

84-11

**A Comparison of Performances
of a SUN-2 workstation, VAX 11/780, and the Xerox Dolphin
on a Learning Algorithm**

Anthony R. Nowicki

**Department of Computer Science
University of Illinois
at Urbana-Champaign**

December 1984

ISG 84-9

This work was supported in part by the National Science Foundation under grant NSF DCR 84-06801 and the Office of Naval Research under grant N00014-82-K-0186.

Acknowledgements

I wish to thank my advisor Professor R. S. Michalski for encouragement and suggestions during the preparation of this report and thanks to Jeff Becker for his aid during the in writing of the Franz Lisp version of the Aq algorithm and for obtaining the benchmarks for the INTERLISP implementation.

Table of Contents

Abstract	1
1. Introduction	1
2. Program Description	1
3. Data Description	1
4. Benchmarks	2
4.1. Aq Algorithm Benchmarks	2
4.2. List Reversal Benchmarks	4
5. Conclusions	5
References	6
Appendix A: Program Listings	7
Appendix B: Data Sets	8

A Comparison of Performances of a SUN-2 workstation, VAX 11/780, and the Xerox Dolphin on a Learning Algorithm

Abstract

This report presents data representing the execution times of a learning algorithm and a list reversing program. Benchmarks for LISP, Pascal, and Prolog versions of various programs are run on the SUN-2 workstation, the VAX 11/780, and the Xerox Dolphin.

1. Introduction

Given that program execution times vary from machine to machine and from programming language to programming language, it is interesting to compare program execution times on a number of machines available here at the University of Illinois and the Artificial Intelligence Laboratory. Two programs were chosen for this comparison. The first is an inductive learning program based on the Aq algorithm, the second, a list reversing program. The Aq algorithm is implemented in a number of different languages. For this experiment, the Pascal version GEM [12], and the LISP version AQII, are used. AQII is implemented in both Franz Lisp [2] and INTERLISP [1]. The list reversing program is implemented in Franz Lisp and UNSW Prolog [13].

The Franz Lisp programs run on the University of Illinois, Department of Computer Science VAX 11/780, and on the Artificial Intelligence Laboratory Sun-2 workstations. The INTERLISP programs run on the VAX 11/780 and the XEROX 1100. The Pascal program GEM runs on the VAX 11/780 and the Sun-2 workstation.

2. Program Description

AQII/Franz is an interactive program for generalization and optimization of discriminant descriptions of object classes. The descriptions are expressed in disjunctive normal expressions in the variable valued logic system VL₁ [8]. The program takes as input sets of example objects belonging to different classes and provides a description of each class which covers the examples of that class. For a more detailed description of the Aq algorithm the reader is referred to [7, 9].

The list reversal benchmarks are obtained and added to a benchmarks table obtained from H. Nakashima and S. Tomaura of the Electric Technical Laboratory of the Scuba Science City, Japan [10]. The program also appears in the July 1984 issue of *Byte* magazine [5]. Two functions are defined in the program (see Appendix A). REV is a slow version of the Lisp REVERSE function, and APP is a slow version of the APPEND function.

3. Data Description

The Aq programs were run on two different data sets (see Appendix B). The first, DATAX, is a small data set consisting of three classes each having three events of four variables each. This data set contains variables of type linear, nominal, and structured. The second data set, PLANT, is a subset of the PLANT data set used to create the rules for the expert system PLANT/ds [11]. This data set contains 50 linear variables and 17 classes, each class being composed of seven events.

These data sets were run in three different modes; VL (sequential) mode, DC, disjoint cover mode, and IC, intersecting cover mode. All examples runs were made with a maxstar of 5 (see [3, 7] for a description of maxstar and covering modes).

4. Benchmarks

The execution times presented in the following tables represent the CPU and real time in which the programs execute. Times are presented in minutes, seconds, and tenths of seconds.

4.1. Aq Algorithm Benchmarks

Table 1 shows the execution times for the program AQII/Franz on the data set DATAX. The first run of the interpreted version was not taken into account due to the extra time required for initial macro expansion. The macros remain expanded on the Sun-2 Franz version, however, the macros appear to be reexpanded at each run of the VAX Franz program. Table 2 represents times obtained for the same programs run on the PLANT data set. Only one mode of operation was tested due to the long execution time (≈ 45 mins.). The mode DC was chosen so that later comparisons can be made with the GEM program.

