in the training material, the desired action would be to accept, and if it is incorrect, the desired action would be to reject. Gabsdil & Lemon (2004) suggest that the system should reject when the WER falls below 50% and clarify above that threshold. However, no theoretical motivation for this is provided.

8.1.1 A decision-theoretic approach

Paek & Horvitz (2003) present a decision theoretic approach to the grounding decision problem, based on the framework of *decision making under uncertainty*. According to this proposal, the optimal grounding action *GA* should satisfy the Principle of Maximum Expected Utility (MEU), which can be defined as follows: *Choose an action a, so that the expected utility EU(a) is maximised*. When making this decision, the world may be in one of the states $h_1, h_2, h_3 \ldots h_n$, and this state may have an impact on the effect of the action taken. This effect can be de-

scribed by the function *Utility(a,h_i)*, which is the utility for action *a* under state h_i. Thus, for each action *a*, the probability for each possible state and the utility for taking action *a*, given that state, should be summed up:

(51) $$GA = \arg\max_a EU(a) = \arg\max_a \sum_{i=1}^{n} P(h_i) \times Utility(a, h_i)$$

In Paek & Horvitz (2003), the utilities used in the model were estimated directly by the dialogue designer. In this chapter, we will move one step further and show how this may be estimated from data. We will also show how the model may account for both task-related costs and grounding-related costs, thus accounting for all decision factors discussed above. Before presenting the model, we give a brief overview of the research done on data-driven action selection.

8.1.2 Data-driven action selection

As noted in 2.3.3.3, a lot of recent effort has been invested in making action selection in spoken dialogue systems data-driven, and the grounding decision problem is clearly an instance of action selection.

One approach to data-driven action selection is supervised learning, where a dialogue corpus is used to learn a mapping between the current dialogue state and the action to be taken. The main problem with this approach is how to collect the large amount of data that is needed. The data should contain human-computer dialogues that are representative for the system that is to be built. One possibility could perhaps be to use a complete spoken dialogue system for the target domain interacting with users to collect the data, but this is normally not available (otherwise one would not want to build the system). Also, the machine learner would probably just learn the strategies already utilised by the system. Another solution is to use a human operator acting as a dialogue manager in a Wizard-of-Oz setting. However, since the amount of data that is needed typically is very large, this may be costly to perform. Such an approach also rests on the assumption that the Wizard's behaviour is an optimal model for dialogue system behaviour. The approach is perhaps more suitable for learning general policies for very specific choices. Bohus & Rudnicky (2005b) is an example of this.

Another data-driven approach is to model action selection as a Markov Decision Process (MDP). MDP's consist of a state space with transition probabilities and cost assignments. Unlike supervised learning, an MDP chooses actions that maximise a long-term cumulative sum of rewards (such as user satisfaction). Thus, it can be said to perform planning. Such a model is trained by reinforcement learning, so that the long term reward may be propagated to the different decisions that led to the outcome. An obvious problem is that reinforcement learning may need even more data than supervised learning. To solve this problem, Levin et al. (2000) present an approach in which they estimate a *user model* (MDP parameters that quantify the users' behaviour) by training a supervised learner on a smaller amount of dialogue data. Reinforcement learning is then used to estimate optimal policies by interacting with the simu-

lated user. Levin et al. (2000) show how their system may learn policies for collecting information from the user that seem to be intuitively sound.

A shortcoming with MDP's is that the current state is supposed to be known. This is problematic, since a fundamental problem for spoken dialogue systems is to deal with uncertainty. Williams & Young (2007) proposes the use of an even more advanced stochastic model for action selection: a Partially Observable Markov Decision Process (POMDP). The strengths of POMDP models are that they combine the techniques of automated planning with parallel dialogue state hypotheses and the use of confidence scores, into one statistical framework that admits global optimisation. However, as pointed out by Williams & Young (2007), it is computationally challenging to scale POMDP models to real-world problems, and it is yet unclear whether they will be applicable to more complex domains. Also, a more general concern for data-driven methods relying on user models is how representative such models are of real users.

8.2 The proposed model

In this chapter, we will show how the utilities in the decision-theoretic model discussed in 8.1.1 above may be estimated directly from a small amount of collected dialogue data, based on task-analysis and boot-strapping. To do this, the problem will be described as that of minimising costs, and a general cost measure will be defined. If we want to consider costs instead of utilities, the principle of MEU can be transformed into the Principle of Minimum Expected Cost (EC), where cost should be understood as negative utility:

(52) $$GA = \arg\min_a EC(a) = \arg\min_a \sum_{i=1}^{n} P(h_i) \times Cost(a, h_i)$$

Now, this can be applied to the grounding decision problem in the following way: *Choose a grounding action a, so that the sum of all task-related costs and grounding costs is minimised, considering the probability that the recognition hypothesis is correct.* Thus, the world may be in two states (*correct* and *incorrect* recognition), and a probability measure for these states is needed, as well as a cost function for calculating the costs of the different grounding actions, given these states. The problem is expressed in the following equation (where *P(incorrect)* equals 1-*P(correct)*):

(53) $$GA = \arg\min_a \begin{pmatrix} P(correct) \times Cost(a, correct) + \\ P(incorrect) \times Cost(a, incorrect) \end{pmatrix}$$

In this chapter, these cost functions will be defined by analysing the consequences of different grounding actions. A unified cost measure (accounting for both task-related and grounding costs) will be defined, and the cost functions will use parameters that can be estimated from data.

