http://dx.doi.org/10.14257/ijsip.2017.10.5.05

The Challenges of Kazakh Coded Character Processing and Relevant Solutions

Jun Dong[1,2,3], Li Cheng[1,3], Yong Yang[4] and Tonghai Jiang[1,3*]

[1]The Xinjiang Technical Institute of Physics & Chemistry.Chinese Academy of Science, Urumchi 830011, China
[2]University of Chinese Academy of Sciences, Beijing 100049, China
[3]Xinjiang Laboratory of Minority Speech and Language Information Processing, Urumchi 830011, China
[4]College of Computer Science, Xinjiang Normal University, Urumchi 830054, China
**jth@ms.xjb.ac.cn*

Abstract

Information systems for Kazakh language processing in China must handle the editing and display problems caused by four special letters: ٵ, ٶ, ٷ, and ٸ. The current solution uses combinations of four alternative letters (ا, و, ۇ, and ى) with the character ٴ to represent these four special letters. However, this approach does not conform to the international Unicode standard or the Chinese national standard GB 21669. In addition, computer programs cannot semantically distinguish the alternative letters from the original letters. This causes problems in Kazakh text-processing applications such as text sorting, script conversion and speech synthesis. We propose a compromise method that avoids most of the shortcomings of the letter substitution method. The new method involves three rules: First, the four special letters should be represented by combinations of themselves and the character ٴ. Second, the glyphs with ٴ of the initial form, medial form, and final form should not be included in OpenType fonts. Third, the glyphs with ٴ of the isolated form should only be used when the four special letters are not adjacent to Kazakh letters. The relevant glyph layout features in the OpenType font format are compatible with the compromise method.

***Keywords**: Kazakh, coded character, Unicode, OpenType*

1. Introduction

There are approximately 1.46 million Kazakh people living in China, mainly concentrated in three counties in Xinjiang province, namely, Ili Kazakh Autonomous Prefecture, Mori Kazakh Autonomous County, and Barkol Kazakh Autonomous County [1]. In these regions, Kazakhs play key roles in government administration, justice systems, education, journalism, and publishing. In recent years, with the growing cultural exchange and trading activities between China and its neighboring countries in central Asia such as Kazakhstan and Uzbekistan, Kazakh has become an important language for international communication.

The Kazakh language uses an alphabetic-writing-based Arabic alphabet. However, four letters (ٵ, ٶ, ٷand ٸ) are very difficult for computer programs to edit and display because of their special writing rules. The current Kazakh processing rules defined in the international (Unicode) standard and the Chinese national standards (GB 21669) are insufficient to handle these four letters. To process Kazakh without standard support for

[*] Corresponding Author

these letters, current information systems typically use combinations of four alternative Kazakh letters (ا, و, ۇ, and ى) with the character ٴ to represent the letters ٴا, ٴو, ٴۇ, and ٴى [2-4]. However, this approach does not conform to the Unicode or GB 21669 standard. In addition, although these four alternative letters may display correctly on screen, the fundamental differences between the two sets of letters cannot be semantically distinguished by computer programs. Therefore, problems occur in the sorting of Kazakh texts. Moreover, additional issues arise in software applications such as script conversion[1] and speech synthesis that require semantic understanding. These challenges are the main reason why our group sought a better method, one that can handle the relevant display and editing needs while simultaneously addressing the language processing issues.

2. Special Writing Rules for the Letters ٴا, ٴو, ٴۇ and ٴى

Kazakh uses the same right-to-left cursive writing rules as Arabic. This means that the current letter must be joined to adjacent letters. Consequently, the same letter may be written in different presentation forms, known as the isolated form, initial form, medial form, and final form, depending on how it is joined to its neighbors (see Figure 1 and Figure 2).