AQII/Franz execution times								
Mode	SUN version				VAX version			
	Interpreted		Compiled		Interpreted		Compiled	
	CPU	real	CPU	real	CPU	real	CPU	real
VL	00:09.97	00:11.15	00:01.30	00:01.30	00:41.91	00:82.00	00:01.09	00:03.20
DC	00:15.00	00:15.30	00:01.95	00:02.16	00:62.93	02:20.00	00:01.66	00:05.00
IC	00:26.89	00:27.23	00:03.59	00:03.87	01:45.15	03:28.67	00:03.01	00:10.30

Table 1. Execution times for AQII/Franz on DATAX.

AQII/Franz (compiled) execution times				
Mode	SUN version		VAX version	
	CPU	real	CPU	real
DC	41:16.15	43:46.00	32:47.48	46:54.00

Table 2. Execution times for AQII/Franz on PLANT.

Table 3 displays data for the same program written in INTERLISP. The program was run on the VAX 11/780 and the Xerox 1100. Interpreted times were not obtained for the VAX version.

AQII/INTERLISP execution times (seconds)						
Mode	Dolphin version				VAX version	
	Interpreted		Compiled		Compiled	
	CPU	real	CPU	real	CPU	real
VL	32.56	32.56	00.74	00.74	00.50	01.17
DC	48.51	48.52	01.09	01.09	00.70	02.58
IC	87.86	87.86	02.02	02.16	01.29	05.97

Table 3. Execution times for AQII/INTERLISP on DATAX.

We have seen some execution times for Lisp implementations of the Aq algorithm. It is interesting to compare these times with a Pascal implementation. The Pascal program, like the LISP version, was run on the two data sets DATAX and PLANT. Tables 4 and 5 present the results of the runs on the respective data sets.

GEM execution times (seconds)				
Mode	SUN version		VAX version	
	System	User	System	User
DC	00.173	00.646	00.140	00.457
IC	00.177	00.950	00.177	00.600

Table 4. Execution times for GEM on DATAX.

GEM execution times				
Mode	SUN version		VAX version	
	System	User	System	User
DC	00:43.80	15:05.60	00:45.47	11:34.83

Table 5. Execution times for GEM on PLANT.

4.2. List Reversal Benchmarks

The list reversing programs in Appendix A are run on a list having six elements. Table 6 presents the times obtained for the sample runs. The Sun-2 entries are added to an existent table as mentioned in section 2.

Execution times of list reversal programs (seconds)			
Machine	Language	Interpreter	Compiler
Xerox 1100	Interlisp-D	16.0	0.3
Symbolics 3600	ZetaLisp	7.0	0.2
	Prolog/KR	117.0	
LMI Series III	ZetaLisp	16.0	0.2
DEC2060	MacLisp	1.0	0.06
	Interlisp	1.8	0.3
	Prolog/KR	8.0	
	Prolog-10	2.8	0.25
	Concurrent-Pro	202.0	
M200F	UtiLisp	0.08	0.01
	Prolog/KR	0.75	
VAX-11/750	Unix/FranzLisp	4.0	0.8
VAX-11/780	Unix/FranzLisp	2.6	0.5
	VMS/C-Prolog	2.5	
Sun-2	Franz Lisp	4.5	0.8
	UNSW Prolog	2.5	

Table 6. List reversal times for various machines (all times in seconds).

The UNSW Prolog implementation required a stack space of 15000 in order to reverse a six element list. The Sun-2 Franz Lisp program appears to be twice as slow as the VAX 11/780 program but it should be pointed out that the times presented in table 6 were obtained from the Electrical Technical Laboratory and some sample runs of the reversal program on this Department's VAX 11/780 gives an average time of ≈ 4 secs, quite different than that which is reported in the table. The discrepancy between the times can not be determined due to the limited information provided with the table.

5. Conclusions

Having seen some of the execution times on both large and small data sets, some judgements can be made concerning the tested systems. Let us first consider the Lisp environment.