The proposed model rests on some simplifying assumptions, which will be discussed in more detailed later on:

- Only one concept in the recognised utterance will be considered as being correct or incorrect.
- The possibility that an incorrect recognition hypothesis may be a substitution for a similar concept is not considered. Alternative hypotheses are not considered.
- The costs and probabilities are not dependent on the dialogue history. For example, the utility of grounding actions do not change when they are repeated subsequently.

To select the optimal grounding action according to equation (53) above, a probability measure of the state *correct* is needed, as well as a cost function for calculating the costs of the different grounding actions, given these states.

8.2.1 P(correct)

The most obvious candidate for an estimation of *P(correct)* is the speech recognition confidence score. Although this score should generally not be used directly as a measure of probability (as discussed in 3.3.1.1), it should be possible to approximate such a probabilistic score by using a phoneme recogniser, filler models, or deduce it from the word graph, as argued by Wessel et al. (2001).

Another possibility is to deduce a probabilistic score given a specific application and data collected within it. We will here analyse the confidence scores obtained in the data collection presented in the previous chapter. In this collection, an off-the-shelf ASR was used, and we did not have access to the exact workings of the ASR confidence scoring. In HIGGINS, the word confidence scores from the ASR are averaged into concept confidence scores, as described in 6.4.2.4. To analyse the relation between these confidence scores and *P(correct)*, all recognised concepts in the data were divided into ten interval groups, depending on their confidence scores, that is, scores around 0.1, 0.2, 0.3, etc. Figure 8.1 (left) shows the total number of instances in each such interval (black bars), as well as the number of correct instances (grey bars). Figure 8.1 (right) shows the proportions of correct instances in each interval (diamonds), with a second order polynomial trendline (dotted). The trendline fits the data nicely ($R^2 = 0.999$), indicating that the confidence scores actually do reflect the probability of correctness, although not with a one-to-one mapping.

The rest of this chapter will continue on the assumption that *P(correct)* can be calculated. If it cannot be directly estimated in the ASR, it may be deduced by a regression analysis on collected data. It should be noted, however, that most scores are centred on the median in these data, and are thus not contributing with much information.

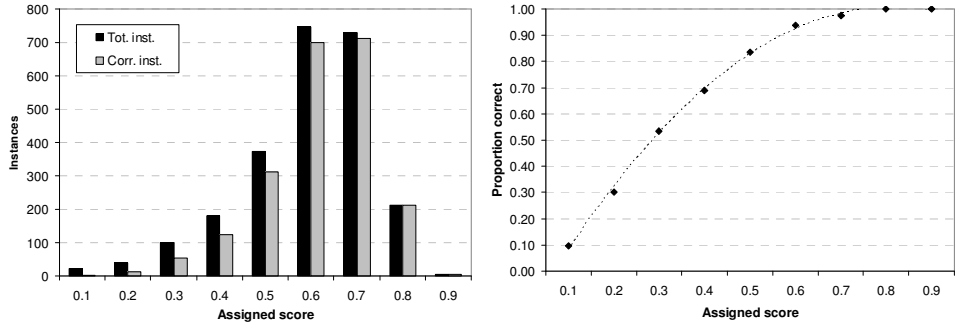

Figure 8.1: The left figure shows the total number of concepts in each confidence interval and the number of concepts that were correctly recognised. The right figure shows the proportion of correct words for each interval, with a second order polynomial trendline.

8.2.2 Cost measure

The model presented in this chapter relies on a unified measure of cost, which may be used for estimating both the task-related costs and the cost of grounding actions. The ultimate measure of cost would be the reduction of user satisfaction. However, user satisfaction is practically only obtainable on the dialogue level, and we need a much more detailed analysis. A cost measure that is relevant for both grounding actions and the task, and that is obtainable on all levels of analysis, is *efficiency*. This is reflected in Clark's principle of least effort (mentioned in 3.1.2.2): "All things being equal, agents try to minimize their effort in doing what they intend to do" (Clark, 1996). Thus, efficiency and user satisfaction should correlate to some degree, at least in a task-oriented dialogue setting as the one used in this chapter. Efficiency may be measured in different ways: by the time spent or the number of utterances, words or syllables used.

To see if these measures had an impact on user satisfaction, the users' estimation of their satisfaction after the dialogues in the collected data were correlated against all these measures of efficiency. As a measure of user satisfaction, the subject's agreement to the statement "I was satisfied with the system" on a scale ranging from 0 to 6 was used. It turned out that all measures of efficiency correlated fairly well with user satisfaction. The one that correlated best was the *total number of syllables* uttered (from both the user and the system). This non-linear regression is shown in Figure 8.2 (logarithmic regression; y = -2.19Ln(x) + 16.54; R2 = 0.622).

