Distributed TF-IDF on Husky

Wu Jiayi
The Chinese University of Hong Kong
1155046964@link.cuhk.eud.hk

Introduction to Husky

- Husky [2] is a data-parallel computing system which was expected to better balance high performance and low development cost.
- It is developed mainly for in-memory large scale data mining, and also serves as a general research platform for designing efficient distributed algorithms.

TF-IDF Algorithm

- Term frequency-inverse document frequency (TF-IDF [1]) is a numerical statistic widely used in text mining to reflect the importance of a term to a document in the corpus.
- Term Frequency (TF): the number of times a term occurs in a document
- TF = (Number of times term t appears in a document) / (Total number of terms in the document)
- Common words, such as “the”, “a” and “for”, will always have big term frequency values.
- Inverse Document Frequency(IDF) measures whether the term is common or rare across all documents.
- IDF is a static value to a term and independent of which documents the term is in.

$$idf(T, D) = log_e \frac{N}{|\{d \epsilon D : t \epsilon d\}|} \quad (1)$$

N : total number of documents in the corpus $N = |D|$

$|\{d \epsilon D : t \epsilon d\}|$: number of documents where the term t appears

- TF-IDF = TF * IDF

Performance

- en-wiki * 1: the English Wikipedia corpus, which contains over 4.8 million documents and 1.7 billion terms.
- en-wiki * 2: duplicating “enwiki * 1” by 2 times
- en-wiki * 3: duplicating “enwiki * 1” by 3 times

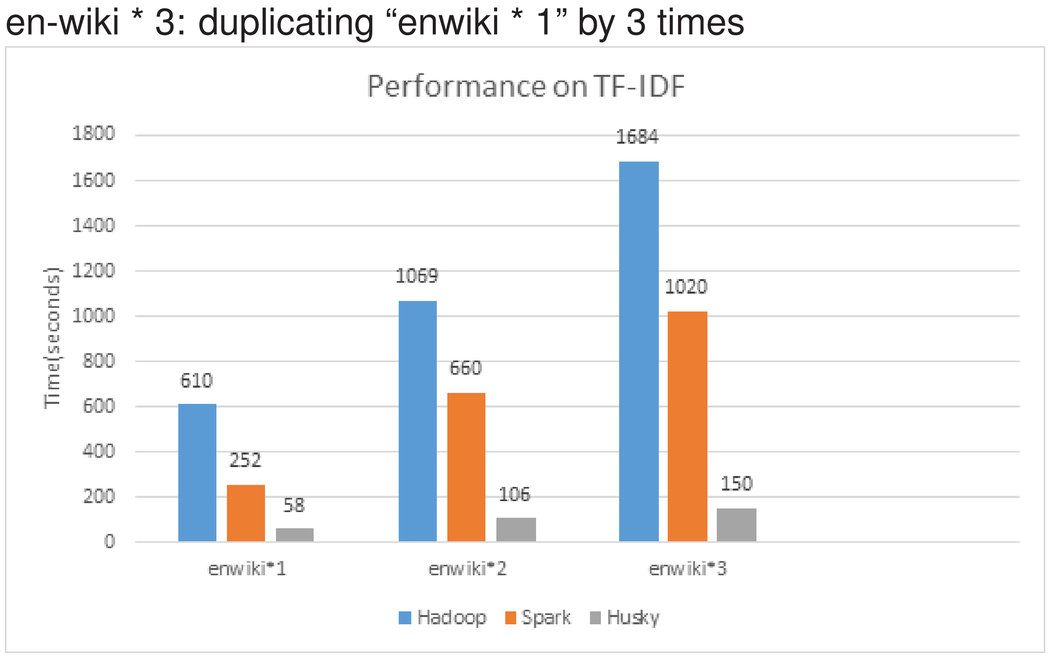

Figure 1: Performance on TF-IDF

- As shown in the Figure 1, Spark is 39% to 58% faster than Hadoop, but Husky is even around 5 times faster than Spark.
- The advantage is more significant when the datasets get larger.

Conclusion

- Husky is an efficient yet easy-to-use data-parallel computing system.
- The distributed implementation of term frequency-inverse document frequency (TF-IDF) on Husky is efficient and scalable.