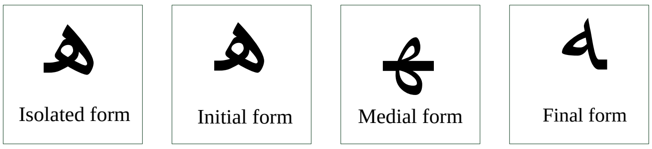

Figure 1. Presentation Forms of the Kazakh Letter ھ

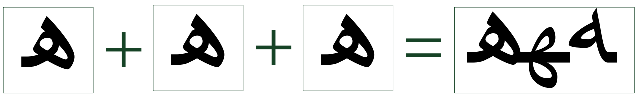

Figure 2. Three Adjacent Instances of the Kazakh Letter ھ

In addition, the four special Kazakh letters ٴا, ٴو, ٴۇ, and ٴى are subject to three more complex additional rules as follows:

(1) A small symbol ٴ appears at the upper right corner of all four letters, ٴا, ٴو, ٴۇ, and ٴى. When these letters appear in a word, the ٴ symbol must be placed at the beginning of the word, for example, ب+ٴى+ز=ٴبىز (meaning "us").

(2) The symbol ٴ should be written only once when several instances of ٴا, ٴو, ٴۇ, or ٴى appear in one word, for example, ٴو+م+ٴى+ر=ٴومىر (meaning "life").

(3) The symbol ٴ should be omitted when any of the letters ٴا, ٴو, ٴۇ, or ٴى and any of the letters گ, ك, or ە occur in the same word, for example, ٴا+ك+ە=اكە (meaning "father").

3. Deficiencies of the Letter Substitution Method

The international Unicode standard provides basic support for Kazakh processing. In Unicode, the coded characters for all Kazakh letters and the symbol ٴ are defined in the Arabic block (character codes: 0600–06FF). The coded characters ٴ (character code:

[1] The Kazakh alphabet that is used in former Soviet Union countries such as Kazakhstan, Uzbekistan and Russia is based on the Cyrillic alphabet and needs to be converted into a Kazakh-based Arabic alphabet for understanding in China.

0674), ٵ (character code: 0675), ٶ (character code: 0676), ٷ (character code: 0677), and ٸ (character code: 0678) are dedicated to processing Kazakh [5]. In addition, both the bidirectional algorithm and the letter cursive rules in Unicode are also applied for Kazakh processing, although they are defined for handling complex scripts such as Arabic [6,7]. However, Unicode does not define the presentation forms of any of the four special letters as coded characters, except for the isolated-form coded character ﯝ (character code: FBDD) for the letter ٷ [8,9]. The Chinese national standard GB 21669:2008 "Information technology-Uyghur, Kazakh, and Kirghiz Coded Character Set" defines coded characters for the presentation forms of the four special letters [10]. However, neither Unicode nor GB 21669 provides a solution for the special rules for writing the letters ٵ, ٶ, ٷ, and ٸ.

At present, mainstream operating systems (OSs) such as Windows and Linux support both the bidirectional algorithm and the letter cursive rules defined in Unicode. Support for the letter cursive rules is commonly achieved by using an OpenType font. In addition to glyphs, an OpenType font defines glyph layout features [11]. An OS can correctly handle the Kazakh right-to-left cursive writing rules because these rules are consistent with the Unicode definition. However, despite the powerful glyph layout capabilities of the OpenType font format, it still cannot handle the special writing rules for the letters ٵ, ٶ, ٷ, and ٸ. For example, the symbol ٴ in the Kazakh word ب+ٸ+ز=ٴبىز (meaning "us") should be displayed at the beginning of the word. However, it will instead be displayed in the middle of the word by an OS that supports OpenType fonts (see Figure 3) because the OpenType font format cannot independently move the ٴ symbol—which is associated with a glyph—to the beginning of the word.

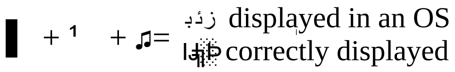

Figure 3. The Display of the Kazakh Word ٴبىز

Currently, the letter substitution method is used instead to process these four special letters in almost all information systems that must handle Kazakh text. One example is the Chaojie Publishing Software that is widely used in Xinjiang province and on major Kazakh websites such as kazak.ts.cn, kazak.xjkunlun.gov.cn, and kazakh.people.com.cn. According to the letter substitution method, the combination of a different Kazakh letter, ا, و, ۇ, or ى, with the character ٴ is employed as the replacement for the letter ٵ, ٶ, ٷ, or ٸ, respectively. For example, the word ب+ٸ+ز=ٴبىز (meaning "us") is spelled as ٴ+ب+ى + ز=ٴبىز when displayed in an OS, as shown in Figure 4. The replacement of the letters is invisible to the reader because the glyphs for the replacement letters and the original ones are identical when the glyphs without ٴ are considered (see Table 1).

