# A Fast and Accurate Approach for Main Content Extraction based on Character Encoding

Hadi Mohammadzadeh, Thomas Gottron, Franz Schweiggert and Gholamreza Nakhaeizadeh

Ph.D. Candidate
Institute of Applied Information Processing
University of Ulm
Germany
hadi.mohammadzadeh@uni-ulm.de

Tir 2011, Augest 2011, Toulouse, France

# Outline

# Outline

# An example of web pages with selected MC

- What are the right to left (R2L) laguages?
  - Laguages which are written from the right side to the left side, Arabic, Persian, Urdu and Pshtoo
- What are our goals?
  - From a technical point of view, most of the main content extraction approaches use HTML tags to separate the main content from the extraneous items.
    This implies the need to employ a parser for the entire web pages. Consequently, the computation costs of these main content extraction(MCE) approaches are increased.
    Thus, the main goal of proposed algorithm is to increase both the accuracy and effectiveness of the MCE algorithms dealing with R2L languages

## Outline

- In UTF-8, ASCII characters need only one byte, with a value guaranteed to be less than 128
- In UTF-8, all letters of right to left languages (Persian , Arabic, Pashto, and Urdu Languages) take exactly 2 bytes, each with value greater than 127
- By using simple condition , we are able to seperate ASCII characters from Non-ASCII characters .
  - If the value of one byte is $< 128 \Rightarrow$ this byte is a member of ASCII characters
  - Otherwise it is a member of Non-ASCII characters

## Outline

## The Phases of DANA

- In the First phase, we count the number of ASCII and Non-ASCII characters of each line of the HTML file, saved in two 1D arrays T1 and T2
- In the Second phase, we are looking to find areas comprising the MC in an HTML file using the arrays T1 and T2
- Finally, we feed all HTML lines determined in the previous phase as an input to a parser. The output of parser shows the main content

## Phase One : Counting ASCII and Non-ASCII characters of each line

DANA reads an HTML file line by line and counts the number of ASCII and Non-ASCII characters of each line

- The number of ASCII characters of each line is saved in one-dimentioanl array T1
- The number of Non-ASCII characters of each line is saved in one-dimentioanl array T2

## Phase Two : Finding areas Comprising MC

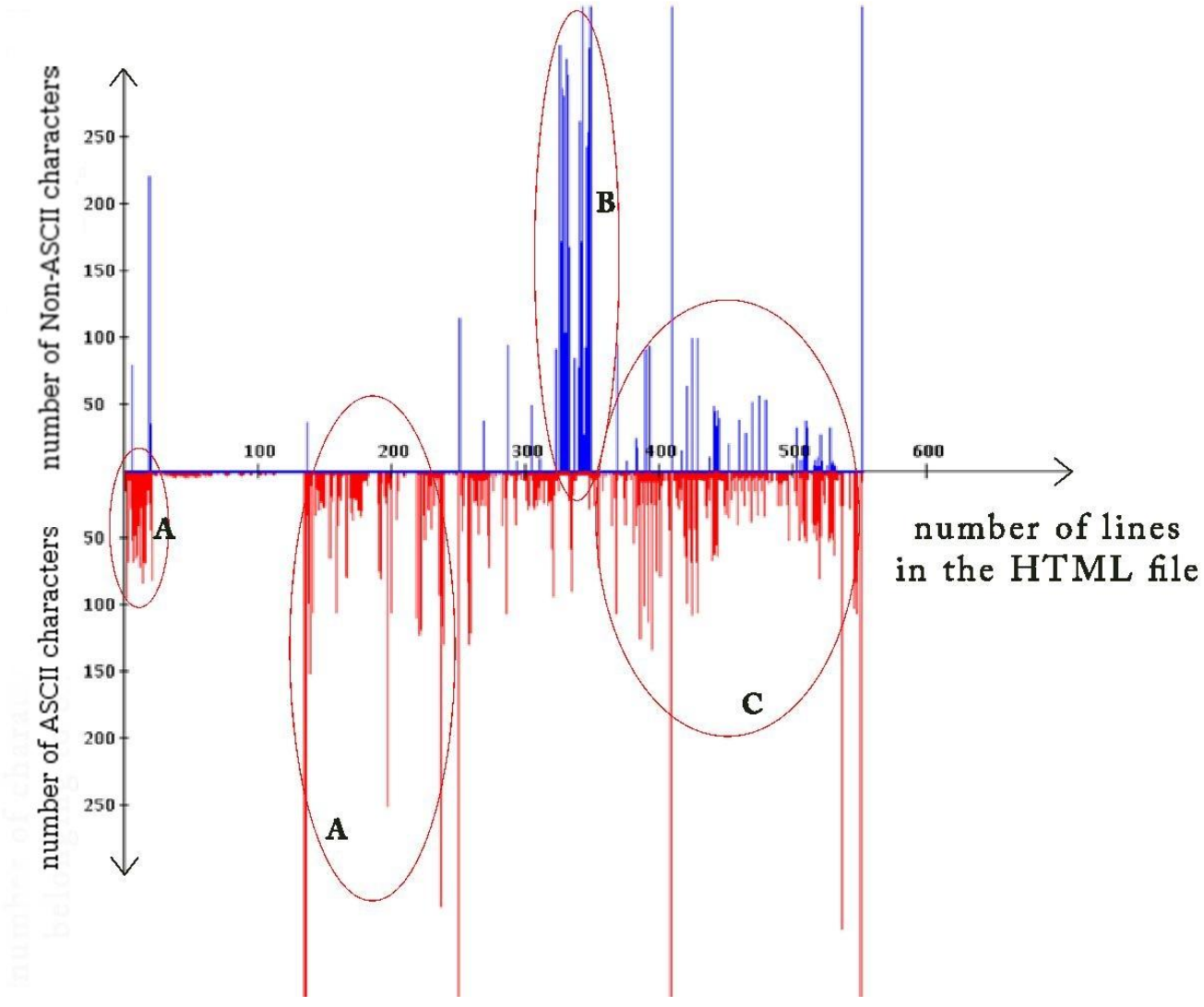
Recognizing areas in the HTML file, in which:

- Non-ASCII Characters have high density
- ASCII Characters have low density

To illustrate our approach, we depict two diagrams. In the first diagram, for each line of the HTML file

- For the Non-ASCII characters, a vertical line with identical length is drawn upside of the x-axis, as stored in T1
- Similarly, for the ASCII characters a vertical line with equal length is drawn downside of the x-axis, as stored in T2

# Phase Two : Finding areas Comprising MC



## Phase Two : Finding areas Comprising MC

Different types of regions:

- Regions with low or near zero density of columns above the x-axis and high density of columns below the x-axis, labeled A
- Region with high density of columns above the x-axis and a low density of columns below the x-axis, labelled B
- Regions with slight difference between the density of the columns above and below the x-axis, labelled C

## Phase Two : Finding areas Comprising MC

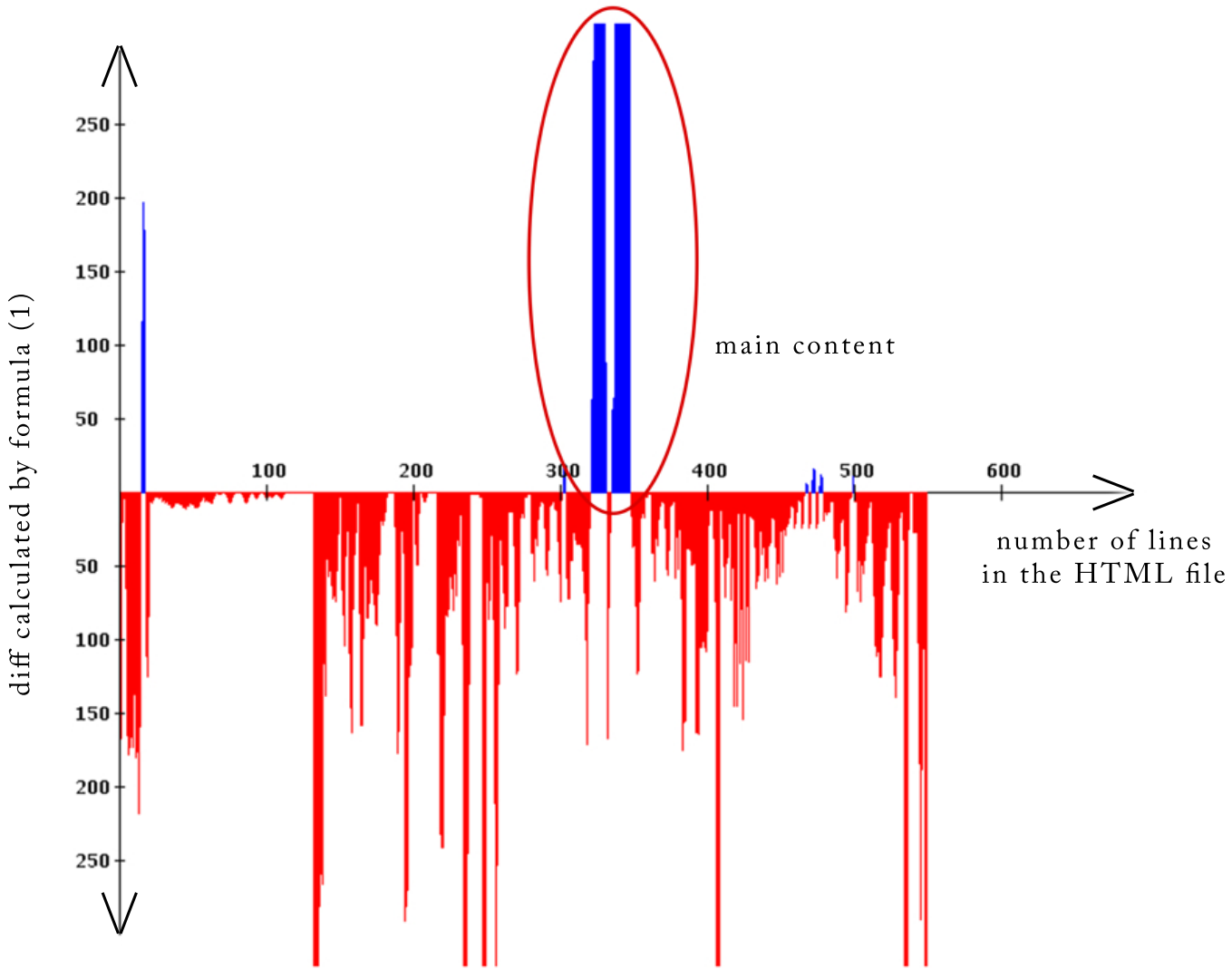
Now the problem of finding MC in the HTML file becomes the problem of finding region B. To find Region B we follow 3 steps:

- 1) Draw smoothed diagram. For all columns we calculate $\text{diff}_i$ and draw new diagram

$$\begin{aligned} \text{diff}_i &= \text{T1}_i - \text{T2}_i \\ &+ \text{T1}_{i+1} - \text{T2}_{i+1} \\ &+ \text{T1}_{i-1} - \text{T2}_{i-1} \end{aligned}$$

  - If $\text{diff}_i > 0$ then we draw a line with the length of $\text{diff}_i$ above the x-axix
