[*Proceedings of the National Symposium on Machine Translation*, UCLA February 1960]

Session 11: EQUIPMENT

SYSTEM DESIGN OF A COMPUTER FOR RUSSIAN-ENGLISH TRANSLATION

Robert E. Wall, Jr.
University of Washington

Introduction

With the sole exception of the work with the photoscopic disc at the International Business Machines Corporation, all research in this country is being performed on general-purpose computers. Because of this fact, the thinking in machine translation is bound to be shaped by the characteristics and capabilities of the general-purpose machine. This seems to be the case in a recent Russian publication. The Russian, Bel'skaya, recently stated [1] that about 4000 stems should be sufficient to translate material restricted to one field of science. We find that at least 21, 000 stems, or about 260, 000 entries, are required for translation in the field of electronics. This agrees with the work of Dr. Sydney Lamb's group at Berkeley, who have compiled a lexicon of 25, 000 stems, also for the purpose of translation in a single field of science.

The question could then be asked, "why would one group estimate that a lexicon of 4,000 stems should be sufficient for translation in one field of science while two other groups nearly agree at 20-25, 000 stems?" Could it be because the group with the low estimate was concerned with English-Russian translation rather than Russian-English translation? It seems unlikely that this could be the reason. A more likely explanation is that the Berkeley and Washington groups were using, or anticipated using, equipment which would allow a lexicon of 20-25, 000 stems, while the Russian group, to the best of our knowledge, could not avail themselves of equipment of anywhere near this size. If this is true, it would seem that the estimates are not made on the basis of the linguistic requirements alone, but are greatly influenced by the capabilities of the equipment with which the group is familiar.

It would seem that perhaps it is time to start considering just what would be an ideal data-processing system for translation, and thus attempt to fit the equipment to the problem rather than the

problem to the equipment. This paper presents the general specifications for a digital data-processing system which would be desirable for machine translation according to the experience of the group at the University of Washington. First the problem of lexicon storage will be considered.

The Large Memory Problem

Naturally, we would like a memory of unlimited capacity, but since such a requirement is rather unrealistic, the estimate must be more modest. Just how large the memory must be depends partly on how the memory is used. The way the memory is used will likely follow one of two different philosophies: storage of word stems with logical processing to handle the inflectional forms; or "wholesale" storage, i. e. , the provision of a separate entry for each paradigmatic form. Although it would seem that dissection schemes will be a necessity since not all paradigmatic forms, and certainly not all compounds, can be predicted; it seems certain that stem storage and logical schemes should not be utilized to any greater extent than necessary, since some information is always lost in the dissection process. One other advantage may also be cited in favor of storage of matching units (source-language words) of greater length than the stem. Ultimately it is desirable that the input to the translator be either speech or printed text. Especially in the instance of speech one can say that the matching units must be at least syllables, and probably longer. The separate storage of stems and endings would thus seem to be dangerous, since the pronunciation of the ending would certainly be modified in many instances by the particular stem associated with the ending. The dictionary of matching units which is provided by a large translation lexicon would undoubtedly have great utility in conjunction with the analysis circuits of a speech-detection system. The matching units in the translation lexicon could then be stored in a code convenient for the particular matching process; perhaps in the form of sampled speech data for the instance of inputs which are spoken text.

One other lexical technique should be examined briefly. It has been suggested that the lexicon may be stored on tape, and that then the search process would proceed by first re-ordering the free forms of a considerable body of text into alphabetical order. The entire

search would then be accomplished by a single pass through the lexicon. This scheme has a disadvantage similar to that encountered in stem-ending dissection. Idiomatic sequences must be handled by means of processing programs, rather than by treating the entire idiom as a single semantic unit with a single lexical entry.

Because of the above, we favor wholesale storage of the predictable free forms (and also predictable idioms) and wish to reserve logical dissection schemes for the unpredictable forms.

The size of the storage required for a lexicon of predictable forms may be determined by estimating first the size of the individual entries, and second, the number of entries (or the number of predictable forms). The storage required for the individual entries will be considered first.

