Towards Optimization of Parallelized Mining of Subgraphs Sharing Common Items Using a Task-Parallel Language

Jing Xu
Kyoto University
Kyoto, Japan
xu.jing.38s@st.kyoto-u.ac.jp

Tasuku Hiraishi
Kyoto Tachibana University
Kyoto, Japan
hiraishi@tachibana-u.ac.jp

Shingo Okuno*
Kyoto University
Kyoto, Japan

Masahiro Yasugi
Kyushu Institute of Technology
Fukuoka, Japan

Keiichiro Fukazawa
Kyoto University
Kyoto, Japan
fukazawa@media.kyoto-u.ac.jp

1 INTRODUCTION

This presentation proposes parallel implementations of a highly complex graph mining problem to extract Closed Common Itemset connected subGraphs (CCIGs) from a graph whose vertices are labeled by their own sets of items, or itemsets in short, each of which satisfies that the cardinality of its common itemset, i.e., the intersection of the itemsets of all of its vertices, is not less than a given threshold.

An efficient sequential algorithm called COPINE [2] was proposed for this problem. This algorithm applies a depth-first search to a search tree. To reduce the search space, COPINE prunes tree edges from which the following three types of subgraphs are derived; subgraphs that have already been visited (Pruning 1), subgraphs of which the itemset has a smaller cardinality than the threshold (Pruning 2), and subgraphs that are not closed because one of their supergraphs has already been visited and their itemsets are identical (Pruning 3).

Later, a parallel extension of the COPINE and its implementations using the Tascell task-parallel language were proposed [1], where the search tree is divided into sub-search trees and a unique set of subtrees that are assigned to each worker are traversed in almost the same way as in the sequential search. The existing parallel COPINE implementations have a problem that search space expands compared to a sequential search, mainly because a worker traversing a search tree cannot prune subtrees on the "left" side for Pruning 3. There is another problem even in sequential search. In the existing implementations, an *itemset table* is employed for achieving Pruning 3 and a worker checks or updates the table at every search step. The cost for such operations is considerable especially when the number of items registered to the table increases.

To alleviate these issues, we propose two mechanisms for the parallel COPINE. First, we allow a worker to prune a subtree on the "left" side of a search tree node when certain conditions are satisfied. With this new algorithm, workers can prune sub-search trees more aggressively and we can expect the search space can be reduced. Second, we let a worker check and update an information table only when a certain custom condition is satisfied rather than at every search step. This mechanism enlarges the search space but the overhead for the table reference can be reduced.

2 OUR PROPOSAL

Right-to-Left (RTL) Pruning. In the parallel COPINE, an itemset table for Pruning 3 is shared by all workers. The existing parallel implementations introduce a constraint that *a worker can only refer to information for Pruning 3 that had been registered earlier in a sequential search.* That is to say, pruning only "from left to right" is allowed. This constraint is necessary for avoiding excessive pruning but brings search space expansion. We alleviated this constraint so that a worker can apply Pruning 3 "from right to left" as follows. Suppose a worker w_1 visits a search tree node n_1 first and w_2 visits n_2 later. The worker w_2 can refer to information registered at n_1 *even if* n_1 *is on the right side of* n_2 when the children of n_2, being a set of vertices, is a subset of the children of n_1.

Reducing the Number of Itemset Table Access. To reduce the cost of the table access, we proposed a simple mechanism: a worker stops checking an itemset table for Pruning 3 at *every* search step. At each step, a worker evaluates some condition and performs the access only when it is satisfied. Here, we can determine the condition in an arbitrary manner. In this research, we employed the condition that a worker performs the table access only at a search step where the degree of the last added vertex is not less than a given threshold d.

3 PERFORMANCE EVALUATION

We evaluated the performance of the implementations described above using a single node of the Laurel 2 supercomputer of Kyoto University and a real protein network. The execution results show that, with the right-to-left pruning mechanism, the search space can be reduced when the number of workers is not less than 8 and the size of the search space decreases as the number of workers increases. We also confirmed that we can improve the traversal speed using the mechanism for reducing the number of itemset table access.

REFERENCES

[1] Shingo Okuno, Tasuku Hiraishi, Hiroshi Nakashima, Masahiro Yasugi, and Jun Sese. 2014. Parallelization of Extracting Connected Subgraphs with Common Itemsets. *IPSJ Trans. Programming* 7, 3 (2014), 22–39.

[2] Jun Sese, Mio Seki, and Mutsumi Fukuzaki. 2010. Mining Networks with Shared Items. In *Proc. 19th ACM International Conference on Information and Knowledge Management (CIKM '10)*. 1681–1684.

*Presently with Fujitsu Limited.