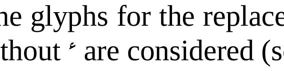

Figure 4. The Display of the Kazakh Word ٴبىز According to the Letter Substitution Method

Table 1. The Presentation Forms and Glyphs for the Letters (ٵ, ٶ, ٷ, and ٸ) and (ا, و, ۇ, and ى)

	The presentation forms and glyphs for the letters ٵ, ٶ, ٷ, and ٸ				The presentation forms and glyphs for the letters ا, و, ۇ, and ى			
Letter	ٵ	ٶ	ٷ	ٸ	ا	و	ۇ	ى
Isolated form	ا	و	ۇ	ى	ا	و	ۇ	ى
	ٵ	ٶ	ٷ	ٸ				
Initial form				ىـ				ىـ
				ٸـ				
Medial form				ـىـ				ـىـ
				ـٸـ				
Final form	ـا	ـو	ـۇ	ـى	ـا	ـو	ـۇ	ـى
	ـٵ	ـٶ	ـٷ	ـٸ				

However, the letter substitution method does not conform to Unicode or GB 21669 because the coded characters ٵ (character code: 0675), ٶ (character code: 0676), ٷ (character code: 0677), and ٸ (character code: 0678) are not used. These four coded characters are dedicated to the processing of Kazakh, according to both the Unicode and GB 21669 standards. Moreover, the letters (ٵ, ٶ, ٷ, and ٸ) and (ا, و, ۇ, and ى) are semantically different. Although a computer system can employ this method for correct editing and display, it cannot distinguish the semantic difference between the original words and the transformed ones. Therefore, serious problems occur when a text-sorting function is required. For example, Kazakh alphabetical order dictates that the word باتىلدىق (meaning "courage") should be sorted ahead of the word بايتەرەك (meaning "big tree"). However, the word بايتەرەك (meaning: big tree) will be sorted ahead of the word باتىلدىق (meaning: courage) when the letter substitution method is used (see Table 2).

Table 2. Sorting when the Letter Substitution Method is used

	Correct ordering	The ordering in the letter substitution method
1	ب+ا+ت+ى+ل+د+ى+ق=باتىلدىق(courage)	ب+ا+ي+ت+ە+ر+ە+ك=بايتەرەك (big tree)
2	ب+ٵ+ي+ت+ە+ر+ە+ك=بايتەرەك (big tree)	ب+ا+ت+ى+ل+د+ى+ق=باتىلدىق(courage)

Moreover, the letter substitution method will not work in software applications such as script conversion and speech synthesis that require semantic understanding. For example, the word ٶ+م+ٸ+ر=ٶمىر (meaning "life") should be spelled as ٴ+و+م+ى+ر=ٶمىر according to the letter substitution method. A computer cannot convert this word into the Cyrillic alphabet form via letter correspondence because it cannot determine whether the character و represents the letter ٶ or و or whether the character ى represents the letter ٸ or ى. For the same reason, a computer also cannot produce the word ٶ+م+ٸ+ر=ٶمىر (meaning "life") in synthesized speech.

4. The Compromise Method for Processing the Letters ٵ, ٶ, ٷ, and ٸ

Upon analysis, we found that an OS can correctly process the four special letters in accordance with their writing rules if the OS has the capability of performing the following four operations:

A. Determine whether the letter is adjacent to other Kazakh letters, and select the glyph with ٴ of the isolated form to display when it is not adjacent to other Kazakh letters.

B. If the letter is adjacent to other Kazakh letters, then select its presentation form according to the letter cursive rules defined in Unicode.

C. Select the presentation form of the glyph to display, either with ٴ or not, when any of the letters گ, ك or ە appear in the word.

D. Move the symbol ٴ to the beginning of the word if a glyph with ٴ has been selected and the letter ٵ, ٶ, ٷ or ٸ is not at the beginning of the word.

