

देवनागरी लिपि एवं संगणक

अक्षर भारती

अम्बा कुलकर्णी

विभागाध्यक्ष

संस्कृत – अध्ययन विभाग

हैदराबाद विश्वविद्यालय

हैदराबाद

apksh@uohyd.ernet.in

पृष्ठभूमी:

संगणक और अंग्रेजी का एक अटूट सा नाता बन गया था जिससे लोगों के मन में यह भावना उत्पन्न हो गयी थी कि यदि संगणक पर काम करना है तो अंग्रेजी आना जरूरी है. यह भावना लोगों में संगणक के अविष्कार के बाद, ३ दशक तक बनी रही. यद्यपि देवनागरी लिपि का प्रयोग संगणक पर करने के लिए कई प्रयत्न हो रहे थे, संगणक के अस्तित्व के लगभग तीन दशक होने पर भी उस पर देवनागरी लिपि का प्रयोग संभव नहीं हो सका था. उसके बाद देवनागरी लिपि का प्रयोग तो संभव हुआ परन्तु अक्षर बहुत ही भद्दे से दिखते थे, मात्राएँ आगे पीछे होती थी. फिर एक समय ऐसा आया, जब संगणक पर देवनागरी लिपि के अक्षर सुन्दर दिखने लगे, एक से अधिक फॉन्ट्स उपलब्ध हुए, DTP का काम आम आदमी भी बिलकुल आसानी से कर पाने लगा, फिर भी एक गंभीर समस्या थी. अंग्रेजी की तरह हिन्दी में एक मशीन पर तैयार किया हुआ document दूसरी किसी मशीन पर पढना संभव नहीं था, बशर्ते दूसरे के पास भी वही साफ्ट्वेयर और फान्ट्स मौजूद हों जो पहले के पास है. युनिकोड के आने से यह समस्या भी दूर हो गयी और अब कोई भी आम आदमी अंग्रेजी की तरह ही हिन्दी में बिना कोई परेशानी काम कर सकता है!

देवनागरी लिपि को लेकर इन सारी कठिनाईयों का जो सामना करना पडा, और साफ्ट्वेयर इंजिनीयर्स को हल ढूँढने पडे, उससे मन में यह प्रश्न उठता है कि क्या देवनागरी लिपि इतनी क्लिष्ट है कि संगणक के लिए हल ढूँढने में इतना अधिक समय लगे? क्या

यह सचमें युनिकोड के आने से संभव हुआ या और कोई वजह है इसके सफलता की?

इस लेख में हम इन प्रश्न का उत्तर खोजने का / जानने का प्रयास करेंगे. साथ ही में हम देवनागरी लिपि की विशेषताएँ, देवनागरी लिपि को संगणक में उपलब्ध कराने के शास्त्रीय तरीका, युनिकोड एवम् देवनागरी के बारे में विशेष जानकारी देने का प्रयास करेंगे.

देवनागरी लिपि की विशेषताएँ:

यद्यपि भारतीय भाषाएँ भिन्न परिवार से जुडी हुई हैंx, फिर भी यह सर्वज्ञात है कि भारतीय लिपियाँ (Perso-Arabic को छोड़कर) का मुल स्रोत है ब्राह्मी लिपि.

इसलिए हम पायेंगे कि सभी भारतीय लिपियों के वर्ण लगभग समान है (कुछ अपवादों को छोड़कर). अक्षर लिखने के ढंग इनमें एक साम्य है और इसीलिए जब कोई संस्कृत भाषा का साहित्य दूसरी लिपि में लिखता है तो उसमें कहीं भी विकृति नहीं होती. इसी कारण हम यह प्रतिपादन कर सकते हैं कि, यह विशेषताएँ देवनागरी तक ही सीमित नहीं है बल्कि वे विशेषताएँ सभी भारतीय लिपियों की भी हैं.

देवनागरी लिपि के मूल वर्ण इस प्रकार है:

व्यंजन:

क् ख् ग् घ् ङ्
च् छ् ज् झ् ञ्
ट् ट् ड् ढ् ण्
त् थ् द् ध् न्
प् फ् ब् भ् म्
य् र् ल् व् श् ष् स् ह्

स्वर: अ आ इ ई उ ऊ ऋ ॠ ए ऐ ओ औ अं अः

इनमें से ऋ एवम् ॠ का प्रयोग अत्यंत कम मात्रा में होता है.

