Unified Devanagari Rendering Engine for Nepali Language

Team Members

Amar Dura [THA077BCT007]
Ayush Bhandari [THA077BCT014]
Harish Joshi [THA077BCT018]
Sugam Pokharel [THA077BCT044]

Supervised By
Er. Praches Acharya

Co-supervised By
Er. Santa B. Basnet

Department of Electronics and Computer Engineering
Institute of Engineering
Thapathali Campus
Kathmandu, Nepal

PRESENTATION OUTLINE

- Motivation
- Objectives
- Scope of Project
- Project Applications
- Methodology
- Results
- Analysis of Results
- List of Remaining Tasks
- References

MOTIVATION

- Lack of open source tools for pdf rendering for Nepali language
- Inconsistency in composite character representation in different Devanagari fonts

  [अङ्क or अङ्क]

OBJECTIVE

- To implement glyph ordering mechanism and standardize composite character representation
- To develop an open source unified Devanagari rendering engine for JVM using Apache PDFBox

SCOPE OF PROJECT

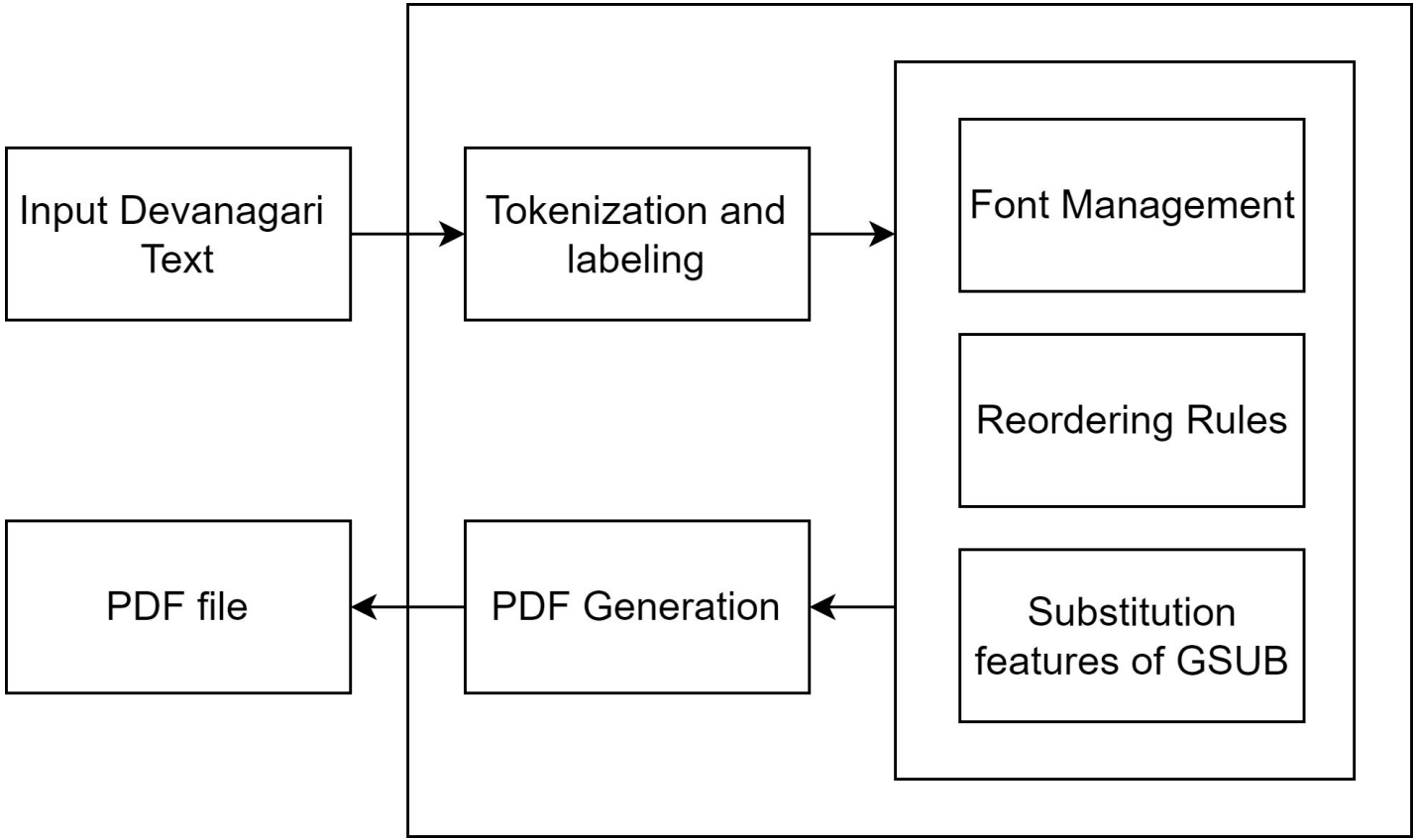
- Ensures the correct ordering of glyphs for Nepali text
- Focus on rendering on PDF file
- Only implemented for multi-byte Unicode fonts
- Apache PDFBox works on JVM only