Processing these four special letters requires executing different operations depending on the position of the letter in the word and the selected glyph, as shown in Table 3. A mainstream OS will typically process glyph layouts using the glyph layout features of the OpenType font format. We analyzed all the glyph layout features of the OpenType font format and found that the feature <calt> can assist an OS in determining whether a letter is adjacent to Kazakh letters (operation A), whereas the features <isol>, <init>, <medi>, <fina>, and <rlig> can help the OS to select the presentation form in accordance with the letter cursive rules defined in Unicode (operation B) [12]. However, we did not find any glyph layout feature that can assist the OS in determining whether any of the letters گ, ك or ە appears in the word (operation C) or in moving the symbol ٴ to the beginning of the word (operation D).

Table 3. The Operations Needed to Process the Letters ٵ, ٶ, ٷ, and ٸ[2]

Glyph \ Position		Not adjacent to other Kazakh letters	Beginning of word	Middle of word	End of word
Isolated form	With ٴ	A	B, C		
	Without ٴ		B, C		
Initial form	With ٴ		B, C	B, C, D	
	Without ٴ		B, C	B, C	
Medial form	With ٴ			B, C, D	
	Without ٴ			B, C	
Final form	With ٴ			B, C, D	B, C, D
	Without ٴ			B, C	B, C

To solve the problem posed by the fact that no glyph layout features exist to help an OS to perform operations C and D, we propose a compromise method that consists of the following three rules:

The first rule: Each of the letters ٵ, ٶ, ٷ, and ٸ should be represented by the combination of its own coded character and the coded character for the symbol ٴ. For example, the word ب+ٸ+ز=ٴبىز (meaning: us) should be spelled as ٴ+ب+ٸ+ز=ٴبىز.

The second rule: The glyphs with ٴ of the initial form, medial form, and final form for the letters ٵ, ٶ, ٷ, and ٸ should not be included in the OpenType font for Kazakh.

The third rule: The glyphs with ٴ of the isolated form should only be used when the four special letters are not adjacent to Kazakh letters.

According to the first rule, users who are editing Kazakh text should input the coded character ٴ (coded: 0674) at the beginning of the word when they believe that the glyph with ٴ is needed based on context. The key concept motivating this first rule is that the

[2] A blank cell in Table 3 indicates that the presentation form corresponding to the indicated row cannot be used at the position in a word indicated by the column according to the letter cursive rules.

determination of whether any of the letters گ, ك, or ە appears in the word (operation C) can be left to human users instead of the computer.

According to the first rule, the symbol ٴ of the glyph with ٴ is still displayed at the upper right corner of the glyph. This can cause a display error in which the symbol ٴ will still be displayed in the middle of the word (see Figure 5). This display error can occur only when one of the four special letters appears in the middle or at the end of a word. However, only the initial form, medial form or final form can be used in the middle or at the end of a word, according to the letter cursive rules. Thus, we propose the second rule of the compromise method to prevent such display errors. According to the second rule, among all glyphs for the initial, medial, and final forms of the four special letters, only the glyphs without ٴ are included in the OpenType font. Thus, the OS can directly use the glyph without ٴ for display when these three presentation forms are needed (see Figure 5), and operation D is no longer needed because no symbol ٴ needs to be moved.

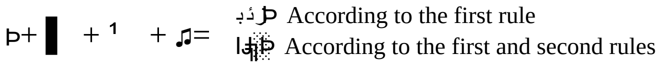
According to the first rule
According to the first and second rules

Figure 5. The Display of the Kazakh Word ٴبىز According To the Compromise Method

It is important to note that not all glyphs with ٴ for the four special letters are excluded from the OpenType font for Kazakh according to the second rule. This is because doing so would result in the glyphs for the letters (اٴ, ٴو, ۇٴ, and ٴى) and (ا, و, ۇ, and ى) being identical, preventing users from distinguishing between the two sets of letters during editing (see Table 1).

The glyph with ٴ of the isolated form can be used in only two contexts, according to the writing rules for the four special letters. One is when the letter is at the beginning of a word, and the other is when the letter is not adjacent to another Kazakh letter. The glyph without ٴ can also be used at the beginning of a word according to the writing rules for the four special letters. Which glyph, *i.e.*, with or without ٴ, should be used at the beginning of a word is determined by whether any of the letters گ, ك, or ە appears in the word (operation C). However, an OS cannot perform operation C even with the assistance of the OpenType font format. Therefore, the OS cannot decide which glyph of the isolated form should be used at the beginning of a word. Thus, we propose the third rule of the compromise method. This third rule can be adopted because the user will input the coded character ٴ at the beginning of the word when the glyph with ٴ of the isolated form is needed according to the writing rules for the four special letters.