देवनागरी लिपि मूलतः अक्षरात्मक है. स्वरयुक्त व्यंजनों का समुह याने अक्षर. अक्षर में व्यंजन का होना जरूरी नहीं है. इस तरह से अ, क्ष, र, त्र्यं, स्त्री. यह सब अलग अलग अक्षर हैं. इनमें "स्त्री" अक्षर में "ी" है. इस चिह्न को मात्रा कहते हैं. अब प्रश्न यह है, रोमन या अन्य लिपियों में भी स्वर और व्यंजन तो होते हैं. परन्तु मात्रा की परिकल्पना तो केवल देवनागरी लिपि में ही (या भारतीय लिपियों में ही) मिलती है. यह "मात्रा" है क्या? देवनागरी में इसकी जरूरत क्यों पड़ी? इन प्रश्नों के उत्तर हमें मदद करेंगे अपनी लिपि को अच्छी तरह से समझने में.

यह बात सर्व विदित है कि

हम जानते हैं कि

क = क् + अ

का = क् + आ

कि = क् + इ

क्ष = क् + ष् + अ

त्रि = त् + र् + इ

ट्र = ट् + र + अ

त्र्य = त् + र् + य् + अ

इन समीकरणों के बायीं ओर है अक्षर और दाहिनी ओर उसी अक्षर में स्थित वर्ण उनके क्रम के अनुसार. यह स्पष्ट है कि दाहिनी ओर के वर्णों के समुदाय को बायीं ओर स्थित अक्षरों से कई गुना अधिक जगह लगती है. अक्षरात्मक पद्धति एक तरह से लघु-लेखन का एक तरीका है और देखने में भी इकाई के रूप में सभी व्यंजन के साथ दिखाई देते हैं. अल्प जगह में अधिक वर्णों को शामिल करके लिखने का यह ढंग Perso Arabic लिपि में उच्चतम कोटि को पहुँचा हुआ हम पायेंगे.

देवनागरी लिपि में अक्षर लिखने के तरीके की ओर यदि हम गौर करेंगे तो हमें लिपि की निम्न विशेषताएँ नजर आएंगी.

१. व्यंजनों के उच्चारण के लिए स्वरों की जरूरत पडती है. जैसा कि

क = क् + अ

याने, 'क' का चिह्न व्यंजन 'क्' सहित स्वर 'अ' को दर्शाता है. देवनागरी के सभी व्यंजनों के चिह्न 'अ' - स्वरसहित है. 'अ' स्वर

का उपयोग अन्य स्वरों की तुलना में अधिक होता है इसलिए शायद व्यंजनों के चिह्नों को 'अ'-सहित माना गया. यदि स्वर रहित व्यंजन को लिखित रूप से व्यक्त करना है, तो व्यंजन में अंतर्निहित अ को हटाना होगा, यह हटाने का काम करता है 'क्' चिह्न, जो कि हलन्त नाम से जाना जाता है. इस तरह 'क्' का अर्थ है स्वर-रहित व्यंजन 'क'. यदि दूसरे किसी स्वर को व्यंजन के साथ जोड़ना है तो पहले व्यंजन में अंतर्निहित-'अ' को हटाकर बाद में उसमें इच्छित स्वर को जोड़ना होगा.

उदाहारणार्थः

$$क + ि = क - अ + इ$$

इसी को concise तरीके से लिखने के लिए शायद 'मात्रा' की परीकल्पना की गयी हो.

उदाहारणार्थः

$$क - अ + इ = कि$$

हिन्दी में १२ स्वर है जो प्रचलित हैं. इन स्वरों के लिए अलग अलग चिह्नों का उपयोग किया जाता है और इन चिह्नों को दशानि के लिए व्यंजन के स्थान के चारों ओर - उपर, नीचे, बायें और दाहिने का प्रयोग किया जाता है. (देखें figure-1)

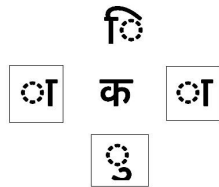


figure - 1

स्वर और स्वर की मात्रा में फिर सम्बन्ध इस तरह होगा

$$कि = क + ि = क् + इ$$

और

$$ि = ् + इ.$$

इस तरह से, हम यह कह सकते हैं कि,

स्वर की मात्रा = अ-स्वर को घटाकर, इच्छित स्वर का जोड़.

