2006 CCRTS

THE STATE OF THE ART AND THE STATE OF THE PRACTICE

Title: Determining Course of Action Alignment with Operational Objectives

Topics: C2 Modeling and Simulation, C2 Analysis, C2 Experimentation

Authors: Duane Gilmour, Zhongfei (Mark) Zhang

POC: Duane Gilmour

Organization: Air Force Research Laboratory

Address: Air Force Research Laboratory, IFTC
525 Brooks Road
Rome, NY 13441-4505

Phone: 315-330-3550

Fax: 315-330-2953

Duane.Gilmour@rl.af.mil

Report Documentation Page

Form Approved
OMB No. 0704-0188

Public reporting burden for the collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington VA 22202-4302. Respondents should be aware that notwithstanding any other provision of law, no person shall be subject to a penalty for failing to comply with a collection of information if it does not display a currently valid OMB control number.

1. REPORT DATE **JUN 2006**	2. REPORT TYPE	3. DATES COVERED **00-00-2006 to 00-00-2006**

4. TITLE AND SUBTITLE
Determining Course of Action Alignment with Operational Objectives

5a. CONTRACT NUMBER

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER

6. AUTHOR(S)

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Air Force Research Laboratory (IFTC),525 Brooks Road,Rome,NY,13441-4505

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

10. SPONSOR/MONITOR'S ACRONYM(S)

11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution unlimited

13. SUPPLEMENTARY NOTES
The original document contains color images.

14. ABSTRACT

15. SUBJECT TERMS

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES **40**	19a. NAME OF RESPONSIBLE PERSON
a. REPORT **unclassified**	b. ABSTRACT **unclassified**	c. THIS PAGE **unclassified**			

Determining Course of Action Alignment with Operational Objectives

Duane Gilmour*, Zhongfei (Mark) Zhang**
Duane.Gilmour@rl.af.mil, zhongfei@cs.binghamton.edu

*Air Force Research Laboratory, Information Directorate
Advanced Computing Technology Branch
525 Brooks Road, Rome, NY 13441-4505

** Department of Computer Science
Watson School of Engineering and Applied Sciences
Binghamton University
Binghamton, NY 13902-6000

Abstract

During the military planning process, commander's intent and objectives are defined and courses of action (COAs) are developed, analyzed and compared to determine their likelihood of achieving the intent and objectives. For each mission, thousands of COAs could be automatically generated but only those in alignment with commander's objectives are worth investigating. The challenge is to be able to automatically determine alignment, given that there is a *semantic gap* for a specific pair of objective and COA. The two not only differ syntactically, but also semantically. In this research, we made two specific contributions towards developing a solution to this problem. First, we discovered that classic symbolic reasoning does not work in developing such a solution, as the semantics involved are always fuzzy and inexact. Second, under the assumptions that both the operational objective and the COAs are represented in a low level semantic hierarchy (such that there is a syntax to represent them in terms of languages), we developed a solution that identifies their alignment as well as divergence. This paper presents results of this research, along with results from testing the proposed solution on a small, hand-crafted ontology.

Keywords: commander's intent, course of action, COA analysis, semantic inference, fuzzy logic

1.0 Introduction

The military planning process depends upon analysis systems to be able to anticipate and respond in real-time to a dynamically changing battlespace with counteractions. Complex technical challenges exist in developing automated processes to derive hypotheses about future alternatives for mission scenarios. The military conducts combat operations in the presence of uncertainty and the alternatives that might emerge. It is virtually impossible to identify or predict the specific details of what might transpire. Plans and strategies, which result in COAs, are evaluated to determine the necessary steps to meet the overall strategic objectives. COA analysis is the process of performing "what if" analysis of actions and reactions and is designed to visualize the flow of the battle and evaluate each friendly COA. Due to the dynamic nature of military campaigns, COAs are continuously generated, developed and analyzed prior to execution. For each mission, thousands of COAs could be automatically generated. Clearly, it is neither possible nor necessary to analyze or execute all of the COAs. Also, it is time prohibitive in current fast paced campaigns to evaluate COAs that don't achieve commander's intent. Instead, prior to analysis and execution of a COA, it must be determined whether a particular COA is in alignment with the commander's intent and objectives for the mission. This capability is extremely important and plays a critical role in dominating the battlefield and consequently succeeding in the campaign. This research addresses the investigation of developing this capability.

There are five fundamental issues that must be considered when developing COAs. A valid COA should be suitable, feasible, acceptable, distinguishable and complete [1]. A COA is suitable if it is in alignment with commander's intent and will accomplish the mission when carried out successfully. A COA is feasible if it can be achieved with the given resources. A COA is acceptable if it balances cost and risk with advantage gained through execution. A COA is distinguishable if it is significantly different from others and a COA is complete if it incorporates major operations and tasks to be accomplished to accomplish the desired end state. The first issue, suitability, relates to the semantic inference on whether the COA matches the commander's intent and will be the focus of this paper.

2.0 Background

A commander's intent is defined in terms of the goal and the end state. The goal is what the military campaign is expected to achieve. The end state is what the conditions are expected to be after the military campaign is over. Due to the existence of a typical military administrative hierarchy in command of a specific military campaign, a commander's intent may also be represented in different levels of a hierarchy, from the strategic level through the operational level to the tactical level.

The strategic level of the commander's intent refers to a high level commander's intent, such as the president's intent. An example of the strategic level intent is shown in Figure 1.

```
We will liberate Orangeland, restore power and control
to her rightful government, and then punish the
aggressor nation for its unlawful attack and
occupation by significantly reducing his ability to
wage war such that he is no longer a regional threat.
```

Figure 1: An example of a strategic level commander's intent.

The operational level commander's intent refers to the actual execution commander's intent, i.e., the intent of the commander in charge of the specific military campaign. At this level, the commander's intent may be represented in several ways, such as end state, purpose, method, and risk. An example of an operational level commander's intent is shown in Figure 2.

End state: a. freedom to operate forces starting with pre-deployment activities; b. no Weapons of Mass Destruction (WMD)/Theater Ballistic Missile (TBM) or terrorist threat to region/US
Purpose: regional stability and US security
Method: Global Strike Task Force initial strikes followed by Air and Space Epeditionary Task Force/Carrier Battle Group persistence forces
Risk: low to US forces; medium for collateral damage.

Figure 2: An example of an operational level commander's intent.

The tactical level of commander's intent refers to the specific objectives that the staff of the commander in charge of the military campaign has outlined in terms of the operational level commander's intent. An example of the tactical level commander's intent is shown in Figure 3.

Disrupt enemy TBM Command & Control (C2) systems.

Figure 3: An example of the tactical level commander's intent.

