Compressing Graph Data by Leveraging Domain Independent Knowledge

Dr. Sirisha Velampalli
Assistant Professor
CR Rao Advanced Institute of Mathematics, Statistics & Computer Science
University of Hyderabad Campus, Hyderabad, Telangana-500046, India
sirisha.crraoaimscs@gmail.com

Abstract

Graphs are used to solve many problems in the real world. At the same time size of the graphs presents a complex scenario to analyze essential information that they contain. Graph compression is used to understand high level structure of the graph through improved visualization. In this work, we introduce CRADLE (CompRessing grAph data with Domain independent knowLEdge), a novel method based on knowledge rule called netting, which reports the number of external networks for each instance of the substructure. By finding such substructures with more number of external networks we can judiciously improve the compression rate. We empirically evaluate our approach using diverse datasets. We compare CRADLE with baseline approaches. Our proposed approach is comparable in compression rate, search space, and runtimes to other well-known graph mining approaches.

Introduction

Graphs provide a meaningful representation that can be used for searching, analyzing, or discovering interesting patterns, because of the capability to represent complex relations. Identifying interesting substructures that can increase the ability to interpret data is of great importance (Chittimoori, Holder, and Cook 1999). These substructures should be able to compress the data. Whenever a compressed graph is able to conserve the characteristics of the original graph it can be visualized easily (Zhou 2015). Efficient storage can be achieved using graph compression (CHAVAN). In earlier work by Cook and Holder, they use background knowledge to further refine the search process for discovering interesting normative patterns (Cook and Holder 1994). In this work, it is our hypothesis that one could use background knowledge in the form of rule netting, augmenting existing evaluation techniques, such as MDL and size (used in approaches like SUBDUE), to discover substructures with improved compression rate, search space and runtimes to other approaches.

Proposed Approach: Our proposed method CRADLE (CompRessing grAph data with Domain independent knowLEdge) can find substructures that increases the ability to compress the data. Our approach uses domain independent knowledge to find interesting substructures.

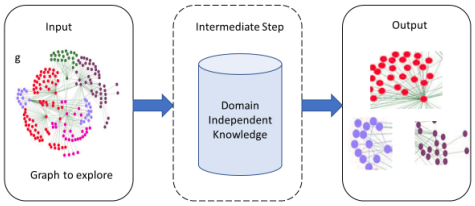

Figure 1: Overview of CRADLE- Given an input graph g, we add domain independent knowledge in the form of rule netting leading to output interesting substructures

Overview of CRADLE is shown in Figure 1. Knowledge is added in the form of rule, netting which reports the number of external networks for each instance of the substructure. It is our hypothesis that substructure with more number of connections can compress the graph well compared to other substructures.

The contributions of this work are as follows:

Novel approach: Propose novel domain independent knowledge rule, called netting which reports the number of external networks for each instance of the substructure.

Novel Algorithms: We propose CRADLE-MDL & CRADLE-Size methods that use Minimum Descriptive Length (MDL) and size as metrics respectively, to aid in discovering interesting substructures.

Experiments: We perform empirical evaluations on diverse datasets. We compare CRADLE with well-known graph mining approaches.

Related Work

Discovering interesting substructures in a structural database improves the ability to interpret and compress the data. Scientists working with a database in their area of expertise often search for predetermined types of structures or for structures exhibiting characteristics specific to the domain. Many graph mining techniques available in literature differ mainly in (1) substructure interestingness and (2) substructure evaluation.

Mathematical graph-theory based approaches focus on mining graph data-sets that are frequent and complete. They use a support or frequency measure in evaluating the substructures. AGM (Inokuchi, Washio, and Motoda 2000) one of the oldest algorithms which uses an apriori level-wise approach. (Kuramochi and Karypis 2001) uses a pattern-growth approach. It adopts an edge-based candidate generation strategy that increases the substructure size by one edge. The limitations of AGM and FSG algorithms are that they generate a huge number of candidates, perform multiple database scans, and mining long patterns is extremely complex. gFSG (Yan and Han 2002) is a variant of FSG which enumerates all geometric subgraphs from the database. FFSM is a graph mining system which uses an algebraic graph framework to address the underlying problem of subgraph isomorphism. However, both gFSG and FFSM are NP-Complete problems. Mining frequent patterns and subgraphs are also used in other graph-based problems, such as anomaly detection, clustering, classification, and analysis. Akoglu et al. propose a structure-based algorithm named OddBall to analyze social network graphs (Akoglu, McGlohon, and Faloutsos 2010). Reza et al. propose an algorithm and a framework to detect anomalies in an unlabelled social network Park and Chung propose a MapReduce algorithm based on graph partitioning to count the number of triangles, an important measure in graph analysis (Park and Chung 2013). Arora et al. propose a Voronoi based Push Preflow method to find the min-cut, which not only exploits the structural properties inherent in image-based grid graphs but also combines the basic paradigms of max-flow theory in a novel way (Arora et al. 2010). Mookiah et al. study the problem of detecting changes in news articles for specific social issues such as human trafficking and road accidents. The authors propose a graph-cut algorithm that can mark the change detected articles (Mookiah, Eberle, and Mondal 2016).

