Implementation of DomCAT: The Domain Complexity Analysis Tool for Natural Language Dialog Processing

Shannon Duvall
Elon University
Box 2189
Elon, University 27244
336-278-6232

sduvall2@elon.edu

ABSTRACT

While dialog system technology is advancing, there is a lack of theory allowing the vastly different domains for systems to be compared. As a result, all predictions of the cost of building a new dialog system must be made by a dialog expert based on intuition and experience. Recently it has been proposed that entropy can be used as a complexity measure for dialog systems. These calculations would require domain specifications and understanding of information theory. This paper introduces the Domain Complexity Analysis Tool, or DomCAT. With this tool, anyone with basic knowledge of dialog systems can calculate system complexities and create new dialog domain specifications, and dialog complexity calculations can become standard for the field.

Categories and Subject Descriptors

I.2.7 Natural Language Processing E.4 Coding and Information Theory D0. General Software.

General Terms

Management, Measurement, Design.

Keywords

Dialog Processing, Complexity.

1. INTRODUCTION

Dialog systems can differ in many ways, including type of interaction (cooperating on a task, asking for a service, explaining, etc...), length of typical dialogs, and degree of mixed initiative. Having such a wide variety of dialog systems makes them widely applicable but has several drawbacks. It is difficult to compare the performance of systems because the tasks they take on are different [2], [3], [5], [9]. This makes it particularly difficult for companies that would like to implement natural language dialog interfaces to predict how long a system will take to implement or how well the final system can be expected to work.

A measure of complexity for dialog system domains is a quantity that gives the difficulty of the dialog task, similar to time and space complexity for algorithms. With a complexity measure, new proposed systems can be compared to existing ones, predicting the expected performance of the finished product. Complexity provides a basis for studying and making claims about the scalability, predictability and robustness of dialog systems. Some complexity measures for dialog processing based on entropy have been proposed but are not yet widely used [7].

This work describes the Domain Complexity Analysis Tool (DomCAT). With this tool, the entropy of the set of possible user utterances can be easily calculated before or after system implementation and used as a dialog complexity measure. The complexity can be calculated based on either a corpus or a grammar. In addition, the tool can be used to guide the user in the creation of a new domain. System analyses for new domains can be completed and the complexity compared to existing dialog systems. Using DomCAT, all of this can be done without expert knowledge of either information theory or dialog processing.

Richard Tapia Celebration of Diversity in Computing Conference'07, October 14–17, 2007, Orlando, Florida, USA.

2. ENTROPY AS COMPLEXITY

Entropy has long been used in natural language processing as a measure of syntactic complexity, and it has recently been proposed as a measure of semantic and ambiguity complexity as well [7]. Entropy is the number of bits per symbol that are required to transmit information. Dialog complexity, then, is the expected bits of information per utterance transmitted from the user to the computer.

For a domain with sentences $S = \{s_1, s_2, ..., s_n\}$ and semantic forms $M=\{m_1, m_2, ..., m_k\}$ the *syntactic complexity(SynC), semantic complexity (SemC)* and *ambiguity complexity (AmbC)* of the domain are defined as follows [7]:

$$SynC = -\sum_{i=1}^{n} p(s_i)\lg(p(s_i))$$

$$SemC = -\sum_{i=1}^{k} p(m_i)\lg(p(m_i))$$

$$AmbC = -\sum_{i=1}^{n} p(s_i)\left[\sum_{j=1}^{k} p(m_j \mid s_i)\lg(p(m_j \mid s_i))\right]$$

Calculating these complexity values for a dialog domain requires defining all possible user utterances as well as the probability of each. The DomCAT tool gives a way to calculate the entropy based on a corpus of user interaction or a probabilistic grammar of syntactic and semantic forms[1]. In addition, the tool provides a way to aid a new dialog system-builder in creating the semantic and syntactic sets from sample dialog.

3. DomCAT FUNCTIONALITY

The DomCAT system implements the entropy calculations described above as well as aiding in creating the domain syntax and semantics. As the syntax and semantics are defined, the complexity is updated for comparison with other implemented systems and their corresponding evaluation measures. This tool allows the complexity theory to be put immediately into practice in analyzing domains, and the analyses can be completed without expert knowledge of the entropy calculations.