On the other hand, a COA actually represents a specific possible option in order to achieve a military mission, and therefore, it may also be represented in a hierarchy at different levels of execution. For example, a higher level COA may be "attack WMD and TBM power" while a lower level COA may be "move FA-18 at speed 500 through route 21". Consequently, a COA may consist of several lower level granularity COAs in sequence.

This research only addresses the suitability issue of COA analysis. In other words, given a commander's intent and a COA, the problem is to determine whether the COA is in alignment with the commander's intent, and if not, how far the COA diverts from the commander's intent. The challenge is that typically there are always semantic uncertainty and fuzziness for both commander's intent and COAs. This semantic uncertainty and fuzziness demand that not only natural language be correctly understood, but also the semantic meaning of each word be correctly understood, given the different context in different application. For example, referring to Figure 1, what do "control" and "ability significantly reduced" exactly mean? Due to this semantic uncertainty and fuzziness, there is a semantic gap between the commander's intent and a COA; the challenge is to develop a solution that overcomes this semantic gap.

In order to address the semantic uncertainty and fuzziness, we developed a fuzzified approach to semantic inference for COA analysis, called CAFSIN, which stands for COA Analysis based on Fuzzified Semantic INference. We demonstrated the effectiveness of the CAFSIN method through preliminary testing and evaluations, and present the results here.

3.0 Related Work

COA analysis has received attention in recent military campaign research for years. COA analysis was studied through computer-generated forces in simulation using cognitive modeling [2]. Based on individual cases, they used cognitive modeling to attempt to develop a generalized strategy for COA analysis using simulations. In a joint research project on COA analysis between Army Research Laboratory (ARL) and Ohio State University (OSU), the multi-criterial decision tool developed at OSU was used to mine ARL combat simulation data in order to gain the battle-planning insights into understanding the COA space [3]. The approach taken in this work is more related to data mining and visualization through user interaction to develop such insights. In a related work, as reported in [4], an effort was accomplished to survey several existing tools for visualizing COA analysis results, including the OSU developed tools. In addition, researchers combined the existing tools together using coevolution and Pareto optimization for COA analysis.

Situation assessment and COA selection were studied [5] using a Commander Model under the Joint Warfare System environment [6]. Fuzzy rules are used due to the typical fuzzy nature of the commander's intent, and users are provided with the ability to modify both the input parameters and the underlying rules. A software system for COA development and analysis was reported on in [7] based on colored Petri Nets [8]. The colored Petri Net model is used in this study to specify the execution and analysis of tasks in a COA.

Recently, COA analysis has been investigated in the context of real-time decision support at the Air Force Research Laboratory (AFRL). The current status of COA analysis was reported on in [9, 10] as well as the approaches AFRL is taking on real-time COA analysis. Preliminary simulation results are reported using high performance computing facilities to achieve real-time COA analysis. A data representation ontology was reported on in [11] and the related schema developed for the COA analysis at AFRL.

It is well-observed [12] that in many real-world problems, classic symbolic reasoning [13, 14] may not work, and consequently, the research on uncertainty reasoning [12, 15] has received intensive attention.

While much of this work relates to COA development and analysis, it fails to address the issue of COA suitability that was mentioned previously.

4.0 CAFSIN Solution

Due to the great challenge of the semantic gap between the commander's intent and a COA, as for the first phase of this investigation, we have made the following assumptions to simplify the solution:

1. The commander's intent is given at the tactical level. This allows a restrictive syntax to be used.
2. The COA is also given in a lower, more specific level. This also allows a restrictive syntax to be used.
3. A domain ontology must be given.

Based on these assumptions, we developed the CAFSIN solution. This solution models the determination of the alignment problem between a commander's intent and a COA as a fuzzified language matching problem. This is a general approach to COA analysis and reasoning because it addresses the uncertain and fuzzy nature of the problem using fuzzy logic analysis, and consequently, the solution leaves a user to define what is considered as a compliant or a diverting

COA. Even though CAFSIN is developed under the assumptions made above, it may also work when the assumptions are relaxed if reliable information extraction (IE) tools are available.

4.1 Ontology Construction

In order to facilitate the search in CAFSIN, the following issues were considered when constructing the ontology:

1. Synonymy: all the synonyms are hard-wired together in a node in the ontology (e.g., “the Pentagon” and “DoD” are wired together as the same word and are represented as a single node).
2. Polysemy: words in different meanings in the ontology are represented in different nodes (e.g., “chair” as a department chair and “chair” as a piece of furniture are represented and located as separate words).
3. Special names: special names and phrases are coined as single words in the ontology (e.g., “WMD support system” as one word).

Given an ontology with these requirements satisfied, a standard hashing function may be used to directly identify a specific node in the ontology.

4.2 Fuzzified Word Similarity

Given two words *w1* and *w2* and an ontology *Ψ*, the similarity function *f* is defined as a Gaussian function:

$$f(w1, w2 \mid \Psi) = \frac{p}{\sqrt{2\pi\sigma^2}} e^{-\frac{(\max\{d_{w1}, d_{w2}\}-1)^2}{2\sigma^2}} \tag{1}$$

where d_{w1} and d_{w2} are the depths of *w1* and *w2* from a nearest common ancestor in *Ψ*; if they do not share a common ancestor, they are set as ∞; *p* is the normalization factor; and σ is the standard deviation.

Based on the definition of this fuzzified word similarity function, given an ontology, the similarity between two words depends on two things: (1) the relative depth difference between the two words in the ontology; and (2) the depth from the nearest common ancestor in the ontology.

Thus, two words have a strong similarity if they are synonyms, or siblings sharing a common parent, or one is a parent of the other. The similarity decreases if the depth difference between the two words increases in the same ontology tree; and/or their nearest common ancestor moves away. The similarity becomes 0 if the two words do not have a common ancestor, i.e., they are located in different ontology trees.

4.3 Language Models

Since we have assumed that the commander's intent is represented as a tactical level command, and since at the tactical level, commands may be represented in a well-defined syntax, we use the following grammar as the language model for the commander's intent:

$$T = <verb> <noun>*+ \tag{2}$$

Similarly, a COA may be represented as a language sentence with the following grammar:

$$C = \{<verb> <attribute\ value>*\}+ \tag{3}$$

4.4 CAFSIN Similarity Function

Now we are ready to define the CAFSIN similarity function based on our CAFSIN design principle. Let $t \in T$, $t = v\ n*$; let $c \in C$, $c = \{u\ m*\}+$. Then the CAFSIN similarity function is defined as:

$$h(t,c \mid \Psi) = \alpha \Sigma_u f(v,u \mid \Psi) H(n^*, m^*) \tag{4}$$

where *H(n*,m*)* is a fuzzified maximum substring matching function between word string *n** and word string *m** using the fuzzified word similarity function *f(n,m|Ψ)*; α is a normalization factor.