The Franz Lisp programs appear to run at comparable speeds on the Sun-2 and the VAX 11/780. For interpreted code, the Sun-2 program requires less CPU time and less real time. The compiled programs are nearly the same as the VAX with the VAX running faster by only a matter of tenths of seconds. It was noted in section 4.1 that the macro expansion takes place at each execution of the interpreted code, this might explain the slower times for the VAX 11/780 implementation. In all cases, the 'real' time required to process the data set was less on the Sun-2 than the VAX 11/780 by a factor of about 10:1. The programs were run on the VAX 11/780 when a load factor of about 4 was reported. With the Department VAX averaging daily loads of greater than 4, this makes the Sun-2 workstation even more appealing as a program development tool. CPU time for the large data set was less on the VAX 11/780 than the Sun-2 by a factor of 1.25:1. As for the smaller data set, the 'real' time was less on the Sun-2.

If we compare the Sun-2 and the Dolphin, we see that interpreted Franz Lisp on the Sun-2 is 3.5 times faster than interpreted INTERLISP on the Xerox Dolphin. With the Sun 'real' time only slightly slower than its CPU time, the Sun-2 executes in less 'real' time than the Dolphin. However, compiled INTERLISP on the Dolphin is twice as fast as compiled Franz Lisp on the Sun-2.

As a final comparison, the Pascal program was run on the same two data sets. The Pascal program was roughly 10 times faster than the Franz Lisp version on the small data set. For the large data set, the Pascal program was 56 times faster. Both measurements reflect the times for the Sun-2 workstation.

Table 7 presents the summary of the measured times that were obtained for the various machines and data sets.

The data obtained from the list reversal programs confirm the above results but it is difficult to compare the Sun-2 with other machines in table 6 due to the discrepancy noted in section 4.2. An interesting point is that the UNSW Prolog implementation of the reversal ran almost twice as fast as the Franz Lisp implementation.

Systems	language	Interpreted Compiled	data set	Performance RATIO
SUN vs. VAX	FranzLisp	I	DATA	0.25:1
SUN vs. VAX	FranzLisp	C	DATA	1.2:1
SUN vs. VAX	FranzLisp	C	PLANT	1.25:1
SUN vs. VAX	Pascal	C	DATA	1.1:1
SUN vs. VAX	Pascal	C	PLANT	0.9:1
DOLPHIN vs. VAX	INTERLISP	C	DATA	1.5:1

Table 7. Benchmark summary table.

References

- (1) — , Interlisp Reference Manual, Warren Teitelman, ed., Xerox Palo Alto Research Center, California, 1978.
- (2) — , The Franz Lisp Manual, John K. Foderaro, Keith L. Sklower, Kevin Layer, eds., University of California, June 1983.
- (3) Becker, J. M., "AQINTERLISP: An INTERLISP Program for Inductive Generalization of VLI Event Sets," UIUCDCS-F-83-911, ISG 83-11, Department of Computer Science, University of Illinois, Urbana, September 1983.
- (4) Becker, J. M., "AQ-PROLOG: A Prolog Implementation of an Attribute-Based Inductive Learning System," Unpublished paper, Department of Computer Science, University of Illinois, December 1983.
- (5) Bortz, J. and Diamant, J., "Benchmarking IQLISP and muLISP", *Byte Magazine*, pp. 288-291, July 1984.
- (6) Falkenhainer, B., "ABACUS: Adding Domain Constraints to Quantitative Scientific Discovery," UIUCDCS-F-84-1195, ISG 84-7, Department of Computer Science, University of Illinois, May 1984.
- (7) Michalski, R. S. and Larson, J. B., "Selection of Most Representative Training Examples and Incremental Generation of VLI Hypotheses: the Underlying Methodology and Description of Programs ESEL and AQ11", Report No. 867, Department of Computer Science, University of Illinois, 1978.
- (8) Michalski, R. S., "Variable-Valued Logic: System VLI", *1974 International Symposium on Multiple-Valued Logic*, West Virginia University, Morgantown, West Virginia, May 1974.
- (9) Michalski, R. S., "A Comparative Review of Selected Methods for Learning from Examples.", In *Machine Learning - An Artificial Intelligence Approach*, R. S. Michalski, J. G. Carbonell, and T. M. Mitchell (Eds.), Tioga Pub., 1983.
- (10) Nakashima, H., and Tomura, S., "Benchmark Test: Speed of Executing a Slow Reverse", The Electrical Technical Laboratory, Scuba Science City, China, October 1983.
- (11) Reinke, R. E., "PLANT/ds: An Expert System for Diagnosing Soybean Diseases Common in Illinois. User's Guide and Program Description," UIUCDCS-F-83-911, Department of Computer Science, University of Illinois, 1983.
- (12) Reinke, R. E., "Knowledge Acquisition and Refinement Tools for the ADVISE META-EXPERT System," M.S. Thesis, UIUCDCS-F-84-921, ISG 84-4, Department of Computer Science, University of Illinois, July 1984.
- (13) Sammut, C. A., and Sammut, R. A., "The Implementation of UNSW-PROLOG," *The Australian Computer Journal*, Vol. 15, No. 2, May 1983.

