Dynamic Programming-Based Multilevel Graph Partitioning for Large-Scale Graph Data

Sergio P. Caetano, Miri Yu, Jiheon Choi, Sangyoon Oh*

Ajou Univ.

sa20ra202124015@ajou.ac.kr, zztiok@ajou.ac.kr, unidev@ajou.ac.kr, *syoh@ajou.ac.kr

대규모 그래프 데이터를 위한 동적 프로그래밍 기반 다단계 그래프 분할

세르지오, 유미리, 최지헌, 오상윤*
아주대학교

Abstract

Multilevel graph algorithms are used to create optimal partitions for large graphs. However, the dynamic changes to the graph structures during partitioning lead to increased memory. These changes involve adding temporal data to arrays or queues during intermediary operations. To enhance efficiency and minimize memory usage, we integrated dynamic programming. Experimental results demonstrate the improved scalability and effectiveness of the proposed approach in terms of memory usage.

Ⅰ. Introduction

Multilevel graph partitioning algorithms create optimal partitions for large-scale graphs through coarsening, initial partitioning, and uncoarsening. During the coarsening and uncoarsening phases, efficient contraction and uncontraction techniques, such as contracting matching (e.g., maximum matching) [1] or clusters (e.g., label propagation) [2], are used. Contraction matching algorithms are very powerful, but this technique is not suitable for partitioning complex and large-scale networks that only admit a small maximum matching. Thus, most graph partition algorithms often use label propagation because it is simpler and more effective [3].

However, using label propagation in multilevel algorithms requires high memory usage because all possible combinations of assigned nodes are included in the search space (i.e., memory usage). The situation worsens as the graph size increases, leading to an exponential growth in search space and presenting a significant challenge.

To overcome this challenge, we propose a novel graph partitioning scheme that consists of label propagation just for coarsening, caching the results of each hierarchal level, and storing them using a dynamic programming technique called memorization, which helps to avoid recalculating intermediary results using label propagation for uncoarsening.

Ⅱ. Related Works

The multilevel graph partitioning process operates across multiple hierarchical levels, starting with an original state and refining the partitioning while coarsening the graph representation [1]. Its coarsening process reduces the original graph while preserving its structural information [2], followed by initial partitioning, which divides the graph recursively until the desired number of partitions is achieved. In uncoarsening (unfolding) phase, the contracted nodes and edges are restored to their original state in the finer graph by undoing the contraction and coarsening operations from the coarsening phase [3].

There are studies that have used multilevel graph partitioning algorithms to reduce computational costs. Bae et al. [2] proposed a label propagation-based parallel graph partitioning scheme for large-scale graph data. In the scheme, the stabilization phase is introduced to prevent the algorithm from getting trapped in local optima by relocating remote and highly connected vertices. Recently, Sanders et al. try to overcome the imbalance faced by many multilevel graph partitioning. In the paper, distributed deep multilevel graph partitioning scheme is proposed, where K-way partitioning is combined with recursive bi-partitioning [3].

III. Dynamic Programming-based Multilevel Graph Partitioning (DPMGP)

To solve the memory usage problem while preserving the cut quality and balance constraint, our scheme uses a multilevel graph partitioning approach for coarsening to create hierarchical levels. Next, a dynamic programming technique is applied to each coarser graph, starting from the bi-partitioning step of the original graph. The scheme consists of three phases, following the established building blocks of multilevel graph partitioning. First, the size-constrained label propagation is applied during the coarsening phase. Then, the bi-partitioning with the Kernighan-Lin (KL) algorithm is used for initial partitioning. Finally, the uncoarsening (refinement) phase using dynamic programming (memoization) is applied. It is specialized for the cached results or subgraphs obtained from the coarsening phase by label propagation, and it is applied to the reconstruction algorithms based on the state transition function: $F_{(n)} = F_{(n-2)} - F_{(n-1)}$, following the order of computation from G_2 to the original graph G_0 (bottom-up approach). All the operations are based on the location of the answer (subgraph): the original graph G_0 for the given example ($F_{(n)}$).

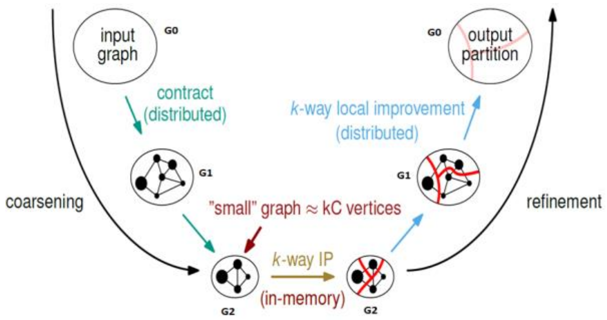

Figure. 1: Dynamic Programing-Based Multilevel Graph Partitioning for large-scale graph data (DPMGP).

IV. Evaluation

To measure the effectiveness of our proposed scheme, we have conducted empirical experiments. The experimental environment with the graph dataset we used is summarized in Table 1.

Table 1: Experimental Environment

Classification	Specifications
Computer System	CPU: Intel Core i7, 3.20 GHz Memory: 32 GB. HDD: 1 TB
O.S.	Ubuntu 22.04 LTS
Graph Dataset	Large_Twitch_Gamers (SNAP dataset)

The results of our experiments are summarized in Table 2. In the table, the memory required for label propagation-based multilevel graph partitioning and dynamic Programming-based graph partitioning algorithms to perform partitioning tasks are shown. Our proposed scheme with dynamic programming performed the partitioning process with less than 10% memory space that is required for the general label propagation scheme. The execution time is also improved, and it performed ten times faster than the label propagation.

Table 2: Memory usage (MB).

Approaches	Maximum (MB)	Minimum (MB)	Standard Deviation
General label propagation	2967.01	2175.77	242.76
Our proposed scheme	188.02	186.41	0.76

V. Conclusion

Multilevel graph partitioning algorithms are commonly used to improve partitioning efficiency. This work proposes a dynamic programming-based multilevel graph partitioning for large-scale graph data on the shared memory. With the algorithm, we achieves near-optimal results in terms of not only memory requirement constraints, which is the main problem addressed in this paper, but also the running time of our algorithm (DPMGP).

ACKNOWLEDGMENT

본 연구는 2023 년도 한국연구재단 기본연구사업 (NRF-2021R1F1A1062779) 및 초고성능컴퓨팅 SW 생태계조성사업(RS-2023-00283799)의 연구비 지원으로 수행하였습니다.

REFERENCES

[1] G. Karypis and V. Kumar, ‘Multilevel k-way partitioning scheme for irregular graphs’, Journal of Parallel and Distributed Computing, Vol. 48, no. 1, pp. 96-129, 1998.

[2] M. Bae, M. Jeong, and S. Oh, ‘Label Propagation-Based Parallel Graph Partitioning for Large-Scale Graph Data’, IEEE Access, Vol. 8, pp. 72801-72813, 2020.

[3] P. Sanders and D. Seemaier, ‘Distributed Deep Multilevel Graph Partitioning’, in Euro-Par 2023: Parallel Processing, 2023, pp. 443-457.