Now the next question is how to compute the fuzzified maximum substring matching function *H*. It is well known that the substring matching problem is NP-complete, and an optimal solution may be found using dynamic programming. Since the actual *t* and *c* typically only have a very few words, the complexity is not an issue.

Assume that there are *N* words for the string *n**, and that there are *M* words for string *m**. Using the dynamic programming technique, we create a table of *H[N+1, M+1]*, and the table *H* is initialized as *H[0, j]=0* for *j=1, … , M+1*; and *H[i, 0]=0* for *i=1, … , N+1*. Thus, the rest of the entries in *H* are computed based on the following recurrence:

$$H[i,j] = \begin{cases} f(n[i],m[j]),\ f(n[i],m[j]) > \max(H[i-1,j],H[i,j-1]) \\ \max(H[i-1,j],H[i,j-1]), otherwise \end{cases} \tag{5}$$

5.0 Proof-of-Concept Testing

The CAFSIN method has been tested with a hand-crafted ontology, which is shown in part in Figures 4 and 5.

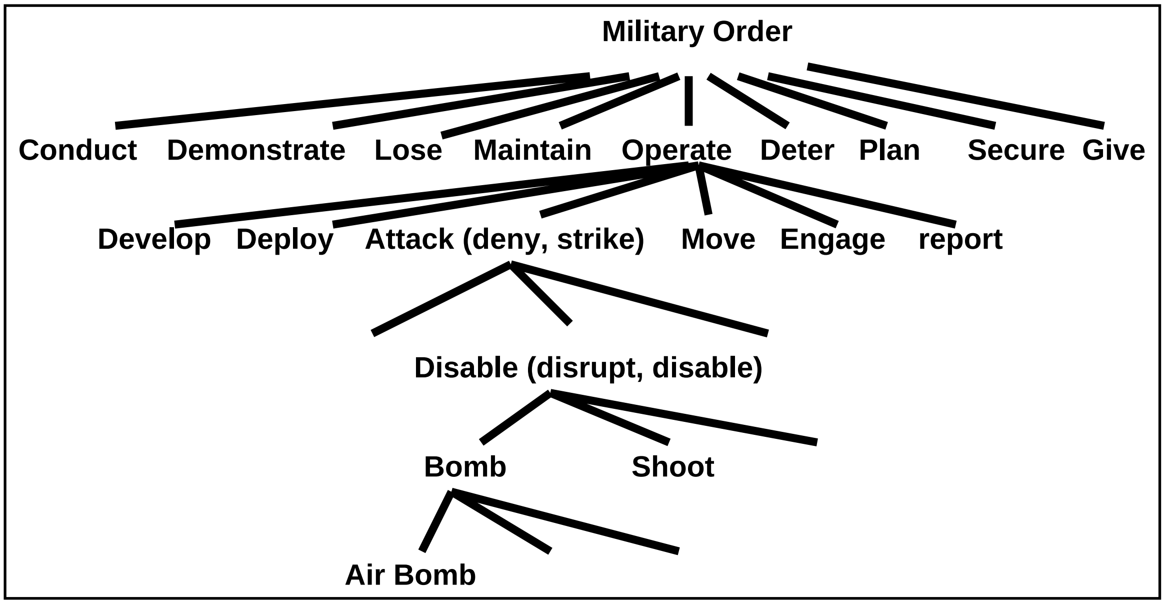


Figure 4: Part of the ontology used in the preliminary testing.

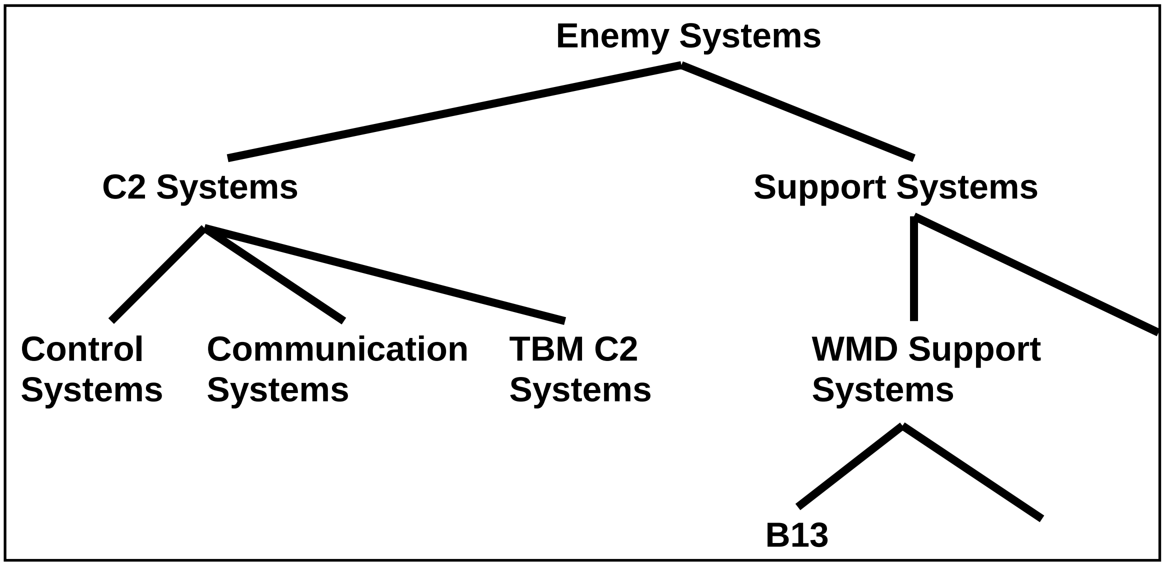


Figure 5: Another part of the ontology used in the preliminary testing.

As a first example, assume that we have a commander’s intent as “disrupt enemy’s WMD support system” and a COA as “engage FA-18 target B13”. After the standard text processing, we have the two strings for the commander’s intent and the COA, as represented in Figure 6.

t = {<disrupt>,(<enemy>,<WMD support system>)}

c = {<engage>,(<FA-18>,<target>,<B13>)}

Figure 6: The commander’s intent and the COA strings after text processing.

Based on the ontologies shown in Figures 4 and 5, we have obtained the distances between the relevant word pairs which are shown in Table 1.

Assuming the parameters of σ=1, $p=\sqrt{2\pi}$, α=1; from Eq. 1 we compute the fuzzified similarity values for the relevant word pairs as shown in Table 2. Finally, the *H* function is computed using dynamic programming based on Eq. 5 as denoted in Table 3, and the final similarity function value between the pair of commander's intent and the COA is determined based on Eq. 4 as 0.607.