It should be noted that correlation with user satisfaction is problematic, partly because it is an ordinal scale. Thus, it is hard to tell whether the non-linear relationship is due to a possible non-linearity of the user satisfaction scale, or to the possibility that user satisfaction reduction decreases as the dialogues get longer. This analysis is only meant to serve as a rough indicator that efficiency is a relevant measure. The correlation is not perfect, and there are of course other factors that are important as well. However, longer dialogues often reflect that a lot of grounding actions (such as clarifications) have been needed, or that misunderstandings have

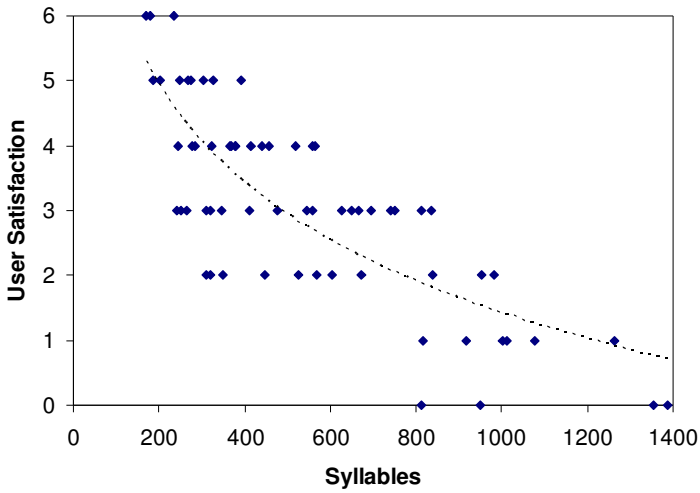

Figure 8.2: Correlation between user satisfaction and total number of syllables per dialogue.

occurred, so that the user has to start all over again. The impact of efficiency on user satisfaction in task-oriented dialogue has also been reported in other studies, such as Bouwman & Hulstijn (1998).

8.2.3 Cost functions

Using efficiency as a cost measure, we will analyse the consequences of different actions, given the correctness of the recognition hypothesis. The actions that will be considered are shown in the following alternative system responses:

(54) U: I can see a red building.
S (ACCEPT): *Ok, can you see a tree in front of you?*
S (DISPLAY): *Ok, a red building, can you see a tree in front of you?*
S (CLARIFY): *A red building?*
S (REJECT): *What did you say? [or just continue]*

We will here analyse the costs (in terms of syllables) for these different grounding actions, given the correctness of the recognition hypothesis. These costs will be based on a set of parameters that are deemed to be important for explaining the costs involved. The parameters are all average estimations over a set of dialogues.

Cost(ACCEPT, correct)

Accepting a correct concept has no cost.

Cost(ACCEPT, incorrect)

Accepting an incorrect concept will lead to a misunderstanding. This error will in many cases somehow slow down the dialogue. Either the user and system have to repair the error, or they might have to start the task all over again. The number of extra syllables the misunderstanding adds to the dialogue will be referred to as *SylMis*.

Cost(REJECT, correct)

If the system rejects a correct concept, the system and user must spend syllables on retrieving a new concept of the same value, either by the system requesting the user to repeat, or by continuing the dialogue and retrieving another concept. The number of syllables it takes to receive new information of the same value as the rejected concept will be referred to as *SylRec.*

Cost(REJECT, incorrect)

Rejecting an incorrect concept has no cost.

Cost(DISPLAY, correct)

Displaying a correct hypothesis will slow down the dialogue by the number of syllables spent on the display utterance and the possible reaction from the user. This will be referred to as *SylDispCor*. Since a concept that is displayed is treated as correct unless the user initiates a repair, it does not matter if the user confirms the display or ignores it.

Cost(DISPLAY, incorrect)

Displaying an incorrect hypothesis will also slow down the dialogue by the number of syllables spent on the display utterance and the possible reaction from the user. This will be referred to as *SylDispInc*. However, since a concept that is displayed is treated as correct unless the user initiates a repair, the user must object to the display. Otherwise, we may say that the grounding has *failed* and a misunderstanding has been introduced (which will prolong the dialogue by *SylMis* number of syllables, as described above). The probability that the user does not correct the system and the grounding fails will be referred to as *P(Fail|Disp,Inc)*. Thus, the expected cost for displaying an incorrect hypothesis is:
SylDispInc + P(Fail|Disp,Inc) x SylMis

Cost(CLARIFY, correct)

Clarifying a correct hypothesis will slow down the dialogue by the number of syllables spent on the clarification request and the possible reaction from the user. This will be referred to as *SylClarCor*. A concept that is clarified is not treated as correct unless the user confirms it. Thus, the clarification of a correct hypothesis will *fail* if the user does not confirm it. The probability

that this happens will be referred to as *P(Fail|Clar,Cor)*. If this happens, the concept is lost and the system and user must spend syllables on retrieving a new concept of the same value (*SylRec*). Thus, the expected cost for clarifying a correct hypothesis is:
SylClarCor + P(Fail|Clar,Cor) x SylRec

Cost(CLARIFY, incorrect)

Clarifying an incorrect hypothesis will slow down the dialogue by the number of syllables spent on the clarification request and the possible reaction from the user. This will be referred to as *SylClarInc*. Since a concept that is clarified is not treated as correct unless the user confirms it, it does not matter if the user disconfirms or ignores the clarification request.