The proposed compromise method conforms well to the Unicode standard because the coded characters used to present the four special letters and the symbol ٴ are identical to their definitions in Unicode. Therefore, this approach allows a computer to distinguish between the letters (اٴ, ٴو, ۇٴ, and ٴى) and (ا, و, ۇ, and ى). It should be noted that this compromise method does not affect the research and development of software applications that require semantic understanding, such as script conversion and speech synthesis. It does prevent the sorting errors caused by letter substitution, however (see Table 4).

Table 4. Sorting Using the Compromise Method

	Correct ordering	The ordering in the compromise method
1	ب+ا+ت+ى+ل+د+ى+ق=باتىلدىق(courage)	ب+ا+ت+ى+ل+د+ى+ق=باتىلدىق(courage)
2	ب+اٴ+ي+ت+ە+ر+ە+ك=بايتەرەك (big tree)	ب+اٴ+ي+ت+ە+ر+ە+ك=بايتەرەك (big tree)

In the compromise method, some coded characters for presentation forms, namely, those that correspond to the glyphs with ٴ defined in GB 21669, are not used. However, application compatibility is only minimally affected, because the coded characters for these presentation forms are used only to display letters, not to store or transfer letters.

5. Setting of the Glyph Layout Features

It is vital for the application of the compromise method that the glyph layout features in the OpenType font be set correctly. Each feature is defined by several fields, and all features have the same fields. There are six such features (<init>, <medi>, <fina>, <isol>, <rlig>, and <calt>), and their three fields (DIRECTION, SUBSTITUTION, and CONTEXT) must be set when an OpenType font is created in accordance with the compromise method.

Each feature has different functions. The features <isol>, <init>, <medi>, and <fina> are used to replace letter glyphs with presentation-form glyphs. The feature <rlig> is used to replace adjacent glyphs with ligature glyphs. The feature <calt> is used to set context-dependent glyph replacements. These features are executed sequentially by the OS. Each feature will be executed in turn based on the results from the previous feature. To obtain the results we desire, the features <isol>, <init>, <medi>, and <fina> should be executed first, and then <rlig> should be executed, followed by <calt>.

Each field also has a different function. The DIRECTION field is used to set the text direction. The value "RTL" (right-to-left) should be set for all six features. The SUBSTITUTION field is used to set the correspondence between the original glyphs and the replacement glyphs. Different values should be set in the SUBSTITUTION fields of the six features depending on their functions. The CONTEXT field is used to set the context for the execution of glyph replacement in the <calt> feature only.

The entire set of Kazakh letter glyphs should be listed as the original glyphs in the SUBSTITUTION field for the features <isol>, <init>, <medi>, and <fina>. The glyphs of the presentation forms, including the isolated, initial, medial and final forms, should be set as the replacement glyphs for the letter glyphs in the SUBSTITUTION field in accordance with the respective functions of the four features. It should be noted that the glyphs with ٴ, including ٴا, ٴو, ٴۋ, and ٴى, must be set as the replacement glyphs for the letter glyphs for the four special letters in the <isol> feature.

When the letter ل appears ahead of the letter ا or ٴا, Kazakh writing rules dictate that the two adjacent letters should be represented by their corresponding ligature لا, ٴلا, ﻼ or ٴﻼ. Thus, all possible adjacent glyph combinations of the letters 'ل and ا' or 'ل and ٴا' should be listed as the original glyphs in the SUBSTITUTION field of the <rlig> feature, and their ligature glyphs should be set as the replacement glyphs. The glyphs ا, و, ۋ, and ى should be set as the replacement glyphs for the glyphs ٴا, ٴو, ٴۋ, and ٴىin the SUBSTITUTION field of the <calt> feature. The CONTEXT field should list all glyphs of the presentation forms of the entire set of Kazakh letters for the <calt> feature.