Table 1: Ontology distances between the words in the example.

d<disrupt> = 2	d<engage> = 1
d<enemy> = ∞	d<FA-18> = ∞
d<enemy> = ∞	d<target> = ∞
d<enemy> = ∞	d<B13> = ∞
d<WMD support system> = ∞	d<FA-18> = ∞
d<WMD support system> = ∞	d<target> = ∞
d<WMD support system> = 0	d<B13> = 1

Table 2: Similarity values between the words in the example.

f(<disrupt>, <engage>) = 0.607
f(<enemy>, <FA-18>) = 0
f(<enemy>, <target>) = 0
f(<enemy>, <B13>) = 0
f(<WMD support system>, <FA-18>) = 0
f(<WMD support system>, <target>) = 0
f(<WMD support system>, <B13>) = 1

Table 3: The dynamic programming table for computing *H* in the example.

		enemy	WMD support system
	0	0	0
FA-18	0	0	0
target	0	0	0
B13	0	0	1

As another example, we have the same commander’s intent as in the first example, but the COA has one more action “bomb target B13” in addition to the original action in the first example. After standard text processing we have the strings specified in Figure 7.

t = {<disrupt>,(<enemy>,<WMD support system>)}
c = {<engage>,(<FA-18>,<target>,<B13>);
<bomb>,(<target>,<B13>)}

Figure 7: Another example of commander’s intent and COA specified as strings after text processing.

The similarity computation for the pair between the commander’s intent and the first action *{<engage>,(<FA-18>,<target>,<B13>)}* of the COA is identical to the one computed in the first example, and is equal to 0.607. In order to compute the similarity for the pair between the same commander’s intent and the second part of the COA, we first determine the ontology distance based on the ontology described in Figures 4 and 5, as reported in Table 4. The word pair similarity values are then computed using Eq. 1 and shown in Table 5, and finally the H function is computed using dynamic programming based on Eq. 5 shown in Table 6. Thus, the final similarity value between the commander’s intent and the COA is the summation of the two parts which becomes 1.607. Clearly, due to the addition of the second part in the COA, the similarity of this COA to the commander’s intent is much higher than the one in the first example, which indicates that the second COA is more in alignment with the commander’s intent than the first COA.

Table 4: Ontology distance between the words for the second pair in the second example.

d<disrupt> = 0	d<bomb> = 1
d<enemy> = ∞	d<target> = ∞
d<enemy> = ∞	d<B13> = ∞
d<WMD support system> = ∞	d<target> = ∞
d<WMD support system> = 0	d<B13> = 1

Table 5: Similarity values between the words for the second pair in the second example.

f(<disrupt>, <bomb>) = 1
f(<enemy>, <target>) = 0
f(<enemy>, <B13>) = 0
f(<WMD support system>, <target>) = 0
f(<WMD support system>, <B13>) = 1

Table 6: Dynamic programming table for computing *H* for the second pair in the second example.

		enemy	WMD support system
	0	0	0
target	0	0	0
B13	0	0	1

As yet another example, let's stay with the same commander's intent used in the previous two examples but the COA now becomes "lose target B13". After the standard text processing, we have the strings specified in Figure 8.

t = {<disrupt>,(<enemy>,<WMD support system>)}
c = {<lose>,(<target>,<B13>)}

Figure 8: The strings specified for the commander's intent and the COA after standard text processing in the third example.

Now the same CAFSIN method is applied to this example and reported in Tables 7 – 9. The final similarity value between the commander's intent and the COA in this example is 0.135, assuming the same parameters are used as in the first two examples. This similarity value is lower than the first two examples (0.607, 1.607). This indicates that the COA does not align with the commander's intent as well as the first two COAs. It may also indicate that the COA diverges from commander's intent and should not be considered further.

Table 7: Ontology distance between the words in the third example.

d<disrupt> = 3	d<lose> = 1
d<enemy> = ∞	d<target> = ∞
d<enemy> = ∞	d<B13> = ∞
d<WMD support system> = ∞	d<target> = ∞
d<WMD support system> = 0	d<B13> = 1

Table 8: Similarity values between the words in the third example.

f(<disrupt>, <lose>) = 0.135
f(<enemy>, <target>) = 0
f(<enemy>, <B13>) = 0
f(<WMD support system>, <target>) = 0
f(<WMD support system>, <B13>) = 1

Table 9: Dynamic programming table for computing *H* in the third example.

		enemy	WMD support system
	0	0	0
target	0	0	0
B13	0	0	1

The above three examples demonstrate the effectiveness of using CAFSIN method to determine the alignment between a pair of given commander's intent and COA, and showcases the proof of the concept for the CAFSIN method. Using the ontology we have hand-crafted shown in Figures 4 and 5, we have tested numerous examples for different commander's intent and COAs and the experiments all show that CAFSIN is a very effective method to serve the purpose of determining whether a COA is in alignment with a commander's intent and by how much relative to comparable COAs.

6.0 Conclusion and Future Work

In this research, we have reviewed the relevant literature regarding the problem of determining the alignment of a course of action with a commander's intent, and have found that the solution to this problem must address explicitly the uncertainty and fuzziness issues in the reasoning process. Consequently, we have concluded that the classic symbolic reasoning does not work. Instead, we developed a specific but general solution to the problem that is based on fuzzified semantic inference called CAFSIN. We demonstrated that CAFSIN is an effective method to solve the problem through proof-of-concept testing.

CAFSIN has the following advantages:

- It is independent of the ontology; we can apply it to any domain to solve the problem.
- It is independent of specific words; the similarity between the words is only dependent on the relative locations between the words in ontology.
- It is always relative; this allows users interaction based on their experience and expertise.

The future work includes:

- **Large scale evaluation:** Large scale evaluation is necessary in order to actually turn CAFSIN method into a real world technology. However, there are several issues that we must address before we are able to conduct large scale evaluations. The first is the construction of the ontology for a specific application domain. How to construct such a reasonably large ontology is always a question. We may use the existing general domain ontology such as Wordnet. But how to tailor the general ontology to the specific application domain also becomes an issue. Finally, even if we have an ontology and a data set ready for the evaluations, how to define the evaluation metrics is another issue.

- **Relaxing the assumptions:** There are two directions that we can go for relaxing the current assumptions of CAFSIN. The first one is to apply CAFSIN to higher level of commander's intent and COA. In this case we need tools for natural language processing. If we have reliable IE tools available, the direct extension of CAFSIN to this scenario is not a problem; the problem now becomes how to interface the IE tools into the CAFSIN method. The second direction is to relax the syntax of the COA to incorporate constraints into the COA. This involves revising the CAFSIN strings matching function.