२. जब अक्षरों में एक से अधिक व्यंजन होते हैं तब इन व्यंजनों का क्रम बताना जरूरी है. देवनागरी लिपि में ये व्यंजन या तो उपर से नीचे (उदा. द्र) या बाएँ से दाएँ (उदा. क्या) लिखे जाते हैं. एक से अधिक व्यंजन वाले अक्षरों में स्वर की मात्रा, एक व्यंजन वाले अक्षरों की तरह ही व्यंजन समूह के बाएँ, दाएँ, उपर या नीचे होती है.

'बुद्धि' इस शब्द के 'द्धि' अक्षर में, हमें प्रथम इ-स्वर की मात्रा, उसके बाद 'द' अक्षर जो कि पूर्ण दिखता है और ध आधा दिखता है. परन्तु उसके उच्चारण में द् व्यंजन के बाद ध् व्यंजन और उसके बाद इ स्वर यह क्रम है. यह उपर दिए हुए नियमों के अनुसार ही है. यद्यपि मात्रा पहले दिखती है, फिर भी अक्षर की परिभाषा को स्वीकार करने पर, स्वर का उच्चारण व्यंजनों के उच्चारण के बाद ही होगा. वैसे ही यदि व्यंजनों का उच्चारण भी उनके क्रमानुसार ही होगा.

देवनागरी लिपि का संगणक पर आविष्कार:

संगणक पर अंग्रेजी documents देखने में हमें कभी भी कोई भी कठिनाई नहीं हुई, क्यों कि प्रत्येक अंग्रेजी वर्ण के लिए एक मानक कोड(ASCII) प्रदान किया हुआ था और सभी जगह इसका पालन होता था. इसी कारण फान्ट, या मशिन, या editor बदलने पर भी अंग्रेजी text के पढ़ने में कोई असुविधा नहीं होती, वैसे ही यदि किसी वेब पेज पर कोई "hello" लिखा हुआ पन्ना ढुँढ़ेगा तो "hello" चाहे किसी भी फान्ट में लिखा हुआ हो, मशिन उसको ढुँढ़ के लाती है.

जैसे रोमन लिपि के लिए ASCII मानक प्रचलित था, वैसे ही देवनागरी लिपि के लिए ISCII नामक मानक अपनाया गया. परन्तु इस मानक के होते हुए भी लोगों ने इसका अनुसरण नहीं किया. संकेतांक देते समय उन्होंने न देवनागरी के वर्णात्मक आविष्कार को अपनाया न ही उसके अक्षरात्मक आविष्कार को. टाइपरायटर

में जैसे अक्षरों को तोड़कर लगभग ५० चिह्नों की सहायता से सभी देवनागरी अक्षर बनाये जाते थे, उसी प्रकार से संगणक पर भी लोगों ने टाइपरायटर के जैसे ही glyphs को आधार मानकर फान्ट्स बनाए. फान्ट्स बनाते समय अलग अलग फान्ट vendors का ध्यान अलग अलग चीजों पर था. कुछ लोगों को फान्ट्स दिखने में सुन्दर चाहिए थे तो कुछ के लिए अंग्रेजी कुंजिपटल से ही बिना किसी विशेष साफ्टवेअर की सहायता से देवनागरी का उपयोग करना अधिक महत्व रखता था. इस के कारण अलग अलग glyphs को संकेतांक दिये गये, न कि वर्ण या अक्षरों को. परन्तु प्रत्येक फान्ट vendors की अपनी अपनी प्राधान्यता होने के कारण, glyph कोड के लिए मानक होते हुए भी (CDAC द्वारा प्रमाणित) किसी के द्वारा उसका अनुसरण नहीं किया गया.