With the above settings for the glyph layout features, an OS will execute the following operations when displaying Kazakh letters. First, replace letter glyphs with the presentation-form glyphs according to the settings of the features <isol>, <init>, <medi>, or <fina>. Second, replace adjacent glyph combinations of the letters 'ل and ا' or 'ل and ٴا' with the corresponding ligature glyph لا or ﻼ. Finally, replace the glyphs ٴا, ٴو, ٴۋ, and ٴى with the glyphs ا, و, ۋ, and ى when these four special letters are adjacent to any Kazakh letter.

6. Conclusion

In this paper, we propose a compromise method for handling the special Kazakh letters اٴ, وٴ, ۇٴ, and ىٴ. The proposed method avoids most of the shortcomings of the letter substitution method. This method conforms well to the Unicode standard, helps a computer to distinguish between the letters (اٴ, وٴ, ۇٴ, and ىٴ) and (ا, و, ۇ, and ى), helps in sorting Kazakh text correctly, and should facilitate the research and development of software applications that require semantic understanding, such as script conversion and speech synthesis. Some coded characters for presentation forms that are defined in GB 21669 are not used in this compromise method; however, application compatibility is only minimally affected because the coded characters for these presentation forms are used only to display letters, not to store or transfer letters.

To verify the feasibility of the compromise method, a few Kazakh OpenType fonts were designed and implemented. The Xinjiang Software Testing Center has accepted the compromise method proposed here and has recommended it to software companies for the development of software applications related to Kazakh language processing.

Acknowledgments

We gratefully acknowledge Ms. Jing Zhang who confirmed the function of glyphs substitution rules. This work was supported by the Xinjiang Province Outstanding Youth Talent Project (No. QN2015JQ009), the Key Technical Personnel Project of Chinese Academy of Sciences (No. Y629431201), the Equipment Function Development Technology Innovation Project of the Chinese Academy of Sciences (No.YBXM-2014-04) and Western Light Talent Culture Project of the Chinese Academy of Sciences (No.YG2012114).

References

[1] "National Bureau of Statistics of the People's Republic of China", The sixth census data of China. http://www.stats.gov.cn/tjsj/, **(2010)**.

[2] M. Eli, T. Qingfu and Y. Imin, "The study on the Uygur, Kazak, Kirgiz font style standard", Lang. Transl., vol. 4, **(2005)**, pp. 51-54.

[3] M. Niyazibek and K. Talp, "The Kazakh information processing and its prospects", J. Chin. Inf., vol. 24 **(2010)**, pp. 111-113.

[4] M. Niyazibek and K. Talp, "The discussion of several issues in the Kazakh information processing situation", Intell. Comp. Appl., vol. 1, **(2011)**, pp. 45-46.

[5] "Unicode 8.0.0 character code charts", Arabic. http://www.unicode.org/charts/PDF/U0600.pdf, **(2014)**.

[6] "Unicode bidirectional algorithm", http://www.unicode.org/reports/tr9/tr9-31.html, **(2014)**.

[7] "The unicode standard version 8.0.0 core specification", Middle East-I modern and liturgical scripts. http://www.unicode.org/versions/Unicode8.0.0/ch09.pdf, **(2015)**.

[8] "Unicode 8.0.0 character code charts", Arabic presentation forms-A. http://www.unicode.org/charts/PDF/UFB50.pdf, **(2015)**.

[9] "Unicode 8.0.0 character code charts", Arabic presentation forms-B. http://www.unicode.org/charts/PDF/UFE70.pdf, **(2015)**.

[10] "GB 21669, information technology - Uyghur, Kazak, Kirghiz coded character set", China Standard Press, Beijing, **(2008)**.

[11] "Microsoft Typography Home, OpenType", http://www.microsoft.com/typography/WhatIsOpenType.mspx, **(2009)**.

[12] "Microsoft Typography Home, OpenType registered features", http://www.microsoft.com/typography/otspec/features_ae.htm, **(2008)**.

Authors

Tonghai Jiang, he is a researcher of Technical Institute of Physics & Chemistry Chinese Academy of Sciences (XJIPC), born in 1963, Master Graduate Education. Mainly engaged in research and development of computer application technology.