Appendix A (Program Listings)

Franz Lisp implementation of the list reversal program.

```

-----
; reverse a list

(defun rev (x)
  (cond ((null x) nil)
        (t (app (rev (cdr x)) (ncons (car x))))))

; modified append

(defun app (x y)
  (cond ((null x) y)
        (t (app (rev (cdr (rev x)))
                  (cons (car (rev x)) y )))))

```

UNSW Prolog implementation of the list reversal program.

```

%-----
% reverse a list

rev([], []).
rev([A|X], Z) :- rev(X, Y),
                 app(Y, [A], Z).

% modified append

app([], X, X).
app(X, Y, Z) :- rev(X, [A|RX]),
                 rev(X, Dummy), % for fairness to lisp
                 rev(RX, RRX),
                 app(RRX, [A|Y], Z).

```

Appendix B Data Sets

The DATAX data set

```
parameters
echo    maxstar    trim    mode
pv      5          yes     dc
```

```
domaintypes
name    levels    type
fam     5         str
nom1    3         nom
lin1    4         lin
```

```
variables
# name
1 x1.nom1
2 x2.lin1
3 x3.nom1
4 x4.fam
```

```
fam-structure
value    subvalue    subvalue
5         1           2
6         3           4
```

```
typeA-events
x1    x2    x3    x4
1      1      1      1
1      3      1      3
2      2      2      3
```

```
typeB-events
x1    x2    x3    x4
1      2      1      2
1      2      2      1
1      1      2      2
```

```
typeC-events
x1    x2    x3    x4
2      1      1      2
2      3      1      2
2      3      2      2
```

The PLANT data set.

The parameter and variable declarations appear below:

```

parameters
echo  maxstar  trim  mode
pv    10       yes   dc

variables
name  type  levels      name  type  levels
x1    lin  11           x26   lin  11
x2    lin  11           x27   lin  11
x3    lin  11           x28   lin  11
x4    lin  11           x29   lin  11
x5    lin  11           x30   lin  11
x6    lin  11           x31   lin  11
x7    lin  11           x32   lin  11
x8    lin  11           x33   lin  11
x9    lin  11           x34   lin  11
x10   lin  11           x35   lin  11
x11   lin  11           x36   lin  11
x12   lin  11           x37   lin  11
x13   lin  11           x38   lin  11
x14   lin  11           x39   lin  11
x15   lin  11           x40   lin  11
x16   lin  11           x41   lin  11
x17   lin  11           x42   lin  11
x18   lin  11           x43   lin  11
x19   lin  11           x44   lin  11
x20   lin  11           x45   lin  11
x21   lin  11           x46   lin  11
x22   lin  11           x47   lin  11
x23   lin  11           x48   lin  11
x24   lin  11           x49   lin  11
x25   lin  11           x50   lin  11