The analysis given above is schematised in Figure 8.3. Together with equation (53), this analysis may then be used to derive cost functions for the different actions, which are shown in Table 8.1.

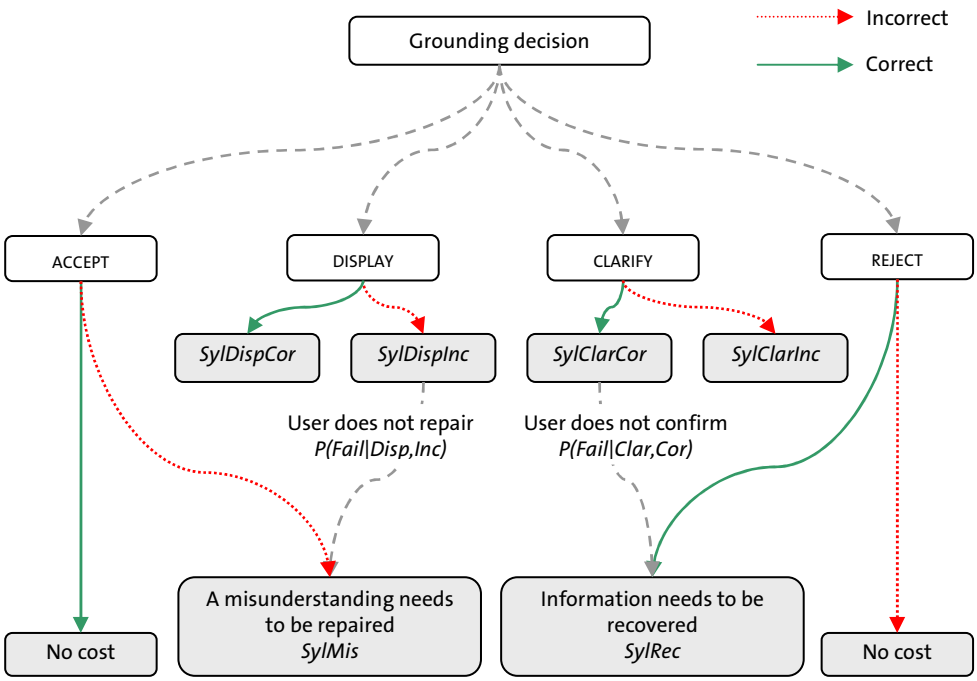

Figure 8.3: Costs involved in taking different grounding actions.

Table 8.1: Cost functions for different grounding actions.

Action	Expected cost
ACCEPT	*P(incorrect)* x *SylMis*
DISPLAY	*P(correct)* x *SylDispCor* + *P(incorrect)* x *(SylDispInc* + *P(Fail|Disp,Inc)* x *SylMis)*
CLARIFY	*P(correct)* x *(SylClarCor* + *P(Fail|Clar,Cor)* x *SylRec)* + *P(incorrect)* x *SylClarInc*
REJECT	*P(correct)* x *SylRec*

8.3 Application to the Higgins navigation domain

The cost functions derived above should be applicable to many dialogue systems, regardless of domain. However, the estimation of the parameters *SylRec* and *SylMis* is highly dependent on the domain. To show how these parameters may be estimated from data, we will make a task analysis specific for the HIGGINS navigation domain used in the previous chapters. In this domain, it is possible to distinguish three different sub-tasks which have different costs associated with them: *positioning the user, establishing the goal*, and *guiding the user*. We will here analyse the first two of these to show how different the task-related costs may be.

8.3.1 Positioning the user

We will start with the positioning task, when the user describes her position, as in the following example:

(55) U: I can see a red building.
S: *Red?*

8.3.1.1 SylRec

The parameter *SylRec* describes the number of syllables it will take to get the same amount of information after a concept has been rejected. This parameter is highly context dependent – it depends on how much information the hypothesised concept provides (its *information gain*), compared to the average concept. This proportion will be referred to as *ConValueH*. The system and the user spent on average 15.0 syllables per important concept[10] accepted by the system. We will refer to this as *SylCon*. Based on these two parameters, *SylRec* can be calculated as follows:

(56) $SylRec = SylCon \times ConValueH$

[10] By important concept, we mean concepts that contribute in the current task. In this example, RED is important, but not BUILDING, since there are buildings everywhere.