- **Improving the computation complexity:** There are two bottlenecks to the complexity of the CAFSIN method. The first is the fuzzified substring matching to compute the H function. Though we claim that typically the strings are not very long and so the complexity would not be a problem. However, in case the strings become very long, we may need to add heuristics into the matching to expedite the substring search. The second bottleneck is the ontology tree search for identifying the correct location of the word. Given the typical scenario that the words in a given COA may be located "close enough" in the ontology, we may be able to add some locality analysis into the ontology tree search for reducing the tree traversal search time.

7.0 Acknowledgements

This material is based on research sponsored by the Air Force Research Laboratory under grant number FA8750-05-1-0234. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation thereon. The authors would like to acknowledge the contributions to this project made by Mr. Martin J. Walter and Mr. William E. McKeever (Air Force Research Laboratory, Information Directorate).

References

[1] Joint Publication 5-00.2, "Joint Task Force Planning Guidance and Procedures", January 1999.

[2] Chandrasekaran and Josephson, "Cognitive Modeling for Simulation Goals: A Research Strategy for Computer-Generated Forces", *Proceedings of the 8th Conference on Computer Generated Forces and Behavior Representation*, 1999.

[3] Kaste, O'May, Heilman, Chandrasekaran and Josephson, "From Simulation to Insights: Experiments in the Use of a Multi-Criterial Viewer to Develop Understanding of the COA Space", *Proceedings of US Army Research Laboratories Collaborative Technology Alliances Symposium*, 2003.

[4] Hillis, Barnes, Suantak, Schlabach, Chandrasekaran, Josephson and Carroll, "Collaborative Visualization Tools for Courses Of Action (COA) in Anti-Terrorist Operations: Combining Coevolution and Pareto Optimization", *Proceedings of US Army Research Laboratories Collaborative Technology Alliances Symposium*, 2003.

[5] Vakas, Prince, Blacksten and Burdick, "Commander Behavior and Course of Action Selection in JWARS", *Proceedings of the 2001 Winter Simulation Conference*, 2001.

[6] Joint Warfare System Commander Behavior Model, *Phase I TechReport*, Joint Warfare System Office, CACI, 2001.

[7] Zhang, Kristensen, Janczura, Gallasch and Billington, “A Coloured Petri Net Based Tool for Course of Action Development and Analysis”, *Proceedings of the Workshop on Formal Methods Applied to Defence Systems*, 2002.

[8] Jensen, *Coloured Petri Net: Basic Concepts, Analysis Methods and Practical Use*, Volumes 1 – 3, Springer Verlag, 1992 – 1997.

[9] Gilmour, Hanna, Koziarz, McKeever and Walter, “High-Performance Computing for Command and Control Real-Time Decision Support”, *AFRL Technology Horizons®*, vol 6 no 1, February 2005.

[10] Gilmour, Hanna, McKeever and Walter, “Real-Time Course of Action Analysis”, *Proceedings of the 10th International Command and Control Research and Technology Symposium*, June 2005.

[11] Hanna, Reaper, Cox and Walter, “Course of Action Simulation Analysis”, *Proceedings of the 10th International Command and Control Research and Technology Symposium*, June 2005.

[12] Kyburg, “Uncertain Inferences and Uncertain Conclusions”. *Proceedings of the 12th Conference on Uncertainty in Artificial Intelligence*, 1996.

[13] Fikes, Jenkins and Zhou, “Including Domain-Specific Reasoners with Reusable Ontologies”, *Proceedings of the International Conference on Information and Knowledge Engineering*, 2003.

[14] Fikes, Frank and Jenkins, “JTP: A System Architecture and Component Library for Hybrid Reasoning”, *Proceedings of the 7th Multi-Conference on Systems, Cybernetics, and Informatics*, 2003.

[15] Harrington, Banks and Santos, “The PESKI Intelligent User Interface”, *TechReport*, Department of Electrical and Computer Engineering, AFIT, 1996.

DETERMINING COURSE OF ACTION ALIGNMENT WITH OPERATIONAL OBJECTIVES

Duane Gilmour
Information Directorate
Air Force Research Laboratory

Prof. Mark Zhang
SUNY Binghamton

Outline

- Background
- Problem statement
- Course of action analysis based on fuzzified semantic inference
- Preliminary proof-of-concept testing
- Summary
- Future work

Motivation

Current Course of Action Analysis Limitations

- **Dynamic Course of Action (COA) analysis is manpower intensive (blue / red teaming)**
- **Automated COA analysis technology**
  - **Static, adversary is pre-scripted**
  - **Attrition based, force-on-force**
  - **Utilized to study scenarios well in advance of operations**
- **Approaches are too laborious and slow for current fast paced operations**
  - **Adversaries act / react / adapt too quickly**
  - **Need an “always on” capability**

Real-Time COA Analysis

- Challenge: Efficient generation and analysis of a range of COA alternatives to anticipate and shape the battlespace
  - Prior to and during operations
- Technology development to support dynamic real-time COA analysis
  - In-house force structure simulation high performance computing (HPC) R&D testbed
  - Effects based / center of gravity modeling
  - Automated scenario generation
  - Modeling intelligent adversary behaviors
  - HPC framework for rapid decision branch analysis
  - COA simulation analysis

“Static” vs. “Dynamic” Simulation

- **“Static” simulation: traditional use of simulations**
  - **Use simulations to study COAs well in advance**
  - **Get general idea of what might happen if similar scenario actually occurs**
- **“Dynamic” simulation: novel use of simulations**
  - **Use simulations to assist decision makers** *while the scenario is happening*
  - **Quickly simulate ahead to glimpse possible futures**
  - **Evaluate possible COAs and multiple decision points within each COA**
  - **Dynamic situational assessment during combat operations, comparison against plans, alerts on new threats or opportunities**

Challenge Problem

- COAs must be continuously developed and analyzed to support operations
- Automated systems can generate (thousands) COAs
- Prior to COA development, analysis and execution, need to determine which generated COAs are aligned with the missions commander’s intent
- Objective: develop a representation scheme for COA generation and assessment to rapidly compare generated COAs to commander’s intent

Comparing Commander’s Intent and COAs

- Commander’s intent may be represented in a hierarchy; strategic to tactical
- COAs may be represented in a hierarchy at different levels of execution; strategic to tactical
- Semantic uncertainty and fuzziness of commander’s intent and COAs
  - e.g., peace, control, ability significantly reduced
  - Correctly understanding the natural language
  - Classic symbolic reasoning does not work
- Semantic gap between the typical higher level commander’s intent and the lower level COA

