CONTEST OVERVIEW: The general idea is to filter a stream of documents using a dynamic set of exact and approximate continuous keyword match. The goal is to maximize the throughput with which documents are disseminated to active queries. Whenever a new document arrives, the system must quickly determine all queries satisfied by this document.

Task Details

- **Matching Types**
  1. Exact Match
  2. Hamming distance
  3. Edit distance

**Problem**
- Among queries in the context, there are often duplicates of words.

**Approach**
- We performed speed-up of matching by memorizing results of calculations.

**Document1:** (A B C D)

**Document2:**

**Document3:**

**Duplicate**
- In naive method, calculations of distances for duplicated words are performed repeatedly.

**Memorization calculation results**

- Use special data structures to deal with string (8, 16, 32 bytes)
- Use bit operations and SIMD operations
- Minimization of the number of calls to "malloc"
- Use Bit-Vector algorithm for edit distance [1]
- Use B-tree for Map and Set [2]

Other implementation techniques

- Use special data structures to deal with string (8, 16, 32 bytes)
- Use bit operations and SIMD operations
- Minimization of the number of calls to "malloc"
- Use Bit-Vector algorithm for edit distance [1]
- Use B-tree for Map and Set [2]

**Experimental Result**

In order to analyze the performance of our methods, we do two experiments. The first measures the execution time given test input with and without these methods. Following table shows that memorization method improves performance by 20% and filtering improves performance by 50%.

**Execution time with and without each methods**

<table>
<thead>
<tr>
<th>Without All methods</th>
<th>With All methods</th>
<th>Without Memorization</th>
<th>Without filter</th>
<th>Sample Program</th>
</tr>
</thead>
<tbody>
<tr>
<td>Execution time [ms]</td>
<td>4449.8</td>
<td>5665.4</td>
<td>8873</td>
<td>1959967</td>
</tr>
</tbody>
</table>

Next experiment measures the cut rate and the precision rate as follow:

- **Cut rate** = \( \frac{\text{The number of words removed by filter}}{\text{The number of words requested to calculate edit distance}} \)
- **Precision rate** = \( \frac{\text{The number of words match document}}{\text{The number of words passed filter}} \)

Following Graph shows that we can process most queries without calculating exact edit distance. The cut rate of long words looks bad. However, the precision rate of that is high. It means that long words in queries and documents are same in many cases.