  - Otherwise , we draw a line with the length of absolute value of $\text{diff}_i$ below the x-axix
- 2) We find a column with the longest length above the x-axis
- 3) Finding the boundaries of the MC region

# Phase Two : Finding areas Comprising MC



## Phase Two : Finding areas Comprising MC

Finding the boundaries of the MC regions, but how

- After recognizing the longest column above the x-axis, the algorithm moves up and down in the HTML file to find all paragraph belonging to the MC. But where is the end of these movements?
- The number of lines we need to traverse to find the next MC paragraph is defined as a parameter P, initialized with 20.
- By considering this parameter, we move up or down, respectively until we can not find a line containing Non-ASCII characters.
- At this moment, all lines we found make our MC.

## Phase Three : Extracting MC

In final phase,
we feed all HTML lines determined in the previous phase as an input to a parser.
Following our hypothesis, the output of the parser is exactly the main content.

# Data Sets

| Web site | Num. of Pages | Languages |
|---|---|---|
| BBC | 598 | Farsi |
| Hamshahri | 375 | Farsi |
| Jame Jam | 136 | Farsi |
| Ahram | 188 | Arabic |
| Reuters | 116 | Arabic |
| Embassy of Germany, Iran | 31 | Farsi |
| BBC | 234 | Urdu |
| BBC | 203 | Pashto |
| BBC | 252 | Arabic |
| Wiki | 33 | Farsi |

- Arabic, Farsi, Pashto, and Urdu
- We have collected 2166 web pages from different web sites.