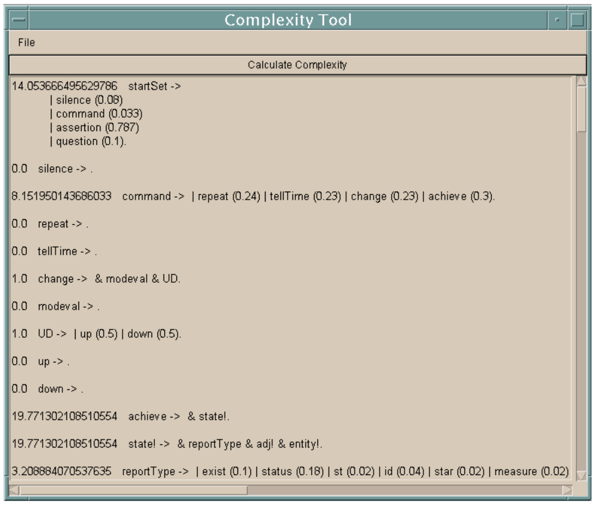
The tool can be used with whatever starting data the system builder has available. This could be a Complexity Grammar, which is a simple stochastic context-free grammar that can represent either semantics or syntax for a dialog system [7]. A corpus of actual or imagined system use may also be available. If no data at all is available, either of these can be built using DomCAT.

There are two ways to calculate the domain complexities described. A corpus can be used as a representative sample of the syntactic and semantic forms available in the domain. These calculations are straightforward from the definitions of complexity. The complexity of the domain can also be calculated based on grammars of the syntax and semantics in the domain. This is especially useful when the system is still in the developmental stages.

The system is started in grammar display mode. The analyst can load a completed Complexity Grammar into the program and calculate its complexity automatically, as seen in Figure 1. Notice that each line tells the complexity of that semantic grammar rule, and the complexity of the overall domain is the complexity of the start rule. If the analyst wants to change the existing grammar or to create a new one, he can enter the grammar building mode. The program prompts the analyst for a corpus of user statements for the domain. The system aids the analyst in creating semantics or syntax rules of the user

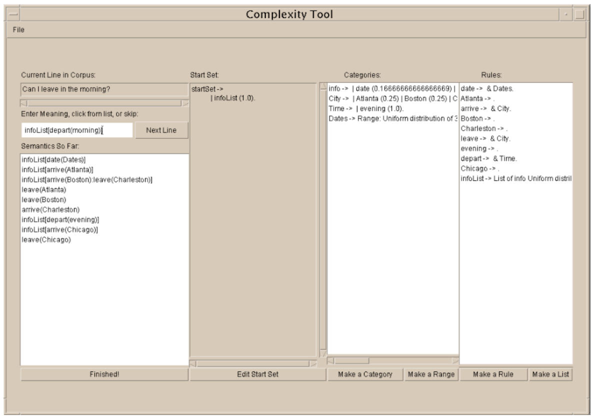
[1] Discussion of these algorithms is beyond the scope of this paper.

statement and calculates probabilities of the rules based on the corpus. In this way, the dialog complexity calculations can be done with as little expert knowledge as possible.



```
14.053666495629786   startSet ->
           | silence (0.08)
           | command (0.033)
           | assertion (0.787)
           | question (0.1).

0.0   silence -> .

8.151950143686033   command ->  | repeat (0.24) | tellTime (0.23) | change (0.23) | achieve (0.3).

0.0   repeat -> .

0.0   tellTime -> .

1.0   change ->  & modeval & UD.

0.0   modeval -> .

1.0   UD ->  | up (0.5) | down (0.5).

0.0   up -> .

0.0   down -> .

19.771302108510554   achieve ->  & state!.

19.771302108510554   state! ->  & reportType & adj! & entity!.

3.208884070537635   reportType ->  | exist (0.1) | status (0.18) | st (0.02) | id (0.04) | star (0.02) | measure (0.02)
```

Figure 1: Grammar Display Mode



Current Line in Corpus:
Can I leave in the morning?

Enter Meaning, click from list, or skip:
infoList[depart(morning)]

Semantics So Far:

```
infoList[date(Dates)]
infoList[arrive(Atlanta)]
infoList[arrive(Boston):leave(Charleston)]
leave(Atlanta)
leave(Boston)
arrive(Charleston)
infoList[depart(evening)]
infoList[arrive(Chicago)]
leave(Chicago)
```

Start Set:

```
startSet ->
      | infoList (1.0).
```

Categories:

```
info ->  | date (0.16666666666666669) |
City ->  | Atlanta (0.25) | Boston (0.25) | C
Time ->  | evening (1.0).
Dates -> Range: Uniform distribution of 3
```

Rules:

```
date ->  & Dates.
Atlanta -> .
arrive ->  & City.
Boston -> .
Charleston -> .
leave ->  & City.
evening -> .
depart ->  & Time.
Chicago -> .
infoList -> List of info Uniform distril
```

Figure 2: Grammar Building Mode

After creating the grammar, the system will switch back into the grammar display mode to calculate the resulting complexity. At any point a grammar can be loaded and then amended. The grammar that is already loaded is used as a starting point in the building process, and more sentences can be read in. The result is a system that can be used to incrementally approach the most accurate complexity given more and more information. To test its functionality, the DomCAT program was used to analyze several domains of existing dialog systems.

