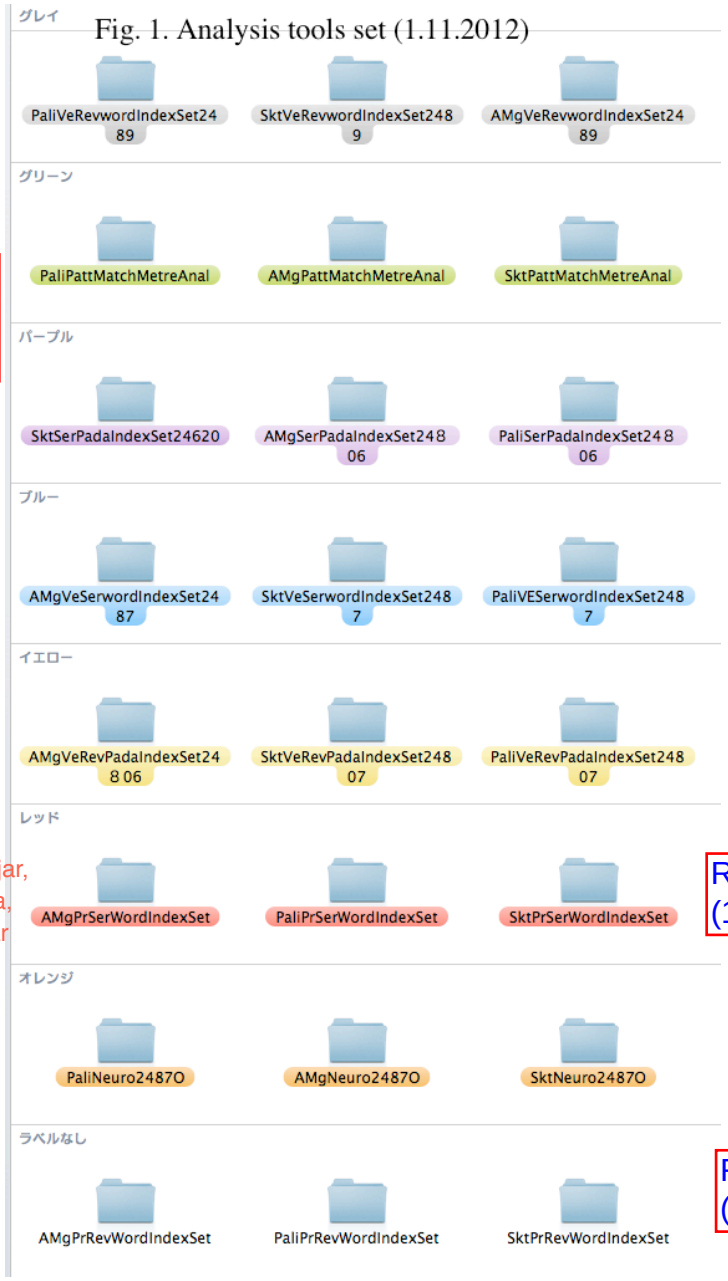
1st Nov. 2012
6th Nov. 2012
7th Nov. 2012
15th & 30th June 2014
17th Aug. 2014

Remarks on revised part of the tools

Yumi Ousaka, Yasutomo Nishi and Sunao Kasamatsu

We released analysis tools that were created in Java for analysing texts in Sanskrit, Pāli and Ardha-Māgadhī in 2005. Recently we revised them so as to be able to display some error messages, to remove some errors in tools, and to improve some tools. Figs. 1 and 2 show our completely renewed tool sets. We will give some remarks on using these tools.

