# Distributed Computing and Big Data

## Assignment 1

**Roudranil Das**
MDS202227
roudranil@cmi.ac.in

**Saikat Bera**
MDS202228
saikatb@cmi.ac.in

**Shreyan Chakraborty**
MDS202237
shreyanc@cmi.ac.in

**Soham Sengupta**
MDS202241
sohams@cmi.ac.in

Chennai Mathematical Institute

January 29, 2023

---

## 1 Best savings on given input

On the given input, the best savings that we have obtained is 97.6%.

## 2 Data processing steps

Pre-processing of the data consisted of mainly the following steps:

- We removed all HTML tags, scripts, links and non-text elements with `soup.get_text()`, and extracted the remnants to text.
- We split this text output by `"\n"`, and we remove the leading and trailing spaces from each of the lines, and append only those into a new list which are not empty strings after stripping.
- We collated a bank of stopwords, and added to them words highly unlikely to occur in Indian addresses. We filtered all these stopwords from each of the lines that we got in the previous text. We also removed characters such as "|" and appended a space to both ends of hyphens. Then we joined all the lines with a newline character to a single string. This ensures that pincodes (which most frequently are preceeded by a hyphen) have atleast one white space preceeding it.
- We used a regular expression pattern to match pincodes and email addresses in the string. Wherever we found a match we stored the start and end indices of a slice of roughly 300-310 characters which possibly contains a pincode (and subsequently an address) and/or an email address. We used a function `remove_overlapping()` to remove overlapping slice ranges. Then we extracted the substrings of the text corresponding to the slices and concatenated all of them to a single string. This ensures that we are extracting only those slices of the text which are most likely to contain an address.

We tried more processing techniques, such as removal of header and footer tags before extracting the fulltext, removal of forms and input sections, extracting address and strong tags directly and extracting div's and other tags with class or id having the word "address". But all of them were inferior in terms of space - saving compared to the one that we finally went with.

## 3 Scope of improvement given time

Given enough time, we identified three key areas where improvement was possible:

1. Better regex filtering to match addresses more efficiently than pincodes: There may be addresses without email id's, phone numbers or even pincodes. We can possibly include filtering with names of states, cities, even the word "India" in this.

2. More filtering words in our stopwords bank: Traditional stopwords list won't be enough as there are multiple frequently used words which are not included in traditionl stopwords list but which most likely won't ever occur in Indian addresses. We can filter out these words from the fulltext.

3. Adaptive guessing of slice ranges: In our work, if `s` and `e` are the starting and ending positions of the regex matches, then we extracted slices `s-201:e+100`. There are only some addresses which span more than 300 characters, and we can tighten this slice much more in most cases. An adaptive guessing of these slices powered by a better regex filtering would have helped to filter out much more words than now.

## 4 Difficulty level and challenges

This task was difficult as addresses, and Indian addresses in particular are highly unstructured pieces of text. Devising perfect regular expression patterns to exactly match Indian addresses is nigh-impossible. We resorted to rule-based heuristics, but for the task of extracting addresses, which come in more formats than we can possibly plan for, such heuristics are also not sufficient.

We faced challenges mainly in the form of websites with inexplicable ways of storing addresses in the content - some choose to do it in body, while there were some who chose to do it in headers/footers. Whenever we planned and implemented a new heuristic in order to trim the number of characters, we would come across some websites which defied all the rules that we had built up. Making one filtering condition to account for such a wide variety of websites and address forms was challenging.