How can *ConValueH* be estimated for an individual concept in the positioning phase? The purpose of the positioning phase is to cut down the number of possible user locations. Thus, the value of a concept can be described as the proportion of the set of possible user locations that are cut down after accepting it, compared to the average concept. The proportion of possible locations that are reduced on average after a single concept is accepted can be estimated from data (on average 0.34, which we will refer to as *CutDownA*). The dialogue system can then use the domain database to calculate the proportion of possible locations that would be cut down if the hypothesised concept would be accepted (*CutDownH*). By accepting *ConValueH* number of average concepts, each leaving a proportion of 1 - *CutDownA* possible locations, a proportion of 1 - *CutDownH* locations should be left. This is expressed in the following formula:

$$(1-CutDownA)^{ConValueH} = (1-CutDownH) \tag{57}$$

For example, if half of all possible positions are cut down on average for each concept (*CutDownA* = 0.5), and the hypothesised concept reduces ¾ of the possible positions (*CutDownH* = 0.75), it will take two average concepts to achieve the same effect (*ConValueH* = 2):
$(1 - 0.5)^2 = (1 - 0.75)$.

By combining equations (56) and (57), *SylRec* can be calculated with the following formula:

$$SylRec = SylCon \times \frac{\log(1-CutDownH)}{\log(1-CutDownA)} \tag{58}$$

8.3.1.2 SylMis

We will now turn to the parameter *SylMis,* which describes the number of extra syllables a misunderstanding adds to the dialogue. The risk of accepting an incorrect concept during the positioning phase is that the set of possible user positions may be erroneously constrained. If this happens, the positioning often has to start all over again. Thus, *SylMis* should reflect the number of syllables a complete positioning takes (on average 97.0, which we will refer to as *SylPos*). However, the set of possible user locations does not *need* to be erroneously constrained when accepting an incorrect concept – the user may actually see a red building, even if this was not what she said. The probability that the correct position actually is lost can be described by the parameter *CutDownH* defined above, which describes the proportion of possible locations that is reduced if the hypothesised concept is accepted. Thus, *SylMis* can be calculated as follows:

$$SylMis = SylPos \times CutDownH \tag{59}$$

8.3.1.3 Grounding parameters

The rest of the parameters can be calculated from the data by counting the number of syllables spent on the grounding subdialogues and the number of times they failed. These parameters are shown in Table 8.2. *SylGA* is the number of syllables involved in the grounding act (in the case of DISPLAY or CLARIFY).

Table 8.2: Initial estimation of parameters for example (6).

Parameter	Value
SylClarCor	*SylGA* + 1.4
SylClarInc	*SylGA* + 2.1
SylDispCor	*SylGA* + 0.1
SylDispInc	*SylGA* + 1.2
P(Fail|Clar,Cor)	0.33
P(Fail|Disp,Inc)	0.82

As discussed in the previous chapter, the high value of *P(Fail|Clar,Cor)*, and especially *P(Fail|Disp,Inc)*, might be explained by the fact that the system did not use an elaborate prosodic model for the realisation of fragmentary DISPLAY and CLARIFY acts, that they were sometimes used in inadequate situations, and that the use of such fragments is still very uncommon in dialogue systems.

8.3.1.4 Examples

We will now consider two examples where the concept information gain differs a lot (the concepts under question are underlined):

(60) I can see a mailbox. (*CutDownH* = 0.782; *SylGA* = 2)
(61) I can see a two storey building. (*CutDownH* = 0.118; *SylGA* = 1)

Figure 8.4 shows the difference in number of possible user positions after accepting these two different utterances. Using these parameters, the cost function for the different grounding actions, depending on *P(correct)*, can be calculated to find out which action has the lowest cost for each value of *P(correct)* and thus derive confidence thresholds, as shown in Figure 8.5 and Figure 8.6. In these figures, the costs for the different actions are plotted as functions of *P(correct)*. For each value of *P(correct)*, the action with the lowest cost can be determined. The thresholds at which the optimal action shifts are marked with vertical lines. As the figures show, example (60) has a much higher information gain and thus a wide confidence interval where a clarification request is optimal, whereas example (61) has less information gain and is optimally either accepted or rejected, but never clarified.

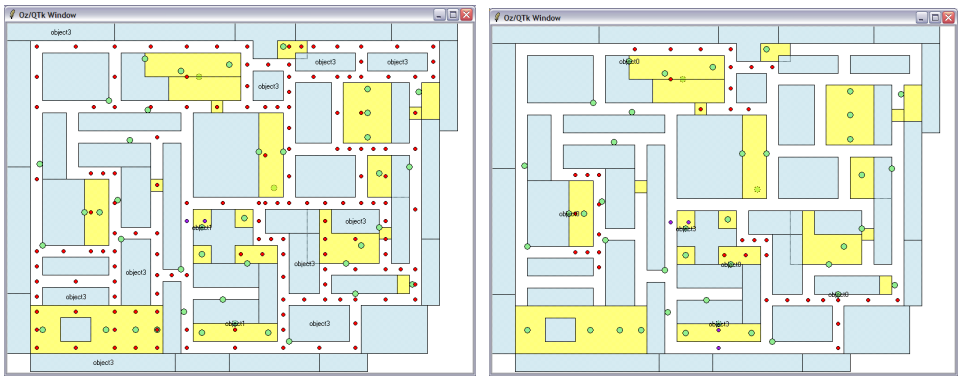

Figure 8.4: Possible user positions (red dots) in the virtual city after accepting the utterance “I can see a two storey building” (left) versus “I can see a mailbox” (right).