Acknowledgements

- I would like to express my special thanks of gratitude to Professor James Cheng who gave me the golden opportunity to do this wonderful project. I would also like to thank Fan Yang, Jinfeng, Yuzhen, Yunjian, Legend and all the teammates for the generous help in this summer research internship.

Implementation

- In this implementation, there are two object lists, the document list and the term list.
- Step 1: Since the calculation only requires information for one document and the workers do not need to exchange information, workers can directly calculate term frequency of each word to each document while use an aggregator to count the total number of documents. During the calculation, each document will also send messages to the terms in the document which are members in the term list.

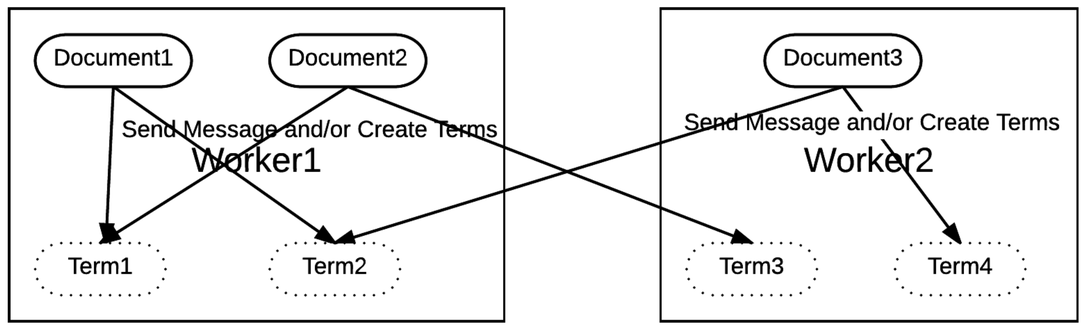

Figure 1: Illustration of sending message in step 1

- Step 2: Once a message is sent to a term that the term list does not contain, the term will be added to the term list. Each term calculate the number of received messages and thus could get IDF.
- Step 3: Each document send requests to all terms in it and the ids of the requests are same as the terms while each term broadcasts its inverse document frequency value as its reply. Then, each request sent by a document can find a corresponding reply by finding the reply given by the term with the same id with itself. So each term for each document can get its TF-IDF value by multiplying the IDF value to the TF value calculated in step 1.

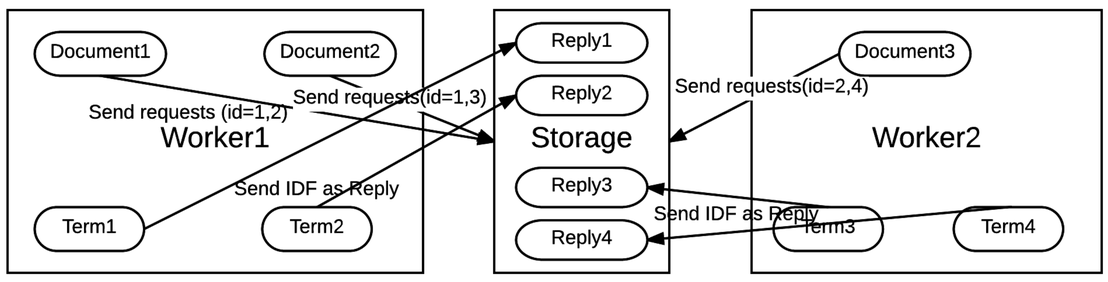

Figure 2: Illustration of sending message in step 3

- Step 4: The implementation offers a function which enables users to check the TF-IDF value of a given word in a given document. This implementation also enables users to get the list name of the document list so that the user could get all the information stored in the document list.

Discussion

- Two different mechanisms are used in step 1 and step 3 for sending and receiving messages
- In step 1, since workers do not know which worker the term that a message is sent to is on, the term list is globalized among all workers which cause much communication.
- In step 3, since each document has already in the document list, it is not necessary to globalize document list. Using this request-and-reply mechanism, the term does not need to know which document requests its reply and thus how the documents are stored on workers does not need to be broadcasted.

References

[1] H. C. Wu, R. W. P. Luk, K. Wong, and K. Kwok. Interpreting TF-IDF term weights as making relevance decisions. *TOIS*, 26(3), 2008.

[2] F. Yang, J. Li, and J. Cheng. Husky: Towards a more efficient and expressive distributed computing framework. *PVLDB*, 9(5):420–431, 2016.