Assumptions

- Commander’s intent is given at the tactical level
  - A restrictive syntax may be assumed
- COA is also given in a lower, more specific level
  - A restrictive syntax may be assumed
- A domain ontology is given

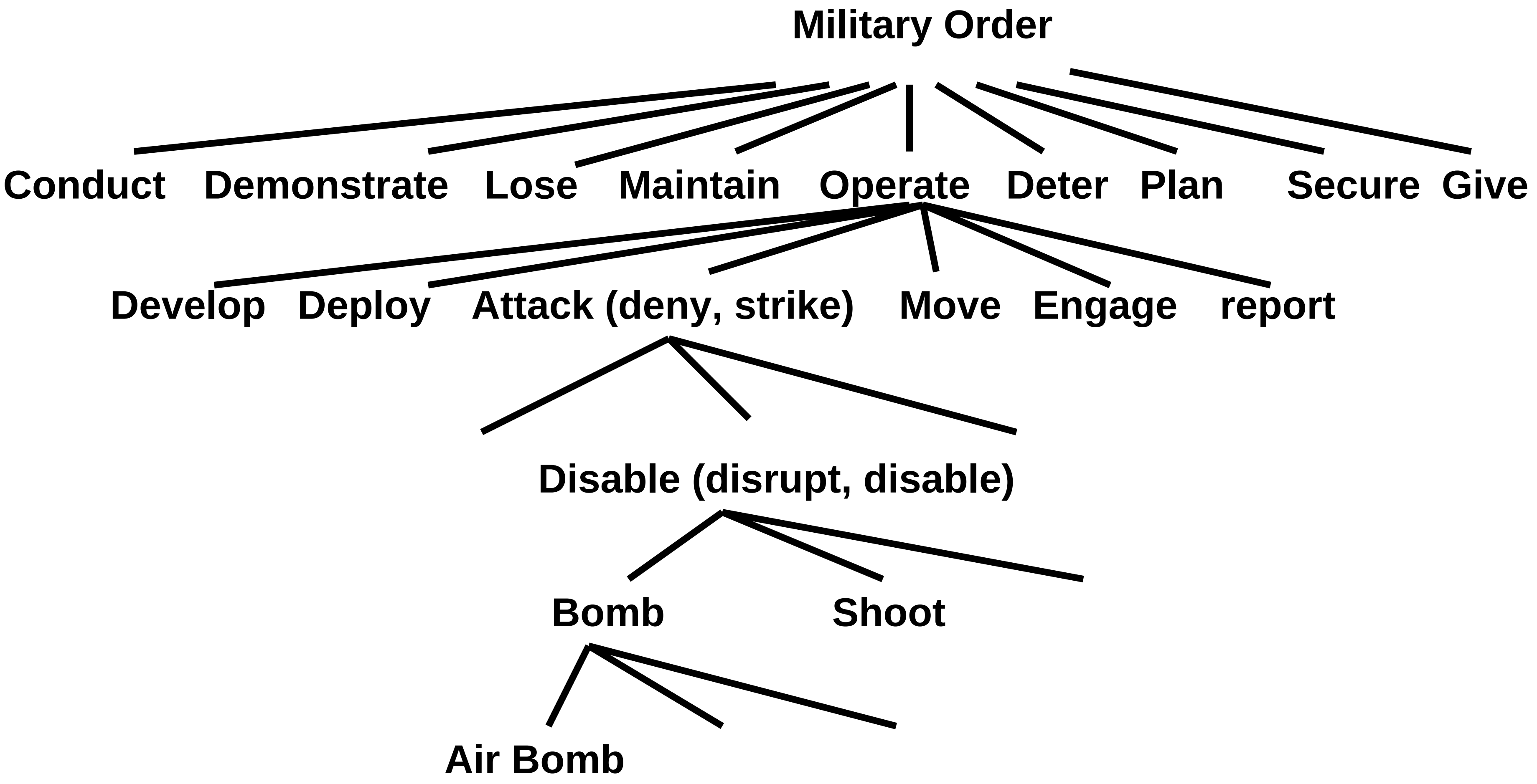
Course of action Analysis based on Fuzzified Semantic INference (CAFSIN)

- **Model the determination of COA alignment with a commander's intent as a fuzzified language matching problem**
- **A general approach to COA analysis and reasoning**
- **Take into account the fuzzy nature of COA and commander's intent ➔ uncertain and fuzzy reasoning**
- **Leave a user to define what is considered as compliant COA or diverting COA**
- **Works even when the assumptions are relaxed, if reliable information extraction tools are available**

Ontology Construction

- Build an ontology in a given domain
  - Synonymy: all the synonyms are hard-wired together in a node
  - Polysemy: words with different meanings in the ontology are represented in different nodes
- Standard hashing function used to directly link to a node in the ontology
- Special phrases are coined as single words (e.g., WMD support system)

