GE ADJUNCT TEST REPORT: OBJECT-ORIENTED DESIGN AND SCORING FOR MUC-4

George Krupka and Lisa Rau
Artificial Intelligence Laboratory
GE Research and Development
Schenectady, NY 12301 USA
E-mail: rau@crd.ge.com
Phone: (518) 387 - 5059

Abstract

This paper reports on the results of the adjunct test performed by GE for the MUC-4 evaluation of text processing systems. In this test, we evaluated the effect of an object-oriented template design and associated matching conditions on the scores. The results indicate that the current MUC-4 "flat" templade design with cross-references closely approximates a true object-oriented design. However the object-oriented design allows for additional performance data to be calculated, facilitating diagnosis.

INTRODUCTION

In this adjunct test, we investigate the issues and effect of transforming the MUC-4 template design automatically into an object-oriented design, with associated object-level matching conditions affecting the overall score.

An object is simply a collection of slots that all refer to one item originating in the text. This collection of slots is logically connected due to their implicit reference to one particular filler. In addition, objects may be nested so that one object contains another object, or may be recursive, with one object pointing to another and back again. Finally, there may be multiple instances of any given object.

In MUC-4 is is possible to isolate three distinct levels of objects. The first level contains a **STORY** object. Attached to a **STORY** object are **INCIDENT** objects, each of which contains participant objects; **TARGET**, **INSTRUMENT** and **PERPETRATOR**. The Figure 1 illustrates an object-oriented template design.

The MUC template design has been moving from a flat structure to a more object-oriented structure with cross-references tying together multiple slots into what has been termed a "pseudo-object". Slots tied together with cross-references are pseudo-objects and not true objects because the cross-references are not enforced. For example, consider the following scenario depicted in Figure 2. Although the human target (''**MARY**'') is wrong, the system is allowed partial credit for the other slots, even though they are clearly cross-referenced to different, and equal incorrect, fills.

MOTIVATION

There are a variety of reasons why an object-oriented design is desirable. First, an object-oriented design is conceptually easier to understand. Instead of a flat listing of all slots of a template, slots pertaining to a single fill are grouped together. Cross-references are no longer needed, as indentation indicates the grouping, so the visual design is cleaner.

Asthetics aside, there are additional performance data that can be obtained when groups of slots are connected as objects. Systems can be scored on how well a given object aligns, in a way analogous to template matching alignment scores (template ID score). Moreover, it is possible to construct an object total that does not consider whether any given object appears in a *matching* template or not. For example in our system, we found that data such as a human target was correctly extracted by the program with all its associated fields, but was put in the wrong template. This resulted in missing and spurious points for all the

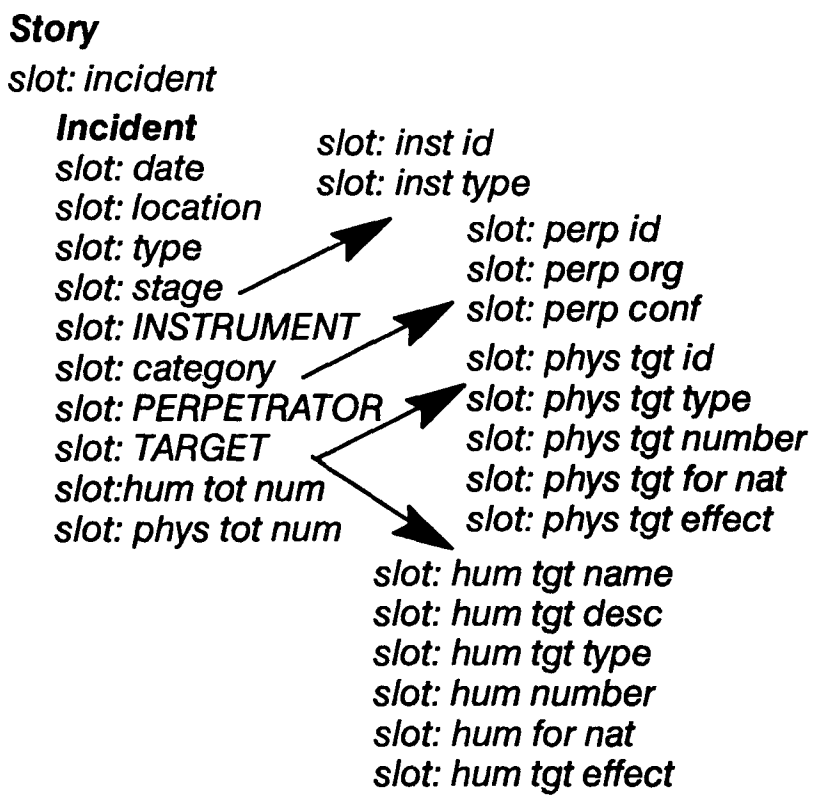

Figure 1: Object-oriented MUC-4 Template Design

fields in the object, although it could be argued that all that was incorrect was the *association* of the object with an incorrect incident. Finally, with objected-oriented totals, it is very easy to isolate performance problems down to the object-level. With a flat design, less-than-perfect templates must be examined to determine where the problems occurred. With object matching totals, it is possible to immediately isolate object-level errors which facilitates the error diagnosis process.