Size of the Storage for an Individual Entry

In the advanced version of the University of Washington operational lexicon, the individual entries contain four separate bodies of information: first, the source-language (Russian) semantic unit; second, the target-language (English) equivalents for that semantic unit; third, a coded "tag" which contains certain grammatical and non-grammatical information about that semantic unit; and fourth, any processing programs which apply to only that one semantic unit. The tag part of the entry is assigned a fixed length of 100 bits at the present time. The other three portions of the entry will vary considerably in length from entry to entry.

No attempt was made to optimize these tags, so a considerable reduction in length could undoubtedly be accomplished by eliminating impossible grammatical combinations from the code. For instance, 36 of the 100 bits are now used to code the case information: 6 bits for each of the 6 cases; 3 of the 6 bits for singular and 3 for plural; then the 3 bits are used for gender information (masculine, feminine, and neuter). In addition to the case information, the form class of the source-language word must be coded in the tag. To code the case possibilities, and in addition to provide space for coding substantive (noun), adjective, cardinal numeral, and multiple-form classes would require 40 bits of tag as we now construct the tag.

For optimal coding, 40 bits is far more tag length than necessary. For example, the appendix shows the case possibilities for the

standard Russian noun, adjective, cardinal numeral, and noun-adjective and noun-cardinal numeral multiple-form class inflections. In these lists there occur 52 noun, 23 adjective, 7 cardinal numeral, 60 noun-adjective and 2 noun-cardinal numeral possibilities. The total number of possibilities is 144. Since 8 bits may be used to code 256 unique possibilities, an 8-bit tag would be adequate to code the 144 possibilities. Hence a saving of 40-8 = 32 bits of storage per entry could be realized by adoption of a completely utilized tag.

The elimination of impossible combinations does increase the complexity of the programming problem, and programming convenience is often more important than possible sacrifices of storage. For this reason, and also because it seems certain that the tags will have to contain more information than that included in the present tag, tag length will probably remain at least 100 bits in length. It does not seem likely that it will have to exceed, say, 200 bits, however.

The lengths of the second and third parts of the entry, those used for storage of the source-language semantic unit and the target-language equivalents, are relatively easy to estimate. The average length of the Russian semantic units in the UW lexicon is 7 letters. A 7-bit code is adequate for coding the Russian alphabet (both upper and lower case), punctuation marks, and numerals. The total storage requirement for the source-language portion of the entry is then $7 \cdot 7 = 49$ bits.

For the target-language alternatives the average is about 2. 5 English words per entry. Since the average length of a word in English is about 5 letters, and since allowance must be made for a segmentation symbol (space), an average of 15 letters, or $7 \cdot 15 = 105$ bits, must be provided in each entry for the target-language material.

The fourth part of the entry is that assigned to the storage of certain processing routines. These routines apply only to the resolution of ambiguities existent in a unique entry. The most common example of routines which apply to only one entry are those used to delete specific alternatives on the basis of examination of context. In practice the context examination is ordinarily made by the general programs which modify the tag. The individual-entry routine then performs the deletions according to the tag pattern which has resulted. Our experience indicates that 8 to 14 program steps are required for

each entry, with an average of about 10 steps. A count of several hundred entries in our lexicon shows that nearly half have a deletion problem. The average length of deletion subroutines would then be about 10/2 = 5 program steps per entry.

More complex examples of individual-entry subroutines are indicated in the case of occurrences of есть , следует and оказывается. The processing required for есть , for instance, requires the search of the text sentence for an occurrence of a substantive in the nominative. If one is found, the equivalent "(to)eat" is deleted from the entry. Such processing required on the order of 25 to 30 program steps. Routines of this type are not required nearly as often as routines for deletion, however, so our present individual-entry subroutines are essentially those required for deletions. The use of individual-entry subroutines does aid in the conservation of high-speed storage, since these routines would not be located in the high-speed storage except when the entry was to be processed. For this reason it is likely that the individual-entry subroutine will be commonly used, with perhaps an average of 10 to 20 program steps per entry. Since, as will be shown, a 22-bit word length should be adequate for the computer, it may be estimated that the requirement will be

$10 \cdot 22 = 220$ bits minimum

$20 \cdot 22 = 440$ bits maximum

It is clear that all routines could be stored in high-speed storage, or on the other hand, nearly all could be stored as individual-entry subroutines. The choice is one of convenience.