Ontology Example



Military Order

Conduct Demonstrate Lose Maintain Operate Deter Plan Secure Give

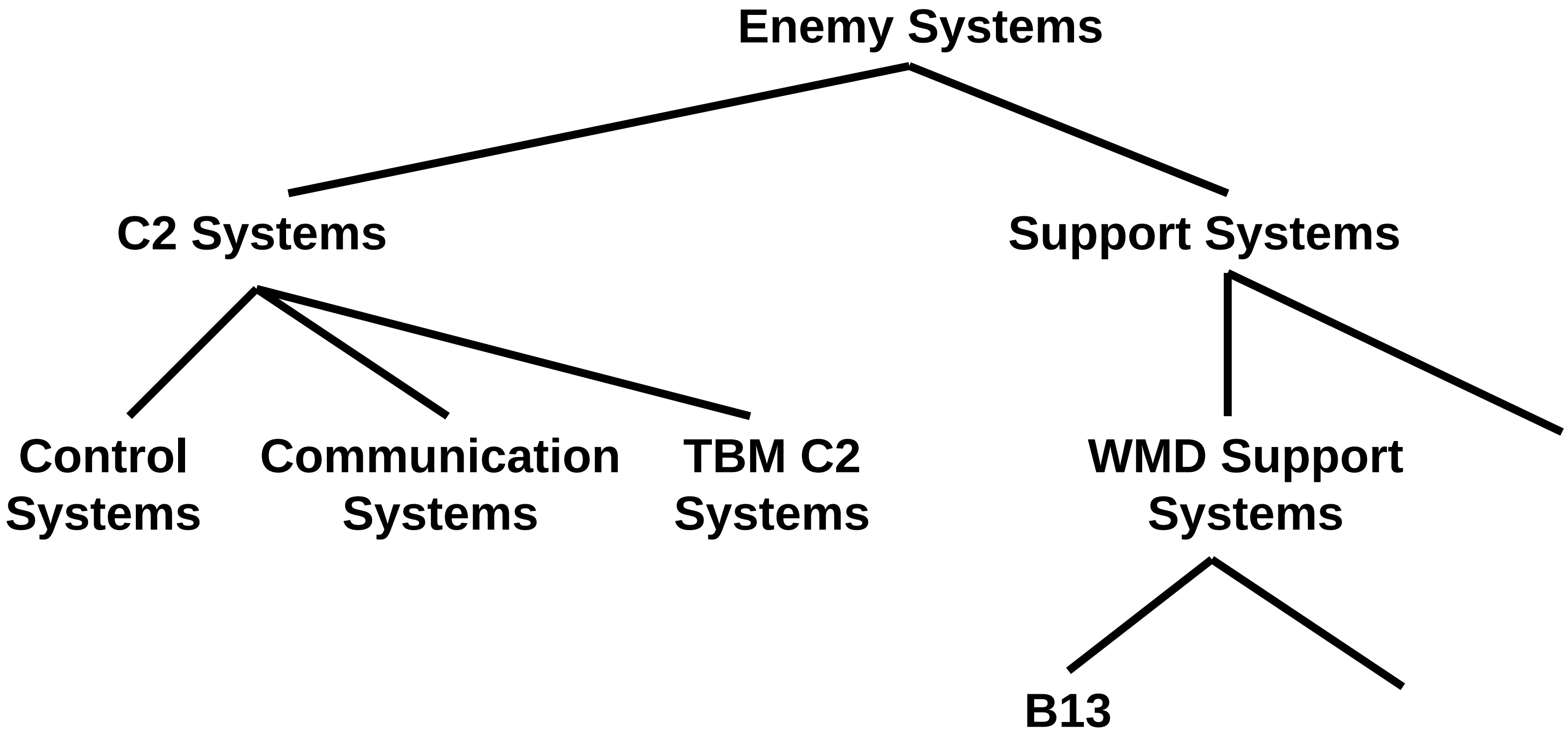
Develop Deploy Attack (deny, strike) Move Engage report

Disable (disrupt, disable)

Bomb Shoot

Air Bomb

Ontology Example, Cont’d



Enemy Systems

C2 Systems

Support Systems

Control Systems

Communication Systems

TBM C2 Systems

WMD Support Systems

B13

Fuzzified Word Similarity

- **Given two words *w1* and *w2* and an ontology Ψ, the similarity function *f* is defined as a Gaussian function:**

$$f(w1, w2 \mid \Psi) = \frac{p}{\sqrt{2\pi\sigma^2}} e^{-\frac{(\max\{d_{w1}, d_{w2}\}-1)^2}{2\sigma^2}}$$

where d_{w1} and d_{w2} are the depths of *w1* and *w2* from a nearest common ancestor in Ψ, if they do not share a common ancestor, they are set as ∞, *p* is the normalization factor, σ is the standard deviation

- **Given an ontology, the similarity between two words depends on two things:**
  - **The relative depth difference between the two words in the ontology**
  - **The depth from the nearest common ancestor in the ontology**

Word Similarity Function

- **Two words have strong similarity if they are:**
  - – Synonyms
  - – Siblings sharing the common parents
  - – One is a parent of the other
- **The similarity decreases if:**
  - – The depth difference between the two words increases in the same ontology tree; and/or
  - – Their nearest common ancestor moves away
- **The similarity is 0 if the two words do not have a common ancestor, i.e., they are located in different ontology trees**

Modeling the Tactical Objective and COA

- The tactical objectives of a commander’s intent typically have a well-defined syntax and may be considered as a language with a grammar

$$T = \langle verb \rangle \ \langle noun \rangle^{*+}$$

- A tactical COA typically has a well-defined syntax and may be considered as a language with a well-defined grammar

$$C = \{\langle verb \rangle \ \langle attribute\ value \rangle^{*}\}^{+}$$

CAFSIN Similarity Function

- ***t*** $\in$ ***T, t* = *v n****
- ***c*** $\in$ **C, *c* = {*u m**}+**
- **The CAFSIN similarity function is defined as:**

$$h(t,c\,|\,\Psi)=\alpha\Sigma_u f(v,u\,|\,\Psi)H(n^*,m^*)$$

where *H(n*,m*)* is a fuzzified maximum substring matching function between word string *n and word string *m** using the fuzzified word similarity function *f(n,m|Ψ),* α is a normalization factor**

Computation of *H*

- **Substring matching is an NP-complete problem; an optimal solution may be obtained from dynamic programming**

- **Since *t* and *c* typically only have a very few words, complexity is not an issue**

- **Assume there are *N* words for string *n*;* there are *M* words for string *m****

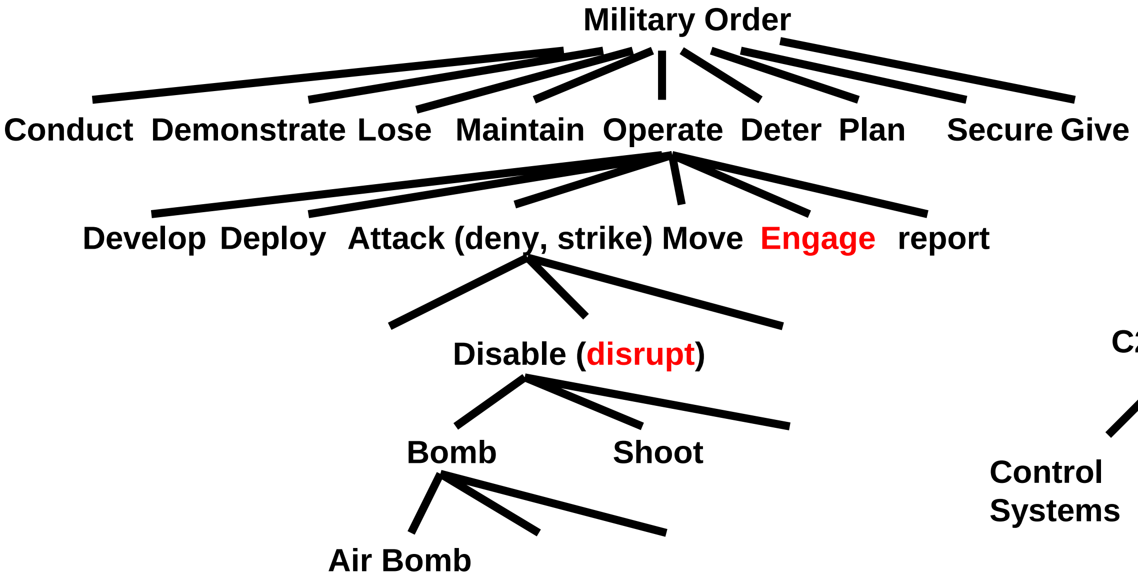
- **Create a table of *H[N+1,M+1*]; initialize the table with *H[0,j] = 0* for *j=1,…, M+1* and *H[i,0] = 0* for *i=1,…, N+1***