यदि साफ्टवेअर engineers या फान्ट vendors ने इस बात का ध्यान रखा होता कि देवनागरी लिपि में वर्णात्मक से अक्षरात्मक या अक्षरात्मक से वर्णात्मक में जाने का एक नियमबद्ध तरीका है, तो देवनागरी लिपि को लेकर जो एक अव्यवस्था थी उसके लिए एक अच्छा हल मिलता परन्तु देवनागरी लिपि की "आत्मा" की ओर ध्यान न देकर केवल उसका दृश्य रूप याने "शरीर" की ओर ध्यान देने से संगणक के जगत् में देवनागरी लिपि को लेकर एक अस्तव्यस्तता फैल गयी थी.

मानक होते हुए भी उनको न अपनाने के पीछे क्या समस्याएँ थी, इसके बारे में जानने के लिए पहले हम संगणक में लिपि का implementation कैसे किया जाता है इसके बारे में कुछ जानकारी लेंगे.

संगणक के कुंजिपटल से हम संगणक को inputs देते हैं. जब कुंजिपटल की कोई कुंजि (एक या एक से अधिक) दबाई जाती है तब उस कुंजि के अनुरूप एक संकेत संगणक के CPU को भेजा जाता है. यह संकेत हर कुंजि के लिए अलग अलग होता है. कुंजि के अनुरूप संकेत भेजने वाले साफ्टवेअर को keyboard driver कहते हैं. रोमन लिपि में केवल २६ वर्ण हैं और प्रत्येक वर्ण के लिए ASCII मानक के अनुसार एक संकेतांक है. कुंजिपटल की कुंजि दबाने से वह संकेतांक keyboard

driver द्वारा निर्माण संगणक के cpu को भेजा जाता है और संगणक का display driver उस संकेतांक के लिए जो फान्ट glyph है वह मानिटर पर प्रदर्शित करता है. रोमन लिपि वर्णात्मक होने के कारण इस पूरे काम में कोई कठिनाई नहीं होती.

यदि देवनागरी का वर्णात्मक रूप हम स्वीकार करते हैं तो ISCII मानक का उपयोग करके देवनागरी लिपि को संगणक पर उपलब्ध करा देना अंत्यत आसान काम है. उदाहरण के लिये, 'कमल' इस तरह से लिखा जाएगा 'क् अ म् अ ल् अ'. 'क्षत्रिय' इस तरह से लिखा जाएगा 'क् ष् अ त् र् इ य् अ'. यह स्वाभाविक है कि इस तरह से लिखने में, देवनागरी का 'सौंदर्य' नष्ट होगा. उसमें जो एक compactness है वह चला जाएगा.

देवनागरी अक्षरात्मक होने के बावजूद वर्णों से अक्षर लिखने का एक शास्त्रीय नियमबद्ध तरीका है. टंकलेकन यंत्र में उसकी अपनी स्मृति (memory) न होने के कारण, इस नियमबद्ध तरीके का उपयोग न हो सका. परन्तु संगणक के पास अपनी स्मृति है. वह कुंजिपटल की कौन कौन सी कुंजियों को किस क्रम में दबाया था उसे याद रख सकती है. इसलिए हम इसका लाभ उठाते हुए, कुंजि वर्णात्मक क्रम से दबाकर उससे अनुरूप अक्षर को संगणक के मानिटर पर दर्शा सकते हैं. जैसे कि 'त्र' लिखने के लिए 'त् र् अ' इन कुंजियों को हम दबाते हैं. Keyboard Driver(या Input Method Editor) संगणक को 'त् र् अ' के अनुरूप संकेतांक भेजता है, जो कि किसी मानक (जैसे कि ISCII या युनिकोड) पर आधारित है. बाद में Display Driver इन संकेतांक के क्रम को पहचान कर, फान्ट्स के कौन से glyph को मानिटर पर प्रदर्शित करना है इसका निर्णय लेता है, और उस glyph का संकेतांक CPU को भेजता है. इस तरह देवनागरी लिपि को दर्शाने के लिए keyboard एवं Display Drivers के माध्यम से और वर्णात्मक एवं अक्षरात्मक प्रणाली का उचित उपयोग करना होगा.