```

The attributes for the variables defined above are explained in the following table:

x1) time_of_occurrence	x26) condition_of_stem
x2) precipitation	x27) stem_lodging
x3) temperature	x28) stem_cankers
x4) cropping_history	x29) canker_lesion_color
x5) damaged_area	x30) reddish_canker_margin
x6) severity	x31) fruiting_bodies_on_stem
x7) plant_height	x32) external_decay_of_stem
x8) condition_of_leaves	x33) mycelium_on_stem
x9) leaf_spots	x34) external_stem_discoloration
x10) leaf_spot_color	x35) location_of_stem_discoloration
x11) color_of_spot_on_reverse_side	x36) internal_discoloration_of_stem
x12) yellow_leaf_spot_halos	x37) sclerotia_internal_or_external
x13) leaf_spot_margins	x38) condition_of_fruit_pods
x14) raised_leaf_spots	x39) fruit_pods
x15) leaf_spot_growth	x40) fruit_spots
x16) leaf_spot_size	x41) condition_of_seed
x17) shot_holing	x42) seed_mold_growth
x18) shredding	x43) seed_discoloration
x19) leaf_malformation	x44) seed_discoloration_color
x20) premature_defoliation	x45) seed_size
x21) leaf_mildew_growth	x46) seed_shriveling
x22) leaf_discoloration	x47) condition_of_roots
x23) position_of_affected_leaves	x48) root_rot
x24) condition_of_leaves_below_affected_leaves	x49) root_galls_or_cysts
x25) leaf_withering_and_wilting	x50) root_sclerotia

The sample events for the PLANT data set are:

Alternaria-events

x1	x2	x3	x4	x5	x6	x7	x8	x9	x10	x11	x12	x13	x14	x15	x16	x17	x18	x19	x20
6	2	2	3	1	0	1	0	2	6	1	1	1	1	2	3	2	1	1	2
5	2	2	3	0	1	1	0	2	6	1	1	1	1	2	3	2	1	1	2
6	2	2	3	1	1	1	0	2	6	1	1	1	1	2	3	2	1	1	2
5	2	2	3	0	0	1	0	2	6	1	1	1	1	2	3	2	1	1	2
5	2	2	3	1	0	1	0	2	6	1	1	1	1	2	3	2	1	1	2
6	2	2	3	0	1	1	0	2	6	1	1	1	1	5	3	2	1	1	2
5	2	2	1	3	0	1	0	2	6	1	1	1	1	5	3	2	1	1	2
x21	x22	x23	x24	x25	x26	x27	x28	x29	x30	x31	x32	x33	x34	x35	x36	x37	x38	x39	x40
1	1	3	1	1	0	0	0	0	0	0	0	0	0	0	0	0	1	1	2
1	1	3	1	1	0	0	0	0	0	0	0	0	0	0	0	0	1	1	2
1	1	3	1	1	0	0	0	0	0	0	0	0	0	0	0	0	1	1	2
1	1	3	1	1	0	0	0	0	0	0	0	0	0	0	0	0	1	1	2
1	1	3	1	1	0	0	0	0	0	0	0	0	0	0	0	0	1	1	2
1	1	3	1	1	0	0	0	0	0	0	0	0	0	0	0	0	1	1	2
1	1	3	1	1	0	0	0	0	0	0	0	0	0	0	0	0	1	1	2
x41	x42	x43	x44	x45	x46	x47	x48	x49	x50										
1	1	2	4	1	1	0	0	0	0										
1	1	2	4	1	1	0	0	0	0										
1	1	2	4	1	1	0	0	0	0										
1	1	2	3	1	1	0	0	0	0										
0	0	0	0	0	0	0	0	0	0										
0	0	0	0	0	0	0	0	0	0										
0	0	0	0	0	0	0	0	0	0										

Anthracnose-events

x1	x2	x3	x4	x5	x6	x7	x8	x9	x10	x11	x12	x13	x14	x15	x16	x17	x18	x19	x20
6	2	1	3	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0
5	2	1	1	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0
5	2	1	3	1	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0
5	2	1	1	0	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0
5	2	1	3	2	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0
5	2	1	3	1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0
5	2	1	1	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0
x21	x22	x23	x24	x25	x26	x27	x28	x29	x30	x31	x32	x33	x34	x35	x36	x37	x38	x39	x40
0	0	0	0	0	1	1	5	4	1	2	1	2	1	0	1	1	1	1	4
0	0	0	0	0	1	1	5	4	1	2	1	2	1	0	1	1	1	1	4
0	0	0	0	0	1	1	5	4	1	2	1	1	1	0	1	1	1	1	2
0	0	0	0	0	1	1	5	4	1	1	1	1	1	0	1	1	1	2	4
0	0	0	0	0	1	1	4	4	1	2	1	1	1	0	1	1	1	2	4
0	0	0	0	0	1	1	4	4	1	1	1	1	1	0	1	1	1	2	4
0	0	0	0	0	1	1	4	4	1	2	1	1	1	0	1	1	1	1	4
x41	x42	x43	x44	x45	x46	x47	x48	x49	x50										
1	2	1	1	2	2	0	0	0	0										
0	0	0	0	0	0	0	0	0	0										
0	0	0	0	0	0	0	0	0	0										
0	0	0	0	0	0	0	0	0	0										
0	0	0	0	0	0	0	0	0	0										
1	2	2	4	2	2	0	0	0	0										
1	2	2	4	2	2	0	0	0	0										