The storage requirements for an individual entry may then be summarized as follows:

	Minimum	Maximum
Tag storage	100	200
Source-language (Russian) storage	49	49
Target-language (English) storage	105	105
Subroutines	220	440
	474	794

The Required Lexicon Storage

The present University of Washington operational lexicon contains just over 170, 000 entries. It can be shown that this lexicon is

fairly complete as far as scientific general language is concerned, but needs augmentation before translations may be made in any specific field of science. Micklesen [2] estimates that about 90, 000 selected entries must be added to the present lexicon in order to allow translation in the field of electronics. The required number of entries for an electronics-translation lexicon would then be at least

260, 000 • 474 = 123 million bits minimum
260, 000 • 794 = 206 million bits maximum

Additional augmentation of the dictionary which would be required to allow translations in fields other than electronics would depend on the similarity of the field to electronics. For physics, the additional expansion should be limited to considerably less than the 90, 000 entries required for electronics translation. If the field were more remote from electronics, such as organic chemistry or biology, the expansion would be probably about another 90, 000 entries, plus a good compound-dissection scheme for the case of organic chemistry.

It would take considerable effort for the University of Washington project to fill a 300 million-bit memory, since the present 170, 000-entry lexicon is not entirely made up with tags. As a matter of fact, it would seem unlikely that any group, or combination of research groups, could fill a 300 million-bit memory at the present. In time a memory of at least this size will be needed for the lexicon, and it is best that research should not be limited by machine capability. We would thus favor a lexical memory of at least 300, 000, 000-bit capacity, with a capability of handling matching units longer than the free form, and with a low access time, say less than 50 milliseconds. The desirable access time is, of course, partly determined by the processing units, since a very high speed processing unit will require a high-speed memory. The high-speed memory of the processing unit will be considered in the next section.

The High-Speed Storage

Both the general processing programs and sufficient text for context analysis must be stored in the high-speed memory of the machine. First, an estimate is made of the required size of text storage.

It is assumed that the translation is to proceed on a sentence-by-sentence basis. Certainly some translation problems cannot be

solved within the confines of the sentence, but at the present enough unsolved problems exist within the sentence to allow us to restrict our required context storage to the sentence.

For the translation to proceed on a sentence-by-sentence basis, the text storage must be at least large enough to store the entries for the longest sentence which will be encountered. If it is further assumed that the access time of the large memory is so long that the machine must have provision for concurrent lookup and processing, then sufficient storage must be provided so that after the processing is completed for one sentence, the next sentence is immediately available for processing.

To determine the distribution of sentence length, a word count of about 150 random Russian sentences was made. The results of the count are displayed in Figure 1, with sentence length plotted against the number of occurrences of sentences of that length. Two possible functions for approximating this curve are the function for the critically damped second-order system, and a hyperbolic function. One example of each of these functions would be:

$$(1) \qquad f(t) = .00583te^{-.0764t}$$

$$(2) \qquad y(t) = \frac{(8.4 \times 10^3)t}{t^{3.9} + 1.43 \times 10^5}$$

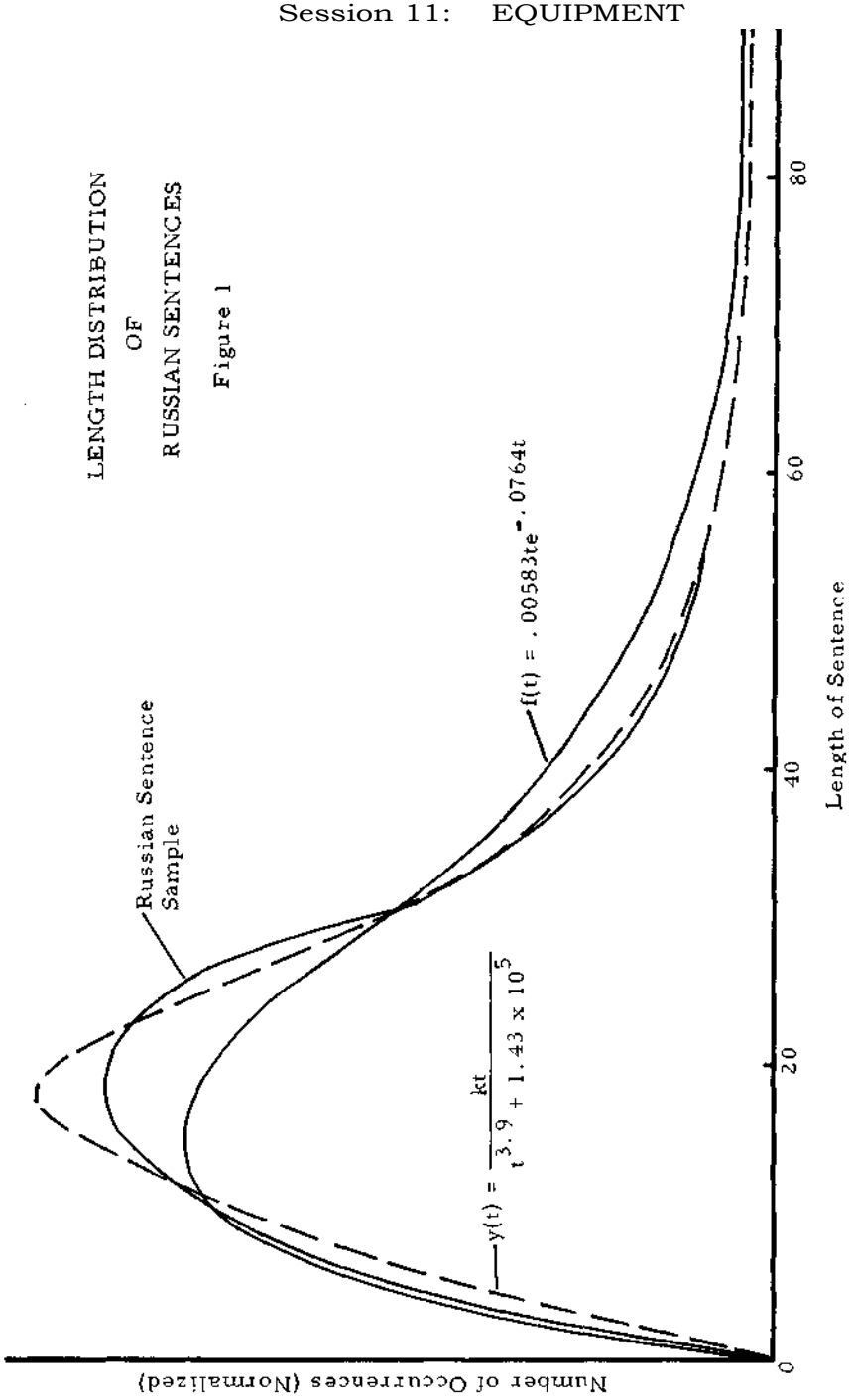
Both of these equations have been normalized to give an integral of 1, for t ranging from zero to infinity; they can then be used as probability density functions.

It is now possible to obtain an estimate of minimum text storage. The problem to be solved is the following: to find the number of consecutive text words which must be stored so that at any instant the probability is essentially 1 that two or more complete sentences are contained in the store. It will be assumed that this problem is equivalent to the following:

Given some ϵ, where $1 > \epsilon > 0$, select an n such that for two randomly selected sentences, the first of which contains a words and the second b words,

$$P[a + b < n] \geq 1 - \epsilon$$

In order to obtain a conservative estimate, Equation (1) will be chosen as the approximating function since this function gives a greater preponderance of long-length sentences than the empirical curve shows.


LENGTH DISTRIBUTION
OF
RUSSIAN SENTENCES
Figure 1
Russian Sentence
Sample
$y(t) = \frac{kt}{t^{3.9} + 1.43 \times 10^5}$
$f(t) = .00583te^{-.0764t}$
Number of Occurrences (Normalized)
Length of Sentence
0
20
40
60
80

First, the joint probability is calculated for two randomly selected sentences having combined lengths less than or equal to some specified amount L. From Equation (1) this can be shown to be-:

$$P = a^4 \int_0^L ue^{au} \int_0^{L-u} ve^{av}\, dv\, du$$

$$= a^2 e^{aL}\left[\frac{aL^3}{6} - \frac{L^2}{2} + \frac{L}{a} - \frac{1}{a^2}\right] + 1$$

where: a = -.0764

L = specified storage length.