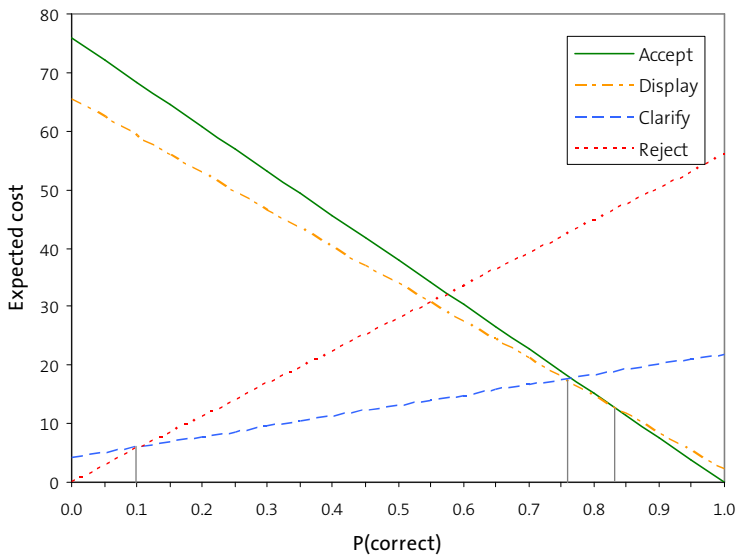

Figure 8.5: Cost functions and confidence thresholds for grounding the concept MAILBOX after “I can see a mailbox”.

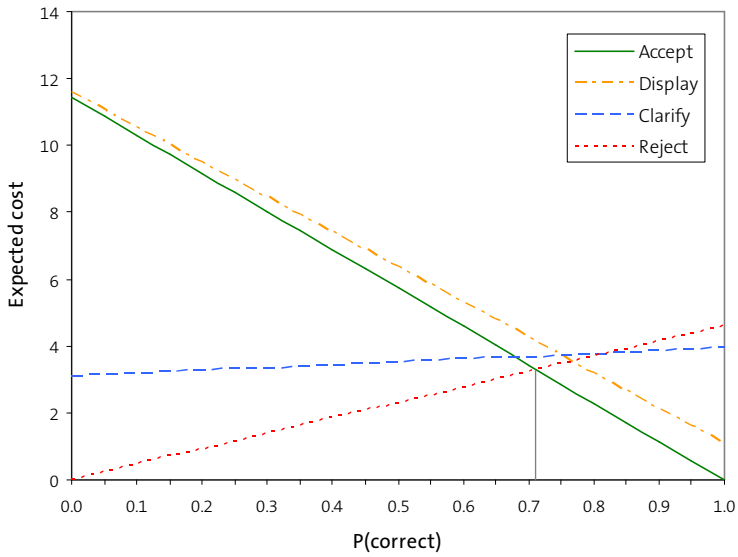

Figure 8.6: Cost functions and confidence thresholds for grounding the concept TWO after "I can see a two storey building".

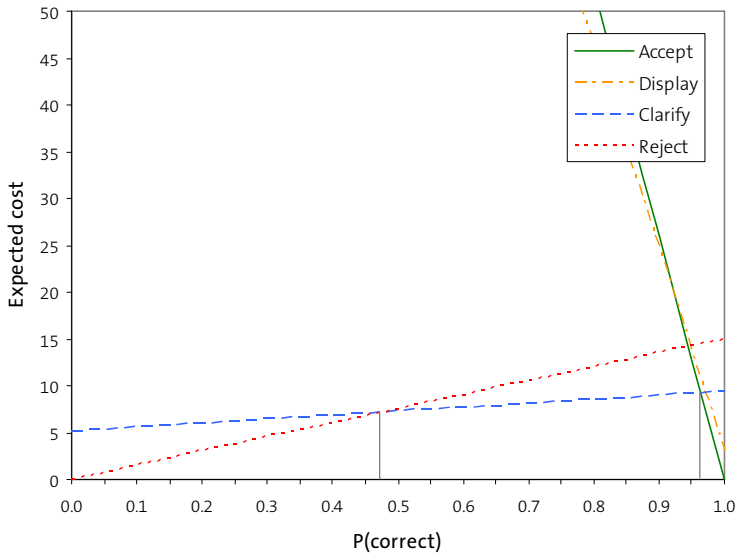

Figure 8.7: Cost functions and confidence thresholds for grounding the concept ATM after "I want to go to an ATM".

The graphs presented above, and the calculation of thresholds, are of course only useful for illustrative purposes. A dialogue system would just calculate the most optimal action, given the value of *P(correct)*. It should be noted that these estimations are based on the data collected with hand-crafted confidence thresholds. If the derived model would be applied to the system, the parameters values would change, thus affecting the parameters in the model. This means that the presented model should be derived iteratively, using bootstrapping, and the parameter values presented here are just the first step in such an iteration. To estimate the parameters, transcription of the dialogues and some annotation is needed. However, given that the logging is adapted for this, we believe that this can be done rather efficiently.