Achieved results in more than 80 computer application projects, especially in computer automatic control, computer information network and computer information management. Published more than 30 papers in domestic and foreign academic journals . Developed the "Hospital Information Management System", which has been applied in more than40 hospitals in Xinjiang. Undertook the project "research of critical technique in application system of comprehensive resource based on domestic Linux", which has been used in Xinjiang government and many other enterprises and got first prize for "Xinjiang scientific and technology progress". Presided over many high-tech plans, such as "modern logistics management system in E-business environment", and "Office suite R&D in language of Chinese，Uyghur, Kazakh and Khalkhas" , all of which have been identified and brought good economic and social benefits. Participated in the "western light" training project in 1996 . Awarded as "outstanding expert in Xinjiang" in 2004.

His research team won the second prize of National Scientific and Technological Progress Award in 2014 due to their outstanding work on key technology research and application of Uyghur, Kazakh and Khalkhas software development. E-mail: jth@ms.xjb.ac.cn

Jun Dong, he was born in 1975, Technical Institute of Physics & Chemistry Chinese Academy of Sciences (XJIPC), Master Graduate Education. Mainly engaged in research and development of computer application technology.

He presided more than 8 computer application projects, especially in computer automatic control, computer information network and computer information management . All of the projects was financially supported by National Natural Science Foundation of China, West Light Foundation of Chinese Academy of Sciences(CAS) and Xinjiang Province Science Foundation for Youths. Published more than 7 papers in domestic and foreign academic journals. His team developed nine Uyghur, Kazakh and Khalkhas information technology standards and 8 multilingual software copyright of China.

As core research member, he undertook the project "Office suite R&D in language of Uyghur, Kazakh and Khalkhas" and "Development and application of multilingual (Uyghur, Kazakh and Khalkhas, Arabic, Russian and Chinese) software test platform" which got first prize for "Xinjiang scientific and technology progress" in 2008 and 2009. All of the projects have been identified and brought good economic and social benefits.

Due to his excellent work on key technology research and application of Uyghur, Kazakh and Khalkhas software development, he was awarded as "High-level faculty engineering of Tianshan mountain of Xinjiang Uyghur autonomous region”in 2012， “Outstanding Youth Talent of Xinjiang Uyghur autonomous region”in 2015 and“Key Technical Personnel Project of Chinese Academy of Sciences” in 2015. He also won the second prize of National Scientific and Technological Progress Award in 2014 together with his tutor Tonghai JIANG. E-mail: dongjun@ms.xjb.ac.cn

Li Cheng, he is a researcher of Technical Institute of Physics & Chemistry Chinese Academy of Sciences (XJIPC), born in 1973, obtained his doctorate in computer science from University of Kansas, USA. Mainly engaged in research and development of computer science.

He has been holding positions as senior software developer and researcher at the University of Kansas, where he designed and developed cloud-based Web applications and online assessment systems that served millions of K-12 and college students. He has been holding positions as senior software architect and researcher at U.S. DoD Biotechnology High Performance Software Institute, where he undertook dozens of research projects funded by DoD, U.S. Army, and NIH. He has developed algorithms and software pipelines for various bioinformatics applications, such as genome sequencing, protein interaction analysis, and computational drug discovery. His research outcomes have been published in high quality journals such as PLOS ONE and BMC Bioinformatics. He was selected as a “1000 talent plan” expert and joined Chinese Academy of Sciences (CAS) as a full professor in 2013. He is currently serving as a committee member of the Chinese language processing society in Chinese Computer Federation.

His research interests include data mining, machine learning, intelligent systems, natural language processing and bioinformatics. E-mail: chengli@ms.xjb.ac.cn

Yong Yang, he is an Associate Professor of College of Computer Science, Xinjiang Normal University, born in 1979, doctoral degree, mainly engaged in the research of natural language processing.

He presided over 2 item Xinjiang Uyghur Autonomous Region Natural Science Foundation Projects in the field of natural language processing, the ministry of education humanities and social science fund projects 2 item, The national language research planning projects 1 item, as the main members to participate in the national natural science fund 2 item Published more than 14 papers in domestic and foreign academic journals. Got two software copyrights.

His papers "the Comparative Study on Feature Selection in Uyghur Text Categorization" and " Research on aspects of statistical machine translation between Chinese and Uyghur” have won the second prize of Xinjiang uyghur autonomous region’s natural science excellent thesis in 2013 E-mail: yangyong1900@163.com