Since one sentence consisted of 115 semantic units, which means 115 dictionary entries, it is interesting to calculate the probability of two randomly selected sentences' having a combined length of 115 entries, and also of a greater amount, say 150 entries. These probabilities can be shown to be:

$P_{115} \approx .973$

$P_{150} \approx .9964$

As was pointed out, storage for 115 entries is about the allowable minimum, and from the fact that $P_{150} \approx .9964$, it would seem that storage for about 150 entries should be adequate. The temporary storage required can then be estimated to be:

115 • 464 = 54, 300 bits minimum

150 • 774 = 116, 000 bits maximum

The high-speed memory must store the processing programs in addition to the text. The required processing-program storage will now be considered.

Program Storage

The processing programs are required to improve the quality of the translation over that obtained from word-for-word translation. The size of the totality of processing programs is thus determined by the amount of improvement desired and also by the ingenuity of the linguist and programmer.

The programs used total about 2, 750 program steps. Since the biquinary-coded IBM 650 was used, this required 7 • 10 • 2750 = 193, 000 bits of storage. If the logical operations AND and OR were available

(they are not in the IBM 650) then the same processing, exclusive of the load and unload operations, could be performed in about 1, 100 to 1, 200 program steps.

The ultimate requirements for processing routine storage cannot be estimated with any degree of assurance at the present time.

Actually, it is often quite arbitrary as to what is considered as dictionary material and what is considered as components of the processing routines.

If the individual-entry routines and addresses of locations where deletions are to be made under specified conditions are considered as parts of the processing information, the processing routines may approach a size comparable to that of the dictionary. As the programs get larger, however, sophisticated programming techniques give such a wealth and diversity of subprograms that adding another routine, even though quite complex in theory, usually results in only a short routine for initializing a programming path through various subroutines which are already included in the master program. Certainly, it can be said that for the case of carefully written programs in computer language, it would require a very considerable programming effort just to fill 500, 000 or 600, 000 bits of storage, entirely aside from the task of discovering just what to program in the first place. Hence, 1 million bits of high-speed memory should be adequate for a translator for the foreseeable future.

The problem of high-speed storage requirements is determined in part by the operation codes available and also by proper selection of computer word length. The minimum word length is set by the amount of information which must be included in an instruction word. For instance, with 64 operation codes, 6 bits must be provided in each instruction word for the operation code. For a store of about 1 million bits, standard practice calls for about 32, 000 words of 36 bits each. This requires about 32,000 addresses, which calls for another 15 bits. It would seem, then, that a word length of at least 21 bits is required.

A 36-bit word length would seem a little awkward. A word length of 22 bits would allow 64 operation codes and 64, 000 addresses.

A core memory might be easier to construct, however, if 32, 000 words were provided of length 44 bits each. In the second instance, two program instructions could be stored in a computer word and would be executed in sequence in the program register.

Closely tied with the problems of word length is the problem of specification of the register storage. The registers, or accumulators as they are usually called, should be of sufficient capacity to store either the entire tag or all target-language equivalents of an entry at one time. For the tag, somewhere between 100 and 200 bits would be required. For the target-language equivalents about 105 bits are required for the average entry. It would seem that about 200 bits of register storage should be adequate.

The Operation Codes

Of the 44 operation codes available in the IBM 650, 13 codes have not been used in any of the University of Washington machine translation programs. On the other hand the logical operations, logical AND and logical OR, and a ROTATE SHIFT operation would be very helpful. From our experience the following set of operation codes should provide all necessary operations for efficient coding in machine language.

NO OPERATION	SHIFT LEFT
STOP BRANCH ZERO	BRANCH ZERO UPPER
ADD UPPER	BRANCH ZERO
SUBTRACT UPPER	BRANCH MINUS
ADD LOWER	RESET AND ADD UPPER
STORE LOWER	RESET AND SUBTRACT UPPER
STORE UPPER	RESET AND ADD LOWER
STORE DATA ADDRESS	PUNCH
SHIFT RIGHT	TRANSFER
ROTATE SHIFT RIGHT	AND
RESET AND ADD COMPLETE	OR
	BRANCH POSITION X ZERO