Although completeness is a fundamental and desirable property, a side effect of the existing mathematical graph theory based approaches is that these systems typically generate a large number of substructures, which by themselves provide relatively less insight about the domain. In comparison to existing mathematical graph theory based approaches which are complete, we propose greedy search based approach. Instead of using frequency or support we use rule called netting as background knowledge to evaluate the substructures and find interesting patterns with improved compression rates, search space, and runtimes over existing approaches.

Proposed Algorithms

What we are proposing in this work is a novel way to detect graph-based substructures that improve upon existing approaches by reducing the compression rates, and time complexity. Our first proposed approach we call CompRessing grAph data with Domain independent knowLEdge based on the MDL evaluation metric (CRADLE-MDL). Algorithm 1 presents the steps of our proposed CRADLE-MDL approach. Our second proposed approach we call CompRessing grAph data with Domain independent knowLEdge based on a size evaluation metric (CRADLE-Size). Algorithm 2 presents the steps of our proposed CRADLE-Size approach.

Algorithm 1 *CRADLE-MDL:* CompRessing grAph data with Domain independent knowLEdge based on the MDL

1: **Goal:** Reduce Time Complexity, Space Complexity with increased Compression Percentage
2: **procedure** CRADLE-MDL–ALGORITHM
3: Find all normative substructures S using **MDL approach** on input graph G which minimizes *DL(S)+DL(G|S). DL= Description Length*
4: Store *S* in list *L*.
5: **for** each substructure S" in L **do**
6: Determine the number of external networks for each instance of the substructure *S"* with other substructures present in Graph G.
7: Store *S"* in ordered list *B* (highest to lowest) in its corresponding ranked position.
8: **end for**
9: Return *B*
10: **end procedure**

Algorithm 2 *CRADLE-Size:* CompRessing grAph data with Domain independent knowLEdge based on the Size

1: **Goal:** Reduce Time Complexity, Space Complexity with increased Compression Percentage
2: **procedure** CRADLE-SIZE–ALGORITHM
3: Find all normative substructures S using **Size approach** on input graph G which minimizes *size(S)+size(G|S). size=* $|V|+|E|$
4: Store *S* in list *L*.
5: **for** each substructure S" in L **do**
6: Determine the number of external networks for each instance of the substructure *S"* with other substructures present in Graph G.
7: Store *S"* in ordered list *B* (highest to lowest) in its corresponding ranked position.
8: **end for**
9: Return *B*
10: **end procedure**

Both algorithms first discover normative substructures Si where a normative substructure S is a subgraph that has an associated description. After which, for each instance in the substructure it searches for number of external networks with other substructures among the substructures Si. The difference between these two algorithms lies in the evaluation metric. While we can discover interesting substructures using either algorithm, there are pros and cons to each approach. For instance, the CRADLE-Size approach is faster because it uses a simple size evaluation metric, whereas calculating compression (CRADLE-Size) is slightly more costly. However, the structure of the graph may affect the discovery process. For example, if there are many overlapping substructures in a graph, the size metric may discover more interesting substructures that the MDL metric may not, and vice-versa. In addition, the MDL metric is more widely

used in the literature, and has many applications in various domains. Thus, we will evaluate both approaches on a variety of different graphs.

Evaluation Metrics: MDL, Size

The hypothesis of this work it that we add domain knowledge in the form of rule netting, to evaluation metrics in order to guide the graph-based substructure discovery process. Our proposed algorithm CRADLE-MDL uses the MDL evaluation metric, whereas our proposed CRADLE-Size approach uses the size evaluation metric. The concept of MDL (Rissanen 1984) was first introduced by Jorma Rissanen in 1978. The MDL principle involves the relation between the regularity in data and the compression of data. The principle implies that whenever we are able to compress the data well, there is much regularity in the data. In particular, MDL is well-suited for model selection problems, such as substructure discovery, decision tree induction, genetic programming and image processing (Cook and Holder 1994),(Quinlan and Rivest 1989), (van Leeuwen and Vreeken 2014), (Quinlan and Rivest 1989). In order to implement our approach, we will use the publicly available SUBDUE system. SUBDUE, one of the well-known substructure discovery algorithms (Cook and Holder 1994), uses a model evaluation method called Minimum Encoding, a technique derived from the MDL principle.

Domain Knowledge to Evaluation Metrics: *Rule Netting*

The hypothesis of our work is that we could use background knowledge in the form of rule netting that will improve upon the ability to discover interesting substructures with improved compression rate, search space and runtimes. Netting reports the number of external networks for each instance of the substructure.

$$Netting = 1+\frac{1}{|I|}log(b+1)+\sum_{iinI} weight(i)*\texttt{Ext\_networks(i)} \quad (1)$$

where,
I is the set of instances of substructure S.
Ext_networks(i) is the number of edges connecting a vertex in an instance to a vertex outside the instance.

$$weight = 1 - \frac{matchcost(S,i)}{size(i)} \quad (2)$$

Also, matchcost(S,i) is the cost required to match an instance to a substructure. Specifically, it is the number of vertices and edges that would need to be changed in order to derive a matching substructure.