Bact_Blight-events

[illegible]

Bact_Pustule-events

[illegible]

Rhizoctonia-events

x1	x2	x3	x4	x5	x6	x7	x8	x9	x10	x11	x12	x13	x14	x15	x16	x17	x18	x19	x20
1	2	0	3	0	2	0	1	0	0	0	0	0	0	0	0	0	0	0	0
1	1	1	1	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0
1	1	1	3	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0
2	2	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
2	1	1	3	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0
2	2	0	1	0	2	0	1	0	0	0	0	0	0	0	0	0	0	0	0
1	2	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
x21	x22	x23	x24	x25	x26	x27	x28	x29	x30	x31	x32	x33	x34	x35	x36	x37	x38	x39	x40
0	0	0	0	0	1	2	3	4	1	1	2	2	3	1	1	1	0	0	0
0	0	0	0	0	1	1	3	4	1	1	2	1	3	1	1	1	0	0	0
0	0	0	0	0	1	2	3	4	1	1	2	1	3	1	1	1	0	0	0
0	0	0	0	0	1	1	3	4	1	1	2	1	3	1	1	1	0	0	0
0	0	0	0	0	1	2	3	4	1	1	2	1	3	1	1	1	0	0	0
0	0	0	0	0	1	1	2	4	1	1	2	1	3	1	1	1	0	0	0
0	0	0	0	0	1	1	3	4	1	1	2	1	3	1	1	1	0	0	0
x41	x42	x43	x44	x45	x46	x47	x48	x49	x50										
0	0	0	0	0	0	1	2	1	1										
0	0	0	0	0	0	1	2	1	1										
0	0	0	0	0	0	1	2	1	1										
0	0	0	0	0	0	1	2	1	1										
0	0	0	0	0	0	1	2	1	1										
0	0	0	0	0	0	0	0	0	0										
0	0	0	0	0	0	1	2	1	1										

BIBLIOGRAPHIC DATA SHEET	1. Report No. UIUCDCS-F-84-929	2.	3. Recipient's Accession No.
4. Title and Subtitle A Comparison of Performances of a SUN-2 Workstation, VAX 11/780, and the Xerox Dolphin on a Learning Algorithm			5. Report Date December 1984
			6.
7. Author(s) Anthony R. Nowicki			8. Performing Organization Rept. No.
9. Performing Organization Name and Address Department of Computer Science University of Illinois Urbana, IL			10. Project/Task/Work Unit No.
			11. Contract/Grant No. NSF DCR 84-06801 ONR N00014-82-K-0186
12. Sponsoring Organization Name and Address National Science Foundation Office of Naval Research Washington, DC Arlington, VA			13. Type of Report & Period Covered
			14.
15. Supplementary Notes			
16. Abstracts This report presents data representing the execution times of a learning algorithm and a list reversing program. Benchmarks for LISP, Pascal and Prolog versions of various programs are run on the SUN-2 Workstation, the VAX 11/780 and the Xerox Dolphin.			
17. Key Words and Document Analysis. 17a. Descriptors Benchmarks LISP Pascal Prolog SUN-2 Workstation VAX 11/780 Xerox 1100			
17b. Identifiers/Open-Ended Terms			
17c. COSATI Field/Group c			
18. Availability Statement		19. Security Class (This Report) UNCLASSIFIED	21. No. of Pages 21
		20. Security Class (This Page) UNCLASSIFIED	22. Price