Fig. 1. Analysis tools set (1.11.2012)



Revise of 6 metre analysis tools (30.06.2014)

revise of AMgPrSerwordIndexMerge3.jar, PaliPrSerwordIndexMerge3.ja, SktPrSerwordIndexMerge3.jar

Revise of 3 Index sets (17.08.2014)

Revise of 3 Index sets (17.08.2014)

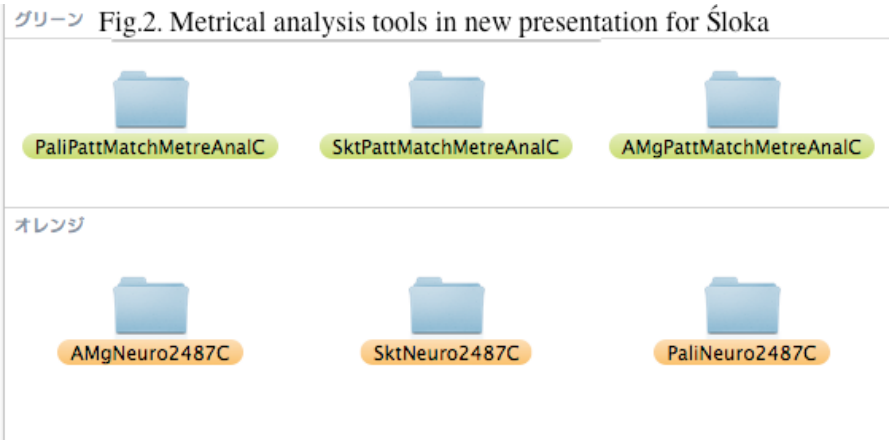


Fig.2. Metrical analysis tools in new presentation for Śloka

1. Pali96 Fonts

Table 1 shows the keyboard input of the Pali96 font on Mac OS and the ASCII codes for the letters. Three alphabet tables for Sanskrit (Skt), Pāli and Ardha-Māgadhī (AMg) are shown in Tables 2, 3 and 4, respectively.

We revised the Pali96 fonts as follows (see Table 1):

(a) We changed the font face of characters r̥ and r̥̄ from r̥ and r̥̄, respectively, but their ASCII codes remain the same . As shown in Table 1, special characters such as 'r̥' are typed by pressing a standard character such as 'r' together with the Option key. These characters are used in texts in Sanskrit .

(b) We created a special character 'l̥' with ASCII code number 239. We can input this by pressing a standard character 'j' together with both the Option key and Shift key. This character is used in texts in Sanskrit.

(c) Note that the special key 'ḷ' keeps its ASCII code and keyboard input as before. This is a consonant in Pāli.

Special attention must be paid to four special characters (/, // , | and '), that are listed in the last part of the table. The first two must be used when preparing text for metre analysis or when producing an index to a canonical text consisting mainly of poetry (verse). The third one is used when producing a pÃda index. The last one is used for the presentation of the prodelision form.

Table 1. Keyboard input of the Pali96 font and its assignment to the ASCII numeric code: Special characters, such as ‘ā’, are typed by simultaneously pressing the Option key and the normal character key (e.g. ‘a’). The same is true for the other characters, such as ‘ṃ, kh,’ etc. The special character ‘ī’ is obtained by simultaneously pressing the Option key and the normal character ‘i’, followed by the keyboard input of the character ‘i’. The character ‘ū’ is obtained in a similar way. The special character ‘ṭh’ is obtained by simultaneously pressing three keys: the Option key, the Shift key and the normal character ‘y’. Normal Roman characters, such as ‘a, i’, etc., are typed in the usual way. (Table is replaced by new one o 6th Nov. 2012. (compare to Table 1))

characters	key stroke	ascii code	characters	key stroke	ascii code
ā	<O> + a	140	ṃr	<O> + }	200
ī	<O> + i, i	148	ṃl	<O> + #	220
ū	<O> + u, u	159	ṃv	<O> + $	221
ḥ	<O> + h	250	ṃś	<O> + %	222
r̥	<O> + r	168	ṃs	<O> + &	223
r̥̄	<O> + <S> + m	229	ṃh	<O> + <S> +0	226
l̥	<O> + <S> + j	239	th	<O> + y	180
ṃ	<O> + m	181	dh	<O> + z	189
kh	<O> + k	251	ph	<O> + p	185
gh	<O> + 9	187	bh	<O> + b	186
ṅ	<O> + g	169	ḷ	<O> + l	194
ch	<O> + c	141	ś	<O> + s	167
jh	<O> + j	198	ṣ	<O> + x	197
ñ	<O> + <S> + i	246	/	<O> + 1	193
ṭ	<O> + t	160	//	<O> + 2	170
ṭh	<O> + <S> + y	231	|	<O> + 3	163
ḍ	<O> + d	182	||	<O> + 4	162
ḍh	<O> + i, U	243	–	<O> + 5	176
ṇ	<O> + n, n	150	∪	<O> + 7	166
ai	<O> + !	218	⊔	<O> + 8	165
au	<O> + "	219	≃	<O> + 0	188
ṃy	<O> +]	199	’	<O> + <S> +[	213
additional characters					
ṁ	<O> + f	196	ã	<O> + n, a	139
ĩ	<O> + `, i	147	ũ	<O> + e, u	156