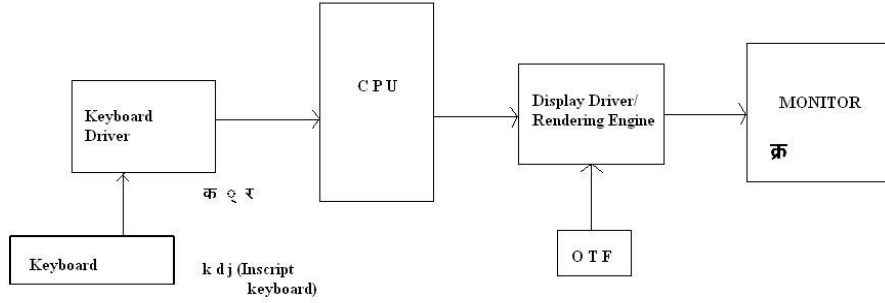


figure – 2

फान्ट्स – पुरानी समस्यायें एवं नए

हल:

पुराने True Type Fonts(TTF) में केवल 256 संकेतको के लिए ही जगह थी. उनमें से कुछ संकेतांक संगणक के control के लिए सुरक्षित/बचा के रखे थे और लिपि के लिए लगभग १९२ संकेतांक ही उपलब्ध होते थे. फान्ट बनाने वालों को केवल इन जगहों का उपयोग करके, glyphs निर्माण करना पड़ता ताकि उनके अलग अलग संयोग से देवनागरी के सभी अक्षर बन सके, और वे सुन्दर भी दिखें. यह काम बहुत ही मेहनतवाला एवं क्लिष्ट था. जैसे की उपर बताया जा चुका है, अलग अलग फान्ट vendors की अपनी अपनी प्राधान्यताएँ होने के कारण glyph के चिह्नों में अथवा उनके संकेतांकों में एक-मान्यता नहीं थी. इसी कारण प्रत्येक फान्ट के साथ यदि उसके अनुरूप लिखा हुआ display driver एवं keyboard driver नहीं है तो उन फान्ट्स का उपयोग कर पाना असम्भव था. CDAC,NCST,श्री लिपि जैसे फान्ट vendors ने फान्ट्स के साथ अपना पुरा साफ्टवेअर तैयार किया था जिसके keyboard और Display drivers महत्वपूर्ण अंश थे. ये अंश साफ्टवेअर का अविभक्त

अंश होने से अंग्रेजी के editors (Word, Wordpad, वगैरह) के साथ इन फान्ट्स को उपयोग करना असम्भव था. साथ में फान्ट्स के साथ उसका व्याकरण (किन किन glyphs के संयोग से कौन कौन से अक्षर बनते हैं इसके बारे में नियम) फान्ट्स के साथ न होने के कारण, दूसरों के लिये glyphs का अर्थ लगाना बहुत ही मुश्किल था.

Open Type Fonts(OTF) के आते ही इनमें से फान्ट के व्याकरण वाली समस्या हट गयी. OTF में कौन कौन से अक्षर किन किन glyphs के संयोग से बनते हैं इसके बारे में जानकारी उसी फान्ट्स के अंतर्गत लिखी होने से glyphs का अर्थ लगाना आसान हो गया, दूसरी ओर Display driver एवं Keyboard driver हर फान्ट के लिए अलग अलग लिखने की जरूरत नहीं रही. एक ही सामान्य keyboard driver एवं display drivers की सहायता से किसी भी साफ्टवेयर के साथ अब देवनागरी लिपि का प्रयोग करना सम्भव हुआ.

युनिकोड, UTF8 एवं देवनागरी:

युनिकोड ISCII-91 पर आधारित है. परन्तु ISCII-91 एवं युनिकोड में एक मूलभूत अंतर है. वह यह कि, ISCII-91 ब्राह्मी लिपि से उत्पन्न सभी भारतीय लिपियों के लिए एक मानक है. अर्थात् 'क' चाहे देवनागरी में लिखा हो या बंगला में या तेलुगु में, उसका सूचकांक एक ही होगा, लिपि के अनुसार बदलेगा नहीं. ISCII-91 में विविध लिपियों के लिए एक ही सूचकांक होने के कारण लिप्यांतरण बहुत ही आसान होता था. विभिन्नता में एकता थी. परन्तु युनिकोड में हर लिपि के लिए अलग अलग संकेत दिये गये हैं. इस के कारण लिप्यांतरण उतना आसान नहीं रहा, जितना ISCII-91 में था. लिपियों की विभिन्नता बनी रही. इससे भी और गम्भीर परिणाम UTF-8 के उपयोग से हुए हैं.