PROJECT APPLICATIONS

- News compilation
- Financial reporting
- Educational materials
- Governmental documentation

METHODOLOGY - [1]

System Block Diagram



METHODOLOGY - [2]

Input Devanagari Text

- from text document, database or text repository.
- contains the unicode devanagari text

Example:

“भानुभक्तका हजुरबुवा श्रीकृष्ण आचार्य जुम्ला जिल्लाको सिन्जा उपत्यकाबाट तनहुँ जिल्लामा बसाइँ सरेका थिए।”

METHODOLOGY - [3]

Text Preprocessing

Character tokenization and labeling

- Each character is a token
- A character fall into a category/label
- Based on Unicode Standards

भानुभक्तका : भ ◌ा न ◌ु भ क ◌् त क ◌ा

Character	भ	◌ा	न	◌ु	भ	क	◌्	त	क	◌ा
Category	WC	DV	WC	DV	WC	WC	HALANTA	WC	WC	DV

METHODOLOGY - [4]

Syllable Tokenizer

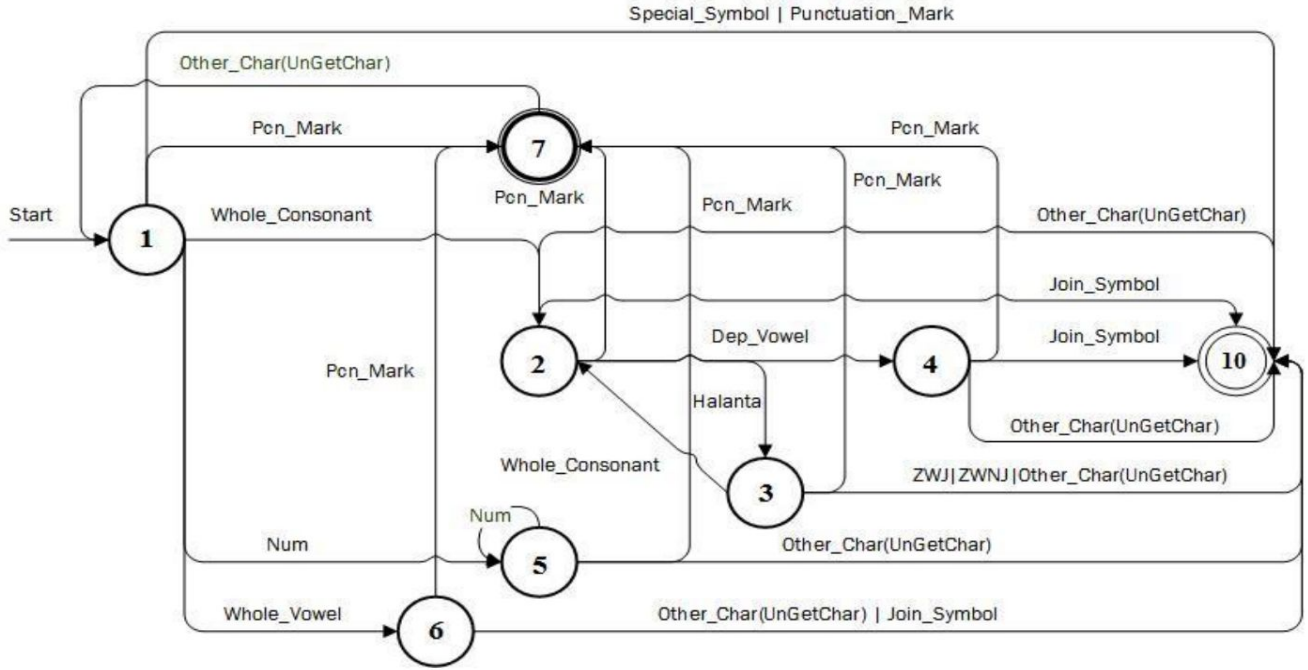


Fig: Finite State Diagram for Tokenizer

METHODOLOGY - [5]

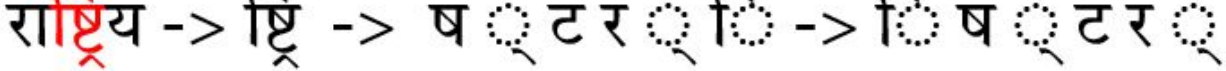
Font Management

- Provides different unicode devanagari fonts.
- OpenType font file provides the required glyphs, GSUB, GPOS table.
- e.g. Tiro Devanagari Sanskrit font, Mangal font