4. COMPLEXITY RESULTS

One of the benefits of DomCAT and the entropy-based measure of complexity is the ability to compare very different dialog systems based on whatever data is available. We have calculated the complexity of several domains according to the data available: either corpus-based or grammar-based, syntactic or semantic, human-human or human-computer.

The dialog systems that have been analyzed include the Duke Programming Tutor [6], an interactive, multi-media system that helps students learn to program in Pascal. The data for this system was from a Wizard of Oz experiment, meaning that the computer's statements in the dialog are actually controlled by a person without the user's knowledge. The next system analyzed is the JUST-TALK system [4], a dialog system used to train police officers to interact effectively with people with mental illness. The Circuit Fix-it Shop [8] is a spoken dialog system in which the computer aids in the repair of a toy circuit. The computer has information about wire configuration, voltages, and settings for making the circuit work correctly. Finally, we analyzed a pseudo-Wizard of Oz corpus from TRAINS-95 [1]. This dialog system asks the user to complete a routing of train cars and cargo. A pseudo-Wizard of Oz dialog is similar to a Wizard of Oz setup, but the user understands that he is speaking with another person.

To compare the systems, we created semantic Complexity Grammars for each system. The results, along with the types of available data, are given in Figure 3. The results show that the DomCAT system can be used to analyze real, sizeable dialog systems. These values can be used as a basis of comparison for other dialog systems.

Dialog System	**Available Data**	**Complexity**
Pascal Tutor	Wizard-of-Oz transcripts	6.067
JUST-TALK	System transcripts with semantics	9.096
Circuit Fix-it Shop	Syntax-to-semantics grammar and system transcripts	14.054
TRAINS	Pseudo-Wizard-of-Oz transcripts	16.217

Figure 3: Semantic Grammar-based Complexity Results

5. CONCLUSION

The area of dialog systems can greatly benefit from a theoretical framework to compare systems and evaluation results. The entropy-based complexity measures can provide this framework, and the DomCAT system allows these complexity values to be calculated easily, with no knowledge of information theory and basic knowledge of dialog processing.

6. REFERENCES

[1] Allen, J., Ferguson, G., Miller, B., and Ringger, E. Spoken dialogue and interactive planning. *Proceedings of the ARPA Spoken Language Systems Technology Workshop (SLST)*, 1995.

[2] Bates, M., and Aguso, D. A proposal for incremental dialogue evaluation. *Proceedings of the Speech and Natural Language Workshop,* 1991, 319-322.

[3] Charfuelan, M., Lopez, C., Gril J., Rodriguez, C., and Gomez, L. A general framework to assess spoken language dialogue systems: Experience with call center agent systems. *Proceedings of the Conference on Natural Language Processing Workshop on Robust Methods in Analysis of Natural Language Data.* 2000.

[4] Frank, G., Guinn, C., and Hubal, R. JUST-TALK: An application of responsive virtual human technology. *Proceedings of the 24th Intersevice/Industrial Training, Simulation, and Education Conference,* 2002.

[5] James, F., Rayner, M., and Hockey, B. *"Do that again": Evaluating speech dialogue interfaces.* Technical Report 00.06, Research Institute for Advanced Computer Science, 2000.

[6] Keim, G., Fulkerson, M., and Biermann, A. Initiative in tutorial dialogue systems. *Proceedings of the American Association for Artificial Intelligence (AAAI) Spring Symposium on Computational Models for Mixed-Initiative Interaction,* 1997.

[7] Pollard, S. *Defining the Complexity of Natural Language Dialog System Domains.* Ph.D. Thesis, Duke University, Durham, NC, 2006.

[8] Smith, R. and Hipp, D. *Spoken Natural Language Dialog Systems: A Practical Approach.* Oxford University Press, NY, 1994.

[9] Walker, M., Litman, D., Kamm, C., and Abella, A. PARADISE: A framework for evaluating spoken dialogue agents. *Proceedings of the 1997 Meeting of the Association for Computational Linguistics and European Chapter of the Association for Computational Linguistics (ACL/EACL),* 2000, 28-34.