The rest of this paper maps out the processes used to transform the flat MUC-4 template design to an object-oriented design. We then overview the scoring experiments we performed to test the effect of various configurations. Finally, we present detailed analyses of the effect of object-oriented design and scoring on the data from MUC-4 systems performance.

```
        KEY                                    RESPONSE

HUM TGT: ID:      ''MARY''              HUM TGT: ID:      ''JOSE''
HUM TGT: DESC:    ''WIFE'': ''MARY''    HUM TGT: DESC:     ''WIFE'': ''GEORGE''
HUM TGT: TYPE     CIVILIAN: ''MARY''    HUM TGT: TYPE     CIVILIAN: ''PEDRO''
HUM TGT: EFFECT: DEATH: ''MARY''        HUM TGT: EFFECT: DEATH: ''RAUL''
```

Figure 2: Example of Unenforced Cross-references

TRANSFORMATION TO O-O DESIGN

The first step in performing this test was to automatically transform the existing template design to the object-oriented design illustrated in Figure 1. This process was aided by the existing cross-references, but was complicated by a variety of special cases we encountered. These special cases fell into two general categories; system glitches, and problems with the MUC-4 template format. The first three problems described below are system glitches; the last three are issues in the design of the templates.

1. **Inconsistent cross-references:** We encountered inconsistent cross-reference strings. For example, ''PEOPLE'' may have been present as a human target description, but a slot intended to cross-reference to it may have read ''TWO PEOPLE''.

2. **Violation of Template Filling Rules:** Some sites cross-referenced the perpetrator confidence to a null value, or to the PERP ID slot for example.

3. **Multiple Set Fills:** We encountered fills such as NO INJURY NO DEATH INJURY DEATH.

4. **Inconsistent treatments:** The treatment of repeated fills, as could be required in sentences such as ''KILLED 3 PEOPLE AND INJURED 2'' was handled in different ways, with PEOPLE repeated as a fill, or with two EFFECTs cross-referenced to one fill.

5. **Ambiguity of ''-'':** When an EFFECT has a blank value, its scoping is ambiguous. When we attempt to group targets into objects, we cannot decide which object this effect belongs to.

6. **Ambiguity of Optional Fills:** It is impossible with the current template design to determine when an optional fill of an optional object was meant, as opposed to a required fill of an optional object.

For the system problems, we manually intervened and allowed the conversion process to proceed. However one sites' responses were too unusual too allow us to transform the output without a great deal of manual interaction. For the template design problems, we came up with adequate methods of working around the problems.

OBJECT-ORIENTED SCORING

After all (execept one) of the sites' answers were transformed into the object-oriented design, a modified version of the MUC-4 scoring program was run in a variety of configurations to test the effect of enforcing object-level matching. This program used the merged history file, and took 10 seconds to score an average run. It took one person-week to convert all the sites' answer templates to the object oriented design, and create a new version of the scoring program to use this design.

We experimented with a variety of conditions for aligning templates and objects. These were:

1. Only incident type match. This was closer to the MUC-2 scoring conditions.

2. Must match on incident type, plus either a match on ID or type for target or ID or ORG for perpetrator. This duplicated the MUC-4 scoring, in that either a match on target or perpetrator would cause the incident to align.

3. Must match on incident type plus a match on the string ID slot of a target. With this condition, we only aligned templates if the targets aligned according to a stricter matching condition. This matching condition required at least a partial match on the target string. Note that virtually no templates have no targets.