The operations listed are common on general-purpose machines except for the RESET AND ADD COMPLETE operation. We list this since it has been found to be tedious to transfer tags and equivalents to the registers from general storage. The difficulty comes about because the entries are stored randomly in general storage. The

address of the first computer word of each portion of an entry is found by a table-lookup operation. Succeeding computer words for the same portion of the entry are then located by adding, consecutively, a 1 to the address of the last computer word. Adding these 1's becomes tedious. A simpler scheme, especially since this operation is very common, is to provide the RESET AND ADD COMPLETE operation. In this operation, it is envisaged that the address of the first computer word to be transferred to register would be entered in the DATA ADDRESS part of the instruction word. When the RESET AND ADD COMPLETE operation is executed, the computer would not only reset and add the addressed computer word into the register, but would continue, adding the next word after the addressed word into the next register, and continuing until all registers were full. Since it is likely that a ring-type register with a capacity of at least 4 or 5 computer words would be optimal, this would provide a considerable programming convenience without a very great increase in complexity in the logical circuitry of the arithmetic unit.

Summary

The translation-computer specifications suggested in this paper indicate several possible deviations from standard general-purpose designs. The large memory should be of a different type from the usual tape units. Since erasibility is not necessary, memories such as the Photoscopic Disc would seem ideal. The high-speed memory of large general-purpose machines would seem to be of adequate size, but it is likely that a more efficient device could be realized with a non-standard word length. The operation codes should be somewhat different from those of large general-purpose machines. Thus it would seem that researchers should be alert to the possibility of prescribing data-processing system design in order to obtain more nearly optimal equipment.

REFERENCES

[1] Bel'skaya, I. K., Some General Problems in Machine Translation, Soviet Developments in Information Processing and Machine Translation, Collection of Articles on MT, Moscow, 15-21 May 1958 (page 8).

[2] Micklesen, Lew R., "Procedural Report", Linguistic and Engineering Studies in Automatic Language Translation, Report No. RADC-TN-58-321, ASTIA Document No. AD-148992, 1958, p. 24.

Session11: EQUIPMENT

ADJECTIVES

*1.	MNS	нов
*2.	MNS/MAS	белый
3.	MNS/MAS/FGS/FDS/FIS/FLS	большой
4.	MGS/MAS/NGS	белого
5.	MGS/MAS/FNS/NGS	женина
6.	MDS/NDS	белому
7.	MDS/FAS/NDS	женину
8.	MIS/DP	волчьим
9.	MIS/NIS/DP	белым
10.	MIS/MLS/NIS/NLS/DP	всем
11.	MLS/NLS	белом
12.	MLS/NIS/NLS	волчьем
*13.	FNS	белая
14.	FGS/FDS/FIS/FLS	белой
*15.	FAS	белую
*16.	FIS	белою
17.	NNS	ново
*18.	NNS/NAS	белое
19.	NNS/NAS/NP/AP	все
20.	NP	новы
21.	NP/AP	белые
22.	GP/AP/LP	белых
23.	IP	белыми

*These combinations are also found in nouns.

When 3 symbols:
1st symbol--gender
2nd symbol--case
3rd symbol--number

When 2 symbols:
1st symbol -- case
2nd symbol--number

Session 11: EQUIPMENT

NOUNS

Masculine	Feminine	Neuter
1. NS француз	1.NS книга	1. GS яблока
2. GS стола	2. AS книгу	2. DS слову
3. DS столу	3. IS книгой	3. IS словом
4. AS дядю	4. GP книг	4. LS слове
5. IS столом	5. DP книгам	5. GP слов
6. LS столе	6. IP книгами	6. DP словам
7. NP французы	7. LP книгах	7. IP словами
8. GP столов	8. NS/AS ночь	8. LP словах
9. DP столам	9. GS/DS/LS/NP дочери	9. NS/AS слово
10. IP столами	10. GS/DS/LS/NP/AP ночи	10. NS/AS/LS поле
11. LP столах	11. GS/NP дамы	11. GS/DS/LS времени
12. NS/AS стол	12. GS/NP/AP книги	12. GS/NP/AP слова
13. GS/AS француза	13. DS/LS книге	13. NP/AP яблоки
14. GS/AS/NP доктора	14. IS/GP/AP тетей	14. all cases, S and P кино
15. GS/DS/LS/NP/AP пути	15. GP/AP дам	
16. GS/NP дяди		
17. GS/NP/AP вечера		
18. DS/LS берегу		
19. IS/GP/AP дядей		
20. LS/NP пролетарии		
21. LS/NP/AP жребии		
22. NP/AP столы		
23. GP/AP французов		