METHODOLOGY - [6]

Reordering

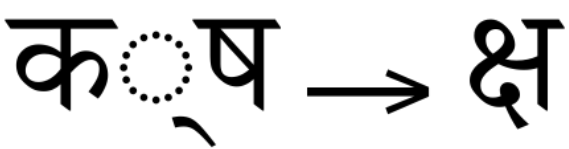
- All the glyphs are tagged with the intended location in the syllable.
- Tagged glyphs are sorted in the stable order.
- The ordered glyphs will have glyphs of same category on the same relative order.



METHODOLOGY - [7]

Basic Substitution

- Basic substitution is applied from GSUB table.
- The order of substitution is already defined.
- Substitutions include akhanda ligatures, reph with single half-consonants, etc.



METHODOLOGY - [8]

Glyph Substitution

GSUB table structure

Name	Example	Substituted glyphs
Akhanda	ज + ञ	ज्ञ
Reph	र + ◌् + क (consonant)	र्क
Rakaar	भ (consonant) + ◌् + र	भ्र
Half form	क (consonant) + ◌् + ख	क्ख

METHODOLOGY - [9]

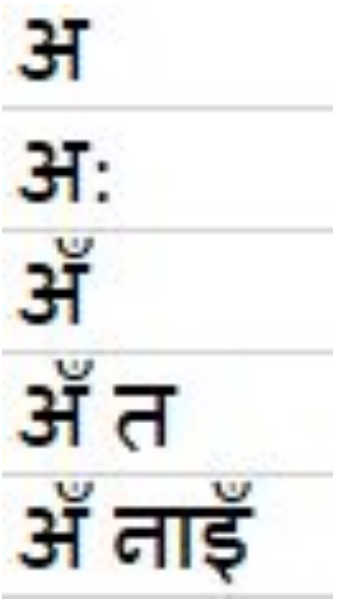
GPOS Application

- Manage the positions of glyphs correctly using GPOS table of the font.
- Mainly glyphs below and above the base.
- Dist and kern are used.

METHODOLOGY - [10]

Word List Preparation

- JSON format of Nepali Brihat Shabdakosh was taken.
- A word list of 1,23,371 words was collected.



METHODOLOGY - [11]

Testing and Validation

- Sequences with different length (uni, bi, tri, etc) will be generated from corpus.
- The generated sequences will be manually labelled.
- The output of the generated sequences against the label.
- Accuracy will be calculated for different length sequences.

Correctness measure

$$\text{Accuracy} = \frac{\text{Correctly ordered glyphs}}{\text{Total words}}$$

$$\text{Average accuracy} = \frac{\sum_{i=1}^{k} \text{Accuracy (Category)}_i}{k}$$

where, k = Number of categories

Categories by length of sequence: Uni, Bi, Tri, Quad etc.

RESULT

Tokenization

Tokens for शिक्षा: [श, ि, क,्, ष, ा]

Labeling: [WC, DV, WC, HALANTA, WC, DV]

ANALYSIS OF RESULT

Analysis of tokenizer

- Characters are tokenized and labelled based on predefined labels
- Syllable tokenizer groups the labelled characters using the finite state automata

Extraction of word list

- 1,23,371 words should be enough to prepare the required test sequences.

LIST OF REMAINING TASK

Test data preparation and labeling

- Manually reorder the characters of test sequences

Implementation of algorithm

- Actual coding in the Apache PDFBox codebase

System evaluation

- Different sub-groups will be made for groupwise test

REFERENCES - [1]

- Microsoft, "Microsoft Typography Documentation," Accessed: Jul. 19, 2024. [Online]. Available: https://learn.microsoft.com/en-us/typography/script-development/devanagari
- U. Consortium et al., “The unicode standard, version 14.0. 0–core specification,” 2021.
- S. B. Basnet and S. Trishna, “Unification of fonts encoding system of devanagari writing in nepali,” Nepalese Linguistics, vol. 32, no. 2, pp. 130–136, 2017.

REFERENCES - [2]

- M. Boualem, M. Leisher, and B. Ogden, “Encoding script-specific writing rules based on the unicode character set.”
- S. P. Mudur, N. Nayak, S. Shanbhag, and R. Joshi, “An architecture for the shaping of indic texts,” Computers & Graphics, vol. 23, no. 1, pp. 7–24, 1999.
- K. Nepal, "Nepali Font Standards," Kathmandu, Nepal.