# Evaluation

$$\mathrm{r} = \frac{\mathrm{length(k)}}{\mathrm{length(g)}}$$

$$\mathrm{p} = \frac{\mathrm{length(k)}}{\mathrm{length(m)}}$$

$$\mathrm{F1} = 2 * \frac{\mathrm{p} * \mathrm{r}}{\mathrm{p} + \mathrm{r}}$$

# Outline

## Evaluation results based on F1-measure

| | **Al Ahram** | **BBC Arabic** | **BBC Pashto** | **BBC Persian** | **BBC Urdu** | **Embassy** | **Hamshahri** | **Jame Jam** | **Reuters** | **Wikipedia** |
|---|---|---|---|---|---|---|---|---|---|---|
| ACCB-40 | 0.8714 | 0.8255 | 0.8594 | 0.8925 | 0.9476 | 0.7837 | 0.8420 | 0.8398 | 0.8997 | 0.7364 |
| BTE | 0.8534 | 0.4957 | 0.8544 | 0.5895 | 0.9606 | 0.8095 | 0.4801 | 0.7906 | 0.8891 | **0.8167** |
| DSC | 0.8706 | *0.8849* | 0.8398 | *0.9505* | 0.8962 | 0.8238 | *0.9482* | 0.9142 | 0.8510 | 0.7471 |
| FE | 0.8086 | 0.0600 | 0.1652 | 0.0626 | 0.0023 | 0.0173 | 0.2251 | 0.0275 | 0.2408 | 0.2250 |
| KFE | 0.6905 | 0.7186 | 0.8349 | 0.7480 | 0.7504 | 0.7620 | 0.6777 | 0.7833 | 0.8253 | 0.6244 |
| LQF-25 | 0.7877 | 0.7796 | 0.8436 | 0.8410 | 0.9566 | 0.8596 | 0.7650 | 0.7372 | 0.8699 | 0.7735 |
| LQF-50 | 0.7855 | 0.7772 | 0.8374 | 0.8279 | 0.9544 | 0.8561 | 0.7673 | 0.7240 | 0.8699 | 0.7719 |
| LQF-75 | 0.7733 | 0.7727 | 0.8374 | 0.8190 | 0.9544 | 0.8516 | 0.7560 | 0.7240 | 0.8699 | 0.7497 |
| TCCB-18 | *0.8861* | 0.8265 | *0.9121* | 0.9253 | 0.9898 | *0.8867* | 0.8712 | *0.9292* | *0.9593* | 0.8142 |
| TCCB-25 | 0.8737 | 0.8608 | 0.9091 | 0.9271 | *0.9916* | 0.8832 | 0.8884 | 0.9240 | 0.9583 | 0.8142 |
| Density | 0.8787 | 0.2016 | 0.9081 | 0.7415 | 0.9579 | 0.8818 | 0.9197 | 0.9063 | 0.9336 | 0.6649 |
| DANA | **0.9845** | **0.9633** | **0.9363** | **0.9944** | **1.0** | **0.9350** | **0.9797** | **0.9452** | **0.9670** | 0.6740 |

## Average processing time (MB/s)

| | **Time Performance (Megabyte/Second)** |
|---|---|
| ACCB-40 | 0.40 |
| BTE | 0.17 |
| DSC | 7.76 |
| FE | 14.33 |
| KFE | 11.76 |
| LQF-25 | 1.25 |
| LQF-50 | 1.25 |
| LQF-75 | 1.25 |
| TCCB-18 | 17.09 |
| TCCB-25 | 15.86 |
| Density | 7.62 |
| DANA | 19.43 |

# Outline

## Conclusion and Future Work

- Result shows that the DANA produces satisfactory MC with F1-measure $> 0.935$
- We do not need to use parser in the first phase, Using parser is time consuming
- Future works
  - Generalizing DANA by differentiating between tags and text/content to propose a new language-independent content extraction
  - Extending DANA for Wikipedia web pages to obtain better results
  - Trying to improve DANA to a free-parameter algorithm

Thank you for your attention