जब कि युनिकोड दुनियाभर की विविध लिपियों के वर्णों को संकेतांक प्रदान करना है, UTF-8 एक मशीन से दूसरी मशीन को डेटा ट्रान्सफर करने का एक प्रारूप है. इस प्रारूप के अनुसार, यदि देवनागरी के कोई भी वर्ण या अक्षर को ट्रान्सफर करना है तो, प्रथम उस वर्ण के संकेतांक जो कि युनिकोड के अनुसार 2

bytes का होता है उसको प्रथम 3 bytes में convert करते हैं और यह 3 byte का, नया संकेतांक एक मशीन से दूसरी मशीन को भेजा जाता है. इसका अर्थ हुआ यदि देवनागरी के १०० वर्ण वाला कोई matter ट्रान्सफर करना है तो $100 * 3$ याने 300 bytes की आवश्यकता होगी. वही यदि अंग्रेजी का 100 वर्णों का कोई matter ट्रान्सफर करना है तो केवल १०० bytes की ही जरूरत होगी. परिणामवश, ISCII की जगह युनिकोड को अपनाने से एवं UTF-8 प्रारूप default होने से हम न केवल एकता को खो बैठे, data transfer/storage में भी हमारा नुकसान हुआ है.

कुछ सुझाव:

अक्षर और वर्णों के बीच के सम्बन्धों को ध्यान में रखते हुए, और तकनिक का लाभ उठाते हुए हम अभी भी देवनागरी को उचित न्याय दे सकते हैं. जरूरत है इच्छा शक्ती की.

यह है कुछ सुझाव:

1. जैसे कि ISCII में सभी भारतीय भाषाओं के लिए एक ही कोड था वैसे ही सभी भारतीय भाषाओं के लिए युनिकोड में भी एक ही कोड हो, ताकि सभी भारतीय भाषाओं की एकता बनी रहे, और भारतीय भाषाओं में आपस में लिप्यांतरण आसान हो.
2. जिन अक्षरों का बारबार प्रयोग होता है उनको अलग संकेतांक दिया जाए ताकि उनके storage एवं transfer में कम bytes लगे.

यह संभव है क्योंकि वर्णात्मक से अक्षरात्मक एवं अक्षरात्मक से वर्णात्मक जाने के लिए नियमबद्ध तरीका है.

निष्कर्ष:

भारतीय लिपियों की विशेषता यह है कि वे रोमन जैसे वर्णात्मक लिपियों से अग्रेसर हैं. हलन्त के उपयोग से बहुत ही बुद्धिमान तरीके से अक्षर की संकल्पना की गई है, जिससे न केवल लिखने में compactness है बल्कि storage एवं data transfer में भी बचत हो सकती है. आवश्यकता है उपलब्ध तकनिक का योग्य उपयोग करने की.

संदर्भ:

Bharati, Akshar, Amba P Kulkarni, Vineet Chaitanya and Rajeev Sangal, "anuvada ke upakarana: sanganaka tathA bhaashaae (Tools of Translation: Computer and Languages), University of Hyderabad, Distance Education Programme, Hyderabad, Feb 1998.

Bharati, Akshar, Nisha Sangal, Vineet Chaitanya, Amba P Kulkarni, and Rajeev Sangal, "Generating Converters between Fonts Semi-automatically", Proc. of SAARC conference on Multi-lingual and Multi-media Information Technology, CDAC, Pune, 1-4 Sept. 1998b

Bharati, Akshar, Vineet Chaitanya, Amba P. Kulkarni, ISCII Plugin for displaying Indian language web pages (presented at Caturang, 5-6 March 2001, IUCAA, PUNE)

Himanshu Garg 'Overcoming the font and script Barriers among Indian Languages'
M. S. Thesis, IIIT, Hyderabad, India, March 2006