Tables 2, 3 and 4. Alphabet tables for Sanskrit (Skt), Pāli and Ardha-Māgadhī (AMg) in alphabetical order.

順番	文字	順番	文字	順番	文字	順番	文字
1	a	15	ṃy	29	j	43	ph
2	ā	16	ṃr	30	jh	44	b
3	i	17	ṃl	31	ñ	45	bh
4	ī	18	ṃv	32	ṭ	46	m
5	u	19	ṃś	33	ṭh	47	y
6	ū	20	ṃs	34	ḍ	48	r
7	ṛ	21	ṃh	35	ḍh	49	l
8	ṝ	22	k	36	ṇ	50	v
9	ḷ	23	kh	37	t	51	ś
10	e	24	g	38	th	52	ṣ
11	ai	25	gh	39	d	53	s
12	o	26	ṅ	40	dh	54	ḥ
13	au	27	c	41	n	55	h
14	ṃ	28	ch	42	p		

順番	文字	順番	文字	順番	文字	順番	文字
1	a	14	gh	27	th	40	v
2	ā	15	ṅ	28	d	41	s
3	i	16	c	29	dh	42	h
4	ī	17	ch	30	n		
5	u	18	j	31	p		
6	ū	19	jh	32	ph		
7	e	20	ñ	33	b		
8	o	21	ṭ	34	bh		
9	ḥ	22	ṭh	35	m		
10	ṃ	23	ḍ	36	y		
11	k	24	ḍh	37	r		
12	kh	25	ṇ	38	ḷ		
13	g	26	t	39	l		

順番	文字	順番	文字	順番	文字	順番	文字
1	a	13	gh	25	t	37	l
2	ā	14	ṅ	26	th	38	v
3	i	15	c	27	d	39	s
4	ī	16	ch	28	dh	40	h
5	u	17	j	29	n		
6	ū	18	jh	30	p		
7	e	19	ñ	31	ph		
8	o	20	ṭ	32	b		
9	ṃ	21	ṭh	33	bh		
10	k	22	ḍ	34	m		
11	kh	23	ḍh	35	y		
12	g	24	ṇ	36	r		

2. Program structure.

We developed analysis tools for three different languages, Sanskrit, Pāli and Ardha-Māgadhī (see Figs. 1 and 2). All the tools for the different languages have analogous program sets. We will explain the program sets by taking as an example a sample set for Sanskrit, which is shown in Table 5.

We have two types of text: one consists mainly of poetry (verse), and the other of prose. For the study of the poetry we prepared the three preprogram tools: metre analysis, pāda index production and word index production. The production of the pāda index is divided into two steps: (1) production of a serial index and (2) production of a reverse index. The serial pāda index production tools consist of two programs called SktVeSerPadaIndex1.jar and SktVeSerPadaIndex2.jar, where the abbreviations Skt, Ve and Ser mean Sanskrit, Verse and Serial, respectively. These programs should be used in order. Instructions about how to use these programs are found in References listed below.

Table 5. Program structure

Text type	Procedure	Tool	Program name
Verse	Metrical nallysis	Neuro method	SktNeuroMetreAnalNo1.jar
		Pattern matching	SktPattMatchMetreAnalNo1.jar
	Pāda index	Serial	SktVeSerPadaIndex1.jar, SktVeSerPadaIndexLine2.jar
		Reverse	SktVeRevPadaIndex1.jar, SktVeRevPadaIndexLine2.jar
	Word index	Serial	SktVeSerWordIndex1.jar, SktVeSerwordIndexLine2.jar,
		Reverse	SktVeRevWordIndex1.jar, SktVeRevWordIndexLine2.jar,
Prose	Word index	Serial	SktPrSerWordIndex0.jar (or) SktPrSerWordIndex1.jar, SktPrSerWordIndexLine2.jar, SktPrSerwordIndexMerge3.jar
		Reverse	SktPrRevWordIndex1.jar, SktPrRevWordIndexLine2.jar, SktPrRevWordIndexMerge3.jar