4. Free-floating object match. For this design, we computed the score if objects were allowed to match each other without considering if they happen to belong to an aligning template object. That is, if a system mistyped an ATTACK as an ARSON but correctly extracted any human or physical targets, credit would be given for these objects.

```
        KEY 1                                   RESPONSE 1

HUM TGT: ID:      ''MARY''              HUM TGT: ID:      -
HUM TGT: DESC:    ''NUN'': ''MARY''     HUM TGT: DESC:     ''NUN''
HUM TGT: TYPE:    CIVILIAN: ''MARY''    HUM TGT: TYPE     CIVILIAN: ''NUN''
HUM TGT: NUM:     1:''MARY''            HUM TGT: NUM:     1: ''NUN''
HUM TGT: EFFECT: DEATH: ''MARY''        HUM TGT: EFFECT: DEATH: ''NUN''

        KEY 2                                   RESPONSE 2

HUM TGT: ID:      ''MARY''              HUM TGT: ID:      ''JOHN''
HUM TGT: DESC:    ''NUN'': ''MARY''     HUM TGT: DESC:     ''NUN'':''JOHN''''
HUM TGT: TYPE:    CIVILIAN: ''MARY''    HUM TGT: TYPE     CIVILIAN: ''JOHN''
HUM TGT: NUM:     1:''MARY''            HUM TGT: NUM:     1: ''JOHN''
HUM TGT: EFFECT: DEATH: ''MARY''        HUM TGT: EFFECT: DEATH: ''JOHN''

        KEY 3                                   RESPONSE 3

HUM TGT: DESC:    ''PEOPLE''            HUM TGT: DESC:    ''PEOPLE''
HUM TGT: EFFECT: INJURY: ''PEOPLE''     HUM TGT: EFFECT: INJURY
```

Figure 3: Differences between MUC-4 score and OO MUC-4 score

RESULTS

There are a few differences between our score computation and that used by the official MUC-4 scoring program. These differences are a direct result of object-oriented alignment. These situations appear in Figure 3.

In all three of these cases, we assign full credit for all the cross-referenced fields, whereas the MUC-4 scoring program would assign partial credit, due to the incorrect cross-reference. We assign full credit for these fields because the objects align (under the duplicated MUC-4 condition) due to the match of the **DESC** field. When objects align, the cross-reference is no longer considered when scoring slots within the object. These differences in credit assignment account for the small increases in scores in between the official score and our object-oriented duplication (see Figure 4).

As expected, the first condition of incident-match only improved every sites' scores. This is because sites are now getting credit for those slots that match even when the events in the key clearly do not match the events in the answers. This is due to the assignment of partial credit as exemplified in Figure 2.

Figure 4 gives the difference between the official MUC-4 score, the duplicated condition, and enforcing string-primacy for object alignment. As can be seen from this table, the duplicated condition causes small increases in every sites' scores. However, enforcing the cross-references by requiring either a partial or full match on the object string (if present) did not significantly change the MUC-4 scores. Finally for "free-floating" object matches, there was across-the-board, although small, improvement in the object scores, and in all the total system scores. The improvement in overall score would be greater than these figures suggest, because each slot of each object that was in the wrong template is counted as both missing (in the correct template) and spurious (in the incorrect template). We believe a more accurate measure of performance would be to give credit for the correct gross recall, with a suitable penalty for the incorrect placement of the object in the template.

Figure 5 illustrates the total object scores (target, perpetrator and instrument) for each site and the delta from the original. Finally, we believe that object-level matching allows for more in-depth diagnosis and evaluation of system performance. Object-alignment totals, and "matched-only" object total increase the amount of information useful for zeroing in on problem areas. This additional information is described

SITE	MUC-4 condition Delta from score	String Primacy Delta from score
BBN	4.80	1.03
GE	1.53	0.0
GE-CMU	1.50	-1.01
LSI	2.72	0.31
MDC	5.51	0.0
NMSU	5.02	1.01
NYU	1.43	-2.01
PARAMAX	1.33	-.01
PRC	1.68	-2.08
SRA	1.82	-1.10
SRI	2.02	0.0
UMASS	1.41	-2.43
UMICH	3.00	-1.52

Figure 4: Effect of OO-Alignment and String Primacy on Scores

SITE	Original Object Tot	Free-Floating Tot	Delta
BBN	30.00	31.77	1.77
GE	51.46	55.00	3.54
GE-CMU	45.90	49.50	3.70
LSI	16.61	20.87	4.26
MDC	18.44	20.62	2.18
NMSU	15.44	18.16	2.72
NYU	36.93	41.18	4.22
PARAMAX	23.09	27.77	4.68
PRC	23.29	26.89	3.60
SRA	22.96	28.13	5.17
SRI	42.53	47.68	5.15
UMASS	43.03	47.34	4.31
UMICH	33.00	39.78	6.78

Figure 5: Effect of Free-Floating Object Scores

META SLOT	POS	ACT	COR	PAR	INC	MIS	SPU	REC	PRE	OVG	PR	2PR	P2R
instruments	52	53	34	0	0	18	19	65	64	36	64.50	64.20	64.80
perpetrators	137	118	62	0	0	75	56	45	53	47	48.67	51.18	46.40
targets	216	269	139	0	0	77	130	64	52	48	57.38	54.03	61.18
incidents	114	12	88	0	0	26	34	77	72	28	74.42	72.95	75.95
TOTAL	519	562	323	0	0	196	239	62	57	43	59.39	57.93	60.93