- ***H* is computed by:**

$$H[i,j]=\begin{cases} f(n[i],m[j]), & f(n[i],m[j])>\max(H[i-1,j],H[i,j-1]) \\ \max(H[i-1,j],H[i,j-1]), & otherwise \end{cases}$$

A Simple Example

- Commander’s intent:
  - Disrupt enemy’s WMD support system
- COA
  - Engage FA-18 on target B13



Military Order

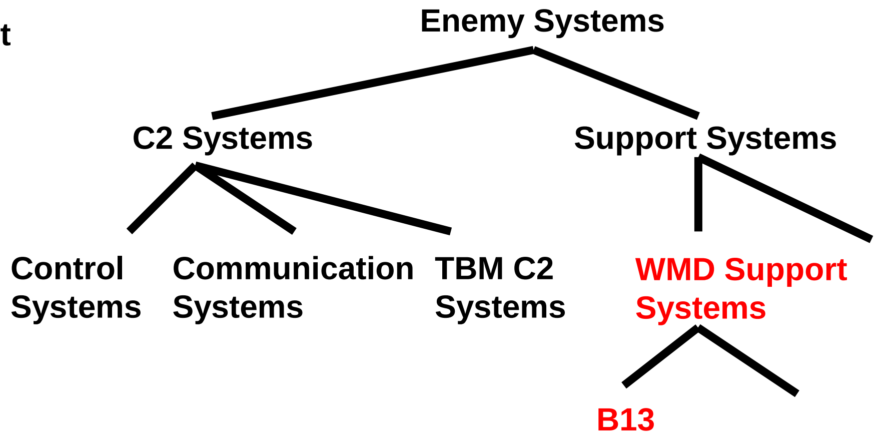
Conduct Demonstrate Lose Maintain Operate Deter Plan Secure Give

Develop Deploy Attack (deny, strike) Move Engage report

Disable (disrupt)

Bomb Shoot

Air Bomb



Enemy Systems

C2 Systems Support Systems

Control Systems Communication Systems TBM C2 Systems WMD Support Systems

B13

CAFSIN Representation

- **After text processing:**
  - ***t* = {<disrupt>,(<enemy>,<WMD support system>)}**
  - ***c* ={<engage>,(<FA-18>,<target>,<B13>)}**
- **Word pairs:**
  - **d<disrupt> = 2, d<engage> = 1**
  - **d<enemy> = ∞, d<FA-18> = ∞**
  - **d<enemy> = ∞, d<target> = ∞**
  - **d<enemy> = ∞, d<B13> = ∞**
  - **d<WMD support system> = ∞, d<FA-18> = ∞**
  - **d<WMD support system> = ∞, d<target> = ∞**
  - **d<WMD support system> = 0, d<B13> = 1**

CAFSIN Representation, Cont'd

- **Specified parameters:**
  - σ**=1,** $p=\sqrt{2\pi}$ **,** α**=1**
- **Fuzzified similarity function values:**
  - ***f*****(<disrupt>, <engage>) = 1/**$\sqrt{e}$ **=0.607**
  - ***f*****(<enemy>, <FA-18>) = 0**
  - ***f*****(<enemy>, <target>) = 0**
  - ***f*****(<enemy>, <B13>) = 0**
  - ***f*****(<WMD support system>, <FA-18>) = 0**
  - ***f*****(<WMD support system>, <target>) = 0**
  - ***f*****(<WMD support system>, <B13>) = 1**

CAFSIN Solution

- **Dynamic programming to compute *H*:**

		enemy	WMD support system
	0	0	0
FA-18	0	0	0
Target	0	0	0
B13	0	0	1

- **Final fuzzified similary between *t* and *c*:**

$$h = 1/\sqrt{e} = 0.607$$

Another Example

- **Commander's intent:**
  - **Disrupt enemy's WMD support system**
- **COA**
  - **Engage FA-18 on target B13; bomb target B13**
- **After text processing:**
  - ***t* = {<disrupt>,(<enemy>,<WMD support system>)}**
  - ***c* = {<engage>,(<FA18>,<target>,<B13>);**

    <bomb>,(<target>,<B13>)}
- **1st pair representation:**
  - ***t* = {<disrupt>,(<enemy>,<WMD support system>)}**
  - ***c* ={<engage>,(<FA-18>,<target>,<B13>)}**
- **2nd pair representation:**
  - ***t* = {<disrupt>,(<enemy>,<WMD support system>)}**
  - ***c* ={<bomb>,(<target>,<B13>)}**

CAFSIN Solution

- **1st pair fuzzified similarity between *t* and *c*:**

$$h = 1/\sqrt{e} = 0.607$$

- **2nd pair fuzzified similarity between *t* and *c*:**

$$h = 1$$

- **Final fuzzified similarity between *t* and *c*:**

$$h = 1 + 1/\sqrt{e} = 1.607$$

Another Example

- **Commander's intent:**
  - **Disrupt enemy's WMD support system**
- **COA**
  - **Lose target B13**
- **After text processing:**
  - ***t* = {<disrupt>,(<enemy>,<WMD support system>)}**
  - ***c* ={<lose>,(<target>,<B13>)}**
- **Final fuzzified similarity between *t* and *c*:**

$$h = 1/e^2 = 0.135$$

Summary

- **Presented an approach to determining COA alignment with commander's intent based upon fuzzy logic inferencing, which is:**
  - **Independent of the ontology**
  - **Independent of specific words; only dependent on the relative locations of the words in an ontology**
  - **Always relative (e.g., may be mapped to the range of [0,1]) allowing users interaction (e.g., to play with different thresholds)**
- **Reasonable assumptions must be made**
- **Presented results CAFSIN approach on a hand-crafted ontology with expected performance**

Future Work

- Large scale evaluations of CAFSIN
  - Requires a domain ontology
  - Wordnet (e.g., how to tailor it to a specific domain?)
  - How to define the evaluation metrics?
- Relax the assumptions to accommodate higher levels commander’s intent and COAs
  - Requires interface to information extraction tools
  - Relax the syntax of COA to accommodate constraints
- Improve computation complexity of CAFSIN
  - Add locality analysis to the ontology tree traversal search
  - Add heuristic search into the dynamic programming string matching