8.3.2 Establishing the goal

In the previous examples, we considered the positioning of the user. However, there is another important task, the establishing of the goal:

(62) U: I want to go to an ATM. (*SylGA*=3)

If this hypothesis would constitute a misunderstanding, it would lead to much higher costs than a misunderstood positioning statement. The misunderstanding might not be identified until the system has actually navigated the user to an ATM, which may take some time. Thus, we can define *SylMis* as the number of syllables it takes on average until the user has reached the (incorrect) goal or restated the goal, which can be estimated to 261.6 from the data. We will assume that *SylRec* is equal to *SylCon* (15.0), and that the other parameters are the same as in the positioning phase. The cost functions and thresholds for grounding "ATM" in the example above are shown in Figure 8.7. Due to the high cost of misunderstandings, a simple accept requires a very high confidence, and goal assertions will therefore most often be clarified.

8.4 Possible extensions

The model presented above may be extended to incorporate other aspects and address some of the simplifying assumptions behind it. We will here briefly discuss such extensions.

8.4.1 Substitutions

In the proposed model, there was no cost associated with rejecting an incorrect concept. This may seem wrong, since the incorrect concept may often be a substitution for a correct concept which is lost and must be recovered. But, if we would add this cost, it should also be added when accepting an incorrect concept, in which case the incorrect concept must be repaired *and* the lost concept must be recovered. If we add the same cost to all actions, the model will not be affected.

However, in the case of DISPLAY or CLARIFY, a substituted concept may be more efficiently recovered, as in the following example:

(63) U: I can see a green building? [RED]
S: *Red?*
U: No, green [GREEN]

Thus, the cost of these grounding actions has been somewhat overestimated. To compensate for this, we must first calculate the probability that an incorrect concept is a substitution for a similar concept, *P(Subst),* and multiply this with *SylRec*. This cost should be added to ACCEPT and REJECT, in the case of an incorrect hypothesis. This cost should also be added to CLARIFY and DISPLAY, if they do not succeed in recovering the substituted concept. The probabilities of this can be described as *P(Fail|Clar,Subst)* and *P(Fail|Disp,Subst),* respectively. Table 8.3 shows the updated costs in case of incorrect hypotheses.

Table 8.3: Costs in case of an incorrect hypothesis that incorporates the possibility of a substitution.

Action	Cost(a, incorrect)
ACCEPT	*SylRep + P(Subst) * SylRec*
REJECT	*P(Subst) * SylRec*
DISPLAY	*SylDispU + SylDispInc + P(Fail|Disp,Inc) * SylRep + P(Fail|Disp,Subst) * P(Subst) * SylRec*
CLARIFY	*SylClarU + SylClarInc + P(Fail|Clar,Subst) * P(Subst) * SylRec*

8.4.2 Concept-level grounding

In the proposed model, only one concept in the hypothesis was considered. The model also accounts for utterance-level grounding, where the whole utterance is considered as being correct or incorrect. However, the model could also be extended to cope with several concepts in an utterance, of which some may be correct and some not, as in the following example (with confidence scores in parenthesis):

(64) U: I can see a red building to the left. [RED (0.7) LEFT (0.2)]

In this case, we should consider 4 possible states instead of 2, as shown in Table 8.4. The combination of grounding actions for each concept may lead to different realisations of grounding moves. Some examples of such actions are shown in Table 8.5.

Table 8.4: Example states when two concepts are considered.

RED	LEFT	P
CORRECT	CORECT	0.14
CORRECT	INCORRECT	0.56
INCORRECT	CORECT	0.06
INCORRECT	INCORRECT	0.24

Table 8.5: Example actions when two concepts are considered.

RED	LEFT	Utterance
CLARIFY	ACCEPT	*Red?*
DISPLAY	CLARIFY	*Do you have the red building on your left?*
CLARIFY	CLARIFY	*A red building on your left?*

8.4.3 Temporal modelling

Another assumption behind the proposed model was that the costs and probabilities were not dependent on the dialogue history, in other words, there is no temporal aspect. However, as Paek & Horvitz (2003) points out, for example repeated requests for repetitions or clarification requests may decrease the utility of such actions.

It would of course be possible to increase the number of parameters and introduce a temporal aspect. However, in the data collected here, there are very few instances of, for example, repeated clarification requests.

Another temporal aspect is that not only the utility, but also the probability of a certain hypothesis should be affected by a history of repeated clarifications. This should ideally be considered in the early error detection, where both the ASR confidence score and the dialogue history could be combined into a more elaborate model of *P(correct)*.

8.4.4 Utterance generation

The model presented in this chapter encourages efficient system utterances by parameters such as *SylClarCor*. However, the model also accounts for the users' recognition of them by parameters such as *P(Fail|Clar,Cor)*. For example, in the data studied here, the system used efficient elliptical clarification requests and display utterances, but this had the negative consequence that they often failed.