Improvement in the program are as follows: In the previous version programs could not show error messages. This was very inconvenient when using our programs. Programs sometimes stopped if the text form was in violation of the rule. We added a function to the programs that can save the text itself into an output file after reading it. So the reason the program stopped is easily found by checking the output of the text. We show this in the figure below, by using the program for the production of the serial pāda index. The left-hand bottom figure shows the input file, and the right-hand bottom file shows the output file. (Note that the error message on the right-hand top may not be displayed.) Usually the end line in the output file shows the text line where the program cannot proceed properly. So you should correct any error included in this line. After removing all errors in the text, this program will finish properly and save the calculated results together with this text line into the file. Then you should remove all text lines in this file since these text lines interfere with running the second program if this program uses the first calculated result to obtain the final result. So please note this when using our new versions of the programs.

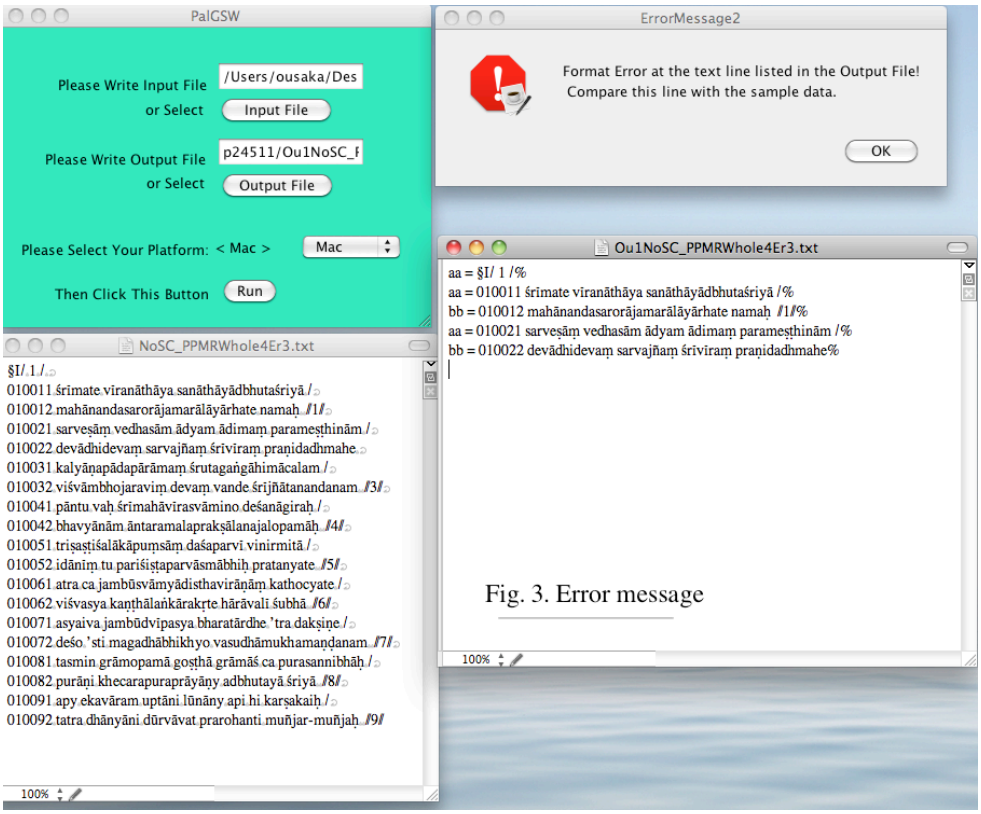


Fig. 3. Error message

We made programs so as to delete every redundant space of the beginning of each text line, but sometimes this process doesn't work well. So be careful to prepare such a text that does not have some redundant spaces.

Finally let note that the program tools are able to process the plain text form in Mac Roman (see reference (2)). In Macintosh many editors , e.g. Jedit X, Nisus Writer, etc. are available for making such text files or for opening the calculated files.