CARDINAL NUMERALS

1.	N / A	три
2.	G / D / L	пяти
3.	G / A / L	трех
4.	G / D / I / L	сорока
5.	D	трем
6.	I	тремя
7.	L	двухстах

Session 11: EQUIPMENT

I. MULTIPLE FORM, ADJECTIVE-NOUN: words have adjectival declension regardless of part-of-speech usage.

	Noun usages	Adjective usages	Example
*1.	MNS, MAS	MNS, MAS	согласный
*2.	MGS	MGS, MAS, NGS	согласного
*3.	MDS	MDS, NDS	согласному
4.	MIS, MDP	MIS, NIS, DP	согласным
*5.	MLS	MLS, NLS	согласном
6.	MNP, MAP	NP, AP	согласные
7.	MGP, MLP	GP, AP, LP	согласных
8.	MIP	IP	согласными
*9.	MNS	MNS, MAS	рабочий
*10.	MGS, MAS	MGS, MAS, NGS	рабочего
11.	MNP	NP, AP	рабочие
12.	MGP, MAP, MLP	GP, AP, LP	рабочих
*13.	MNS	MNS, MAS, FGS, FDS, FLS, FIS	больной
*14.	FNS	FNS	столовая
*15.	FGS, FDS, FIS, FLS	FGS, FDS, FIS, FLS	столовой
*16.	FAS	FAS	столовую
*17.	FIS	FIS	столовою
18.	FNP, FAP	NP, AP	столовые
19.	FGP, FLP	GP, AP, LP	столовых
20.	FDP	MIS, NIS, DP	столовым
21.	FIP	IP	столовыми
22.	FNP	NP, AP	горничные
23.	FGP, FAP, FLP	GP, AP, LP	горничных
24.	MNP, FNP	NP, AP	знакомые
25.	MGP, MAP, MLP, FGP, FAP, FLP	GP, AP, LP	знакомых
*26.	NNS, NAS	NNS, NAS	легкое
*27.	NGS	MGS, MAS, NGS	легкого
*28.	NDS	MDS, NDS	легкому
29.	NIS, NDP	MIS, NIS, DP	легким
*30.	NLS	MIS, NIS	легком
31.	NNP, NAP	NP, AP	легкие
32.	NGP, NLP	GP, AP, LP	легких
33	NIP	IP	легкими

Session 11: EQUIPMENT

Noun usages	Adjective usages	Example
*34. NNS	NNS, NAS	животное
35. NGS, NAS	MGS, MAS, NGS	животного
36. NNP	NP, AP	животные
37. NGP, NAP, NLP	GP, AP, LP	животных

* Combinations covered in list of adjective-only combinations.

II. MULTIPLE FORM, ADJECTIVE-NOUN: nouns have noun declension, adjectives have adjective declension.

	Noun usages	Adjective usages	Examples
*1.	MNS, MAS	MNS	бросок
2.	MNS, MAS	MLS, NLS	том
**3.	MNS, MAS	MNS, MAS, FGS, FDS, FIS, FLS	простой
4.	MGS	FNS	долга
5.	MDS	FIS	простою
**6.	MDS	MDS, NDS	тому
7.	MIS	MNS	знаком
8.	MIS	MLS, NLS	другом
9.	MLS	NNS, NAS	простое
10.	MNP, MAP	NP	долги
*11.	FNS	FNS	дорога
12.	FGS, FNP, FAP	IP	теми
13.	FGS, FNP, FAP	NP	дороги
14.	FDS, FLS	comparative	суше
**15.	FIS	FGS, FDS, FIS, FLS	резкой
**16.	FIS	MNS, MAS, FGS, FDS, FIS, FLS	дорогой
17.	FGP	MNS	дорог
18.	FGP	MIS, NIS, DP	тем
*19.	NNS, NAS	NNS	право
20.	NGS	FNS	добра
21.	NGS, NNP, NAP	FNS	права
**22.	NIS	MLS, NLS	правом
23.	NGP	MNS	прав

* Combinations covered in list of noun-only combinations.

** Combinations covered in list of adjective-only combinations.

MULTIPLE-FORM, CARDINAL NUMERAL - NOUN

	Noun usage	Numeral usages	Example
1.	FGP	N, A	сорок
2.	FNS	G, D, I, L	сорока