The proposed model should therefore be usable for testing the benefits of different utterance realisations. For example, the fragmentary clarification requests used here could be compared with sentential clarification requests (such as "did you say red?"). *SylClarCor* and *SylClarInc* would be higher, but *P(Fail|Clar,Cor)* would perhaps be lower.

8.4.5 User adaptation

Many of the parameters vary considerably between different users; see for example parameter *SylPos* in Figure 8.8. Thus, if the parameters could be tuned for the specific user, the system could adapt its behaviour accordingly. For example, some users may not respond well to clarification requests. This would be reflected in some of the parameters, and the system could avoid making clarifications.

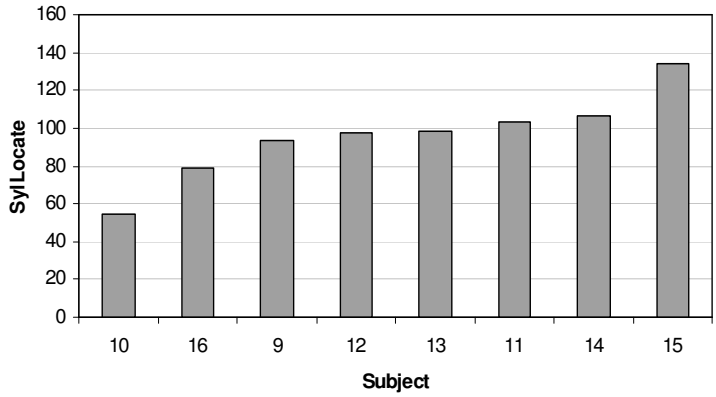

Figure 8.8: How parameter *SylPos* varies between different subjects.

8.5 Discussion

There are still several aspects that are not considered in the proposed model. For example, it is not possible to choose actions that maximise a long-term cumulative sum of rewards (i.e., perform planning). Another limitation is that it only considers one hypothesis from the ASR and cannot hold parallel hypotheses. As discussed in 8.1.2, a more complex model that may account for these factors is Partially Observable Markov Decision Processes (POMDP). The model proposed in this chapter is much simpler and more knowledge-driven (since it is based on task analysis). Thus, it is based on more assumptions and includes more bias, but at the same time it requires less resources and should be easier to apply. As Williams & Young (2007) point out, it is also computationally challenging to scale POMDP models to more complex applications.

Efficiency does not cover all costs involved in dialogue, even in a task-oriented domain such as navigation. For example, the results presented in Chapter 4 indicated that the frustration that the signalling of understanding gives rise to may decrease user satisfaction *per se*, that is, not just by the number of syllables added to the dialogue. It would be interesting to use a more elaborate cost model, for example by applying regression analysis of user satisfaction, as in the PARADISE evaluation framework (Walker et al., 2000a).

The proposed model cannot be directly implemented in the HIGGINS architecture as described in Chapter 6. In this architecture, the grounding action manager (GAM), which only considers the discourse history, is separated from the navigation action manager (NAM), which also looks into the domain database. This distinction was made in order for the grounding actions to be realised as quickly as possible, while the NAM made more complex decisions. However, the model proposed in this chapter relies on the possibility of looking into the domain database for making grounding decisions as well. This shows that it might be unfeasible to maintain the separation of action managers if the system is to take more complex grounding decisions.

As stated above, the cost functions presented in Table 8.1 should be applicable to other domains as well. The two parameters that are task-dependent are *SylRec* and *SylMis*. In this chapter, it was shown how these may be estimated for the HIGGINS navigation domain. For a much simpler domain, such as a standard slot-filling travel booking domain, these parameters could possibly be estimated in a more straightforward manner. In the navigation domain, there is not a fixed set of slots that are to be filled. Thus, each concept may contribute with a different amount of information (the concept information gain). In a domain where a fixed set of slots needs to be filled, this notion is not relevant. Instead, *SylRec* could possibly be mapped directly to *SylCon* (the number of syllables it takes on average to receive a new concept). If there is a final confirmation dialogue at the end of the slot-filling, *SylMis* could possibly be estimated as the number of syllables it takes on average to reach the final confirmation and make the repair.

The presented model also remains to be evaluated, for example by comparing the performance of a system using this model with a system based on handcrafted thresholds, or a more complex model, such as POMDP.

8.6 Summary

This chapter has presented a data-driven decision-theoretic approach to making grounding decisions in spoken dialogue systems, that is, to decide which recognition hypotheses to consider as correct and whether to make a clarification request or display understanding. This model accounts for the uncertainty of the speech recognition hypothesis, as well as the costs involved in taking grounding actions and the task-related costs that a misunderstanding or a rejection would have. Based on a task analysis of the HIGGINS navigation domain, cost functions were derived. It was argued that efficiency– the number of syllables uttered by the user and system – was useful as a cost measure for the navigation domain. Dialogue data was then used to estimate parameters for these cost functions, so that the grounding decision may be based on both confidence and dialogue context. For example, it was shown how concepts with high information gain should more often be clarified than concepts with low information gain, which are either simply rejected or accepted. To silently accept a concept which is associated with a very high cost of misunderstanding, a very high confidence in this concept is required.