3. Simple editor "Edisan5_5.jar" (Revise of keystroke on 8th Nov. 2012)

We designed our Pali96 fount so that each special fount or symbol should be displayed in its proper form on CRT, where every fount should be assiged a single ASCII code with respect to program coding. Our fount was originally designed on Mac OS (Classic OS). So there is no problem to use our fount even on the recent new Mac OSX.

We released the revised simple editor "Edisan5_5.jar." We can type characters of Pali96 fount on Windows OS and on Linux OS as shown in Table 6 (compare to Table 1). We found that some characters cannot be displayed in their proper forms on CRT, although their ASCII codes are preserved correctly. Unfortunately we cannot improve this difficulty, because it seems like to strongly relate with OS itself. We hope this may not cause serious problem on using our tools.

Table 6. Key stroke of Pali96 in Edisan5_5. <A> and <S> mean Alt key and Shift key, respectively. Character ḷ can be typed by simultaneously pressing the Alt key and the normal character key h)

characters	key stroke	ascii code	characters	key stroke	ascii code
ā	<A> + a	140	ṃr	<A> + }	200
ī	<A> + i	148	ṃl	<A> + <S> + 3	220
ū	<A> + u	159	ṃv	<A> + <S> + 4	221
ḥ	<A> + h	250	ṃś	<A> + <S> + 5	222
ṛ	<A> + r	168	ṃs	<A> + <S> + 6	223
ṝ	<A> + <S> + m	229	ṃh	<A> + <S> + 7	226
ḷ	<A> + <S> + j	239	th	<A> + y	180
ṃ	<A> + m	181	dh	<A> + z	189
kh	<A> + k	251	ph	<A> + p	185
gh	<A> + 9	187	bh	<A> + b	186
ṅ	<A> + g	169	ḷ	<A> + l (el)	194
ch	<A> + c	141	ś	<A> + s	167
jh	<A> + j	198	ṣ	<A> + x (or) <A>+ <S> + e	197
ñ	<A>+<S> + i	246	/	<A> + 1 (or) <A>+ <S> + g	193
ṭ	<A> + t	160	//	<A> + 2	170
ṭh	<A> + <S> + y	231	|	<A> + 3	163
ḍ	<A> + d	182	‖	<A> + 4	162
ḍh	<A> + <S> + d	243	–	<A> + 5	176
ṇ	<A> + n	150	∪	<A> + 7	166
ai	<A> + <S> + 1	218	∽	<A> + 8	165
au	<A> + <S> + 2	219	≋	<A> + 0	188
ṃy	<A> +]	199	’	<A> + <S> +[(or) <A>+ <S> + h	213
additional characters					
ṁ	<A> + f	196	ã	<A> + <S> + a	139
ĩ	<A> + <S> + l (el)	147	ũ	<A> + <S> + u	156

4. References

Consult the book (1) below with respect to algorithms of the programs. Instructions about how to use our programs refer to (2) to (4). These books can be downloaded from the site of https://www.cari.ne.jp/search/.

(1) Y. Ousaka , Automatic Analysis of the Canon in Middle Indo-Aryan by Personal Computer with Object Files and Their Programs for Macintosh and Windows OS on CD-ROM, Philologica Asiatica, Monograph Series 19 (2002) 86p.
(see also http://hirose.sendai-nct.ac.jp/~ousaka)

(2) Y. Ousaka , Automatic Analysis of the Canon in Middle Indo-Aryan by Personal Computer II: in both Japanese and English, with Jar Files and Their Java Programs by Java for Macintosh OSX, Windows XP, and Linux on CD-ROM Philologica Asiatica, Monograph Series 21 (2005) 85p.

(3) Y. Ousaka, Metre Analysis in Middle Indo-Aryan Based on New Combined Method of Neural Network and Discriminant Analysis, in Philologica Asiatica, Monograph Series 26 (2010) 45

(4) S. Kasamatsu, Y. Nishi and Y. Ousaka , Automatic Analysis of the Canon in Middle Indo-Aryan by Personal Computer III (in Japanese), Bulletin of Chuo Academic Research Institute, No 41 (2012) 35-52.