OBJECT TOT	POS	ACT	COR	PAR	INC	MIS	SPU	REC	PRE	OVG	PR	2PR	P2R
PHYS-TGT	247	269	101	4	11	131	153	42	38	57	39.90	38.74	41.13
HUM-TGT	609	679	353	19	26	211	281	60	53	41	56.28	54.27	58.46
INSTRUMENT	85	89	52	7	0	26	30	65	62	34	63.46	62.58	64.38
PERP-ORG	103	85	36	1	8	58	40	35	43	47	38.59	41.12	36.35
PERP-INDIV	85	75	34	5	0	46	36	43	49	48	45.80	47.67	44.08
INCIDENT	521	558	324	47	23	127	164	67	62	29	64.40	62.94	65.94
TOTAL	1650	1755	900	83	68	599	704	57	54	40	55.46	54.57	56.37

FF-OBJECT TOT	POS	ACT	COR	PAR	INC	MIS	SPU	REC	PRE	OVG	PR	2PR	P2R
PHYS-TGT	276	269	129	4	15	128	121	47	49	45	47.98	48.59	47.39
HUM-TGT	638	679	381	22	29	206	247	61	58	36	59.46	58.58	60.38
INSTRUMENT87	89	54	8	0	25	27	67	65	30	65.98	65.39	66.59	
PERP-ORG	117	85	40	1	10	66	34	35	48	40	40.48	44.68	37.00
PERP-INDIV	93	75	39	6	0	48	30	45	56	40	49.90	53.39	46.84
INCIDENT	535	558	354	46	21	114	137	70	68	25	68.99	68.39	69.59
TOTAL	1746	1755	997	87	75	587	596	60	59	34	59.50	59.20	59.80

PSUEDO TOT	POS	ACT	COR	PAR	INC	MIS	SPU	REC	PRE	OVG	PR	2PR	P2R
inc-total	539	574	339	57	21	0	20	157	122	209	68	64	27
perp-total	258	233	113	7	22	5	6	91	116	271	45	50	39
phys-tgt-total	249	269	95	15	23	5	10	136	116	628	41	38	50
hum-tgt-total	615	693	342	64	34	18	55	253	175	516	61	54	36

Figure 6: Sample Object Totals for GE System

in the next section.

DISCUSSION

One of the advantages of object-oriented scoring is that it is possible to obtain object alignment totals, and object matching totals. Figure 6 illustrates this type of data for our system on MUC-3 (a description of our system and a summary of our performance can be found in this volume). The **META SLOT** table contains a measurement of how well our system aligned objects.

The **OBJECT-TOT** table is useful to compare the totals for object matches when objects that appear in different templates are not scored, with the **FF-OBJECT TOT** table which presents the "free-floating" object totals, allowing for matches between unaligned objects that appear in incorrect templates to contribute to recall and precision. The **PSEUDO TOT** table gives the pseudo-object numbers presented at the bottom of our score report for comparison.

FUTURE WORK

One benefit of object-oriented design is that there can be only one representation of each unique object. This representation can be pointed to by a variety of slots. This allows for credit to be assigned once for each matching object, with separate credit assigned for attaching the object correctly in whichever relationships

it participates in.

Primarly to ensure that the total number of points in this adjunct test was comparable to the the total number of points in the official scores, we did not make objects unique. However we believe that assigning credit for extracting information from an object once would increase the accuracy of the evaluation.

SUMMARY

This paper has reported on an adjunct test performed in connection with MUC-4 to investigate the utility and issues involved in object-oriented template design and scoring. We have shown that an object-oriented design, even when modified to enforce partial string matching as the criterion for object alignment, does not significantly alter the MUC-4 scores. Object-oriented design is a more intuitive method of representing information that is related. Moreover, objects can be aligned, allowing for object-level scoring. This increases the usefulness of an automated scoring program to perform selective diagnosis for performance evaluation. Also, object-oriented alignment allows for the scoring of objects that match even when they are placed in an incorrect template. This yields a more accurate evaluation of performance than scoring all the slots of a misplaced object as missing and spurious.

PART II: TEST RESULTS AND ANALYSIS (SITE REPORTS)

The papers in this section were prepared by each of the sites that completed the MUC-4 evaluation. The papers are intended to provide the reader with some context for interpreting the test results, which are presented more fully in appendices G and H of the proceedings. The sites were asked to comment on the following aspects of their MUC-4 experience:

* Explanation of test settings (precision/recall/overgeneration) and how these settings were chosen

* Where bulk of effort was spent, and how much time was spent overall on MUC-4

* What the limiting factor was (time, people, CPU cycles, knowledge, ...)

* How the training of the system was done
  - What proportion of the training data was used (and how)
  - Whether/Why/How the system improved over time, and how much of the training was automated

* What was successful and what wasn't, and what system module you would most like to rewrite

* What portion of the system is reusable on a different application

* What was learned about the system, about a MUC-like task, about evaluation