Netting: Example Calculation of netting is explained by taking the same sample graph shown in Figure 2, where the number of vertices is 5 and the number of edges is 4 for a total size of 9. Using Equation (1), the values for the substructures in Figure 2 are shown in Table 1.

Among the substructures shown in Table 1 we can say that substructure B (value highlighted in bold italic) has highest netting value with 2 instances and 4 external connections,

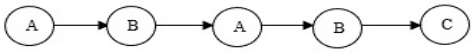

Figure 2: Sample Graph

Table 1: Netting Values for Substructures

Substructures	Netting Value
A	2.5
B	***3***
C	**2**
A-B	2.5
B-A	1.3333
B-C	**2**

and substructures C, B-C (value highlighted in bold) have low netting values with only 1 instance and 1 external connection. It should be noted that for this example, the list is not exhaustive, and only substructures up to two vertices and one edge are shown, even though the largest substructure would consist of 5 vertices.

Experiments

We compare CRADLE with other three well-known graph mining approaches: SUBDUE (Cook and Holder 1994), (Kuramochi and Karypis 2001) and (Yan and Han 2002). All experiments are run under the following Hardware specifications:

- Processor Intel(R) Core(TM) i3-5005U CPU @2.00GHz 2.00 GHz, 2 Core(s), 4 Logical Processor(s).
- RAM 4.00GB.
- Operating system: xubuntu 16.04.

Experimental Evaluations: Synthetic Datasets

We used the subgen tool (Eberle and Holder 2011) for our experiments. subgen is a synthetic generator that generates graphs using the user-specified parameters namely size of the graph, names of vertex and edge labels, substructure pattern, connectivity value, overlap value. For example, using a graph size of 1000 (500 vertices and 500 edges), with a normative pattern of 4 vertices and 4 edges (shown in Figure 3), the compression rates obtained using our approach CRADLE in comparison to other approaches along with runtimes are shown in Table 2.

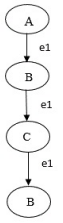

Figure 3: Normative Pattern

Table 2: Results of Artificial Datasets

S.No	Approach	Compression (Percentage)	Runtime (Seconds)
1.	CRADLE-MDL	19%	7.24
2.	CRADLE-Size	22%	4.28
3.	Subdue	8%	17.78
4.	FSG	2%	3.96
5.	gSpan	2%	2.38

Table 3: Results-Chemical Compound Domain (422 molecules)

S.No	Approach	Compression (Percentage)	Runtime (Seconds)
1.	CRADLE-MDL	22%	65.28
2.	CRADLE-Size	25%	50.26
3.	Subdue	19%	132.98
4.	FSG	7%	19.21
5.	gSpan	7%	3.22

Results obtained clearly show that CRADLE outperformed in terms of compression percentage in comparison to other well-known graph mining approaches. However, runtimes needed to discover best substructure using CRADLE is higher compared to FSG and gSpan approaches.

Comparison of CRADLE with SUBDUE, FSG and gSpan on Real-World Datasets

We compare our approach CRADLE with other graph mining systems SUBDUE, FSG and gSpan on chemical compound domain. We experiment with datasets that are available with gSpan (Yan and Han 2002) that contains 422 molecules and the results are shown in Table 3.

Analysis

Similar to synthetic data results, CRADLE outperformed in terms of compression percentage in comparison to other well-known graph mining approaches in real datasets as well. However, runtimes needed to discover best substructure using CRADLE is higher compared to FSG and gSpan approaches.

Conclusion and Future work

In this work, we introduced CRADLE (CompRessing grAph data with Domain independent knowLEdge), a novel method based on knowledge rule called netting, which reports the number of external networks for each instance of the substructure. We demonstrate the effectiveness of our approach on various diverse datasets, through the comparison of execution times and compression rates of our proposed approach against other well-known graph mining approaches. Results of our study is promising for finding required patterns, but more research is needed in terms of scalability, as well as discovering more diverse normative patterns.However, we are currently investigating additional knowledge rules that can be added to discover other interesting substructures that are specific to the domain.

References

Akoglu, L.; McGlohon, M.; and Faloutsos, C. 2010. Oddball: Spotting anomalies in weighted graphs. *Advances in Knowledge Discovery and Data Mining* 410–421.

Arora, C.; Banerjee, S.; Kalra, P.; and Maheshwari, S. 2010. An efficient graph cut algorithm for computer vision problems. In *European conference on computer vision*, 552–565. Springer.

CHAVAN, A. An introduction to graph compression techniques for in-memory graph computation.

Chittimoori, R. N.; Holder, L. B.; and Cook, D. J. 1999. Applying the subdue substructure discovery system to the chemical toxicity domain. In *FLAIRS Conference*, 90–94.

Cook, D. J., and Holder, L. B. 1994. Substructure discovery using minimum description length and background knowledge. *Journal of Artificial Intelligence Research* 1:231–255.

Eberle, W., and Holder, L. B. 2011. Graph-based knowledge discovery: Compression versus frequency. In *FLAIRS Conference*.

Inokuchi, A.; Washio, T.; and Motoda, H. 2000. An apriori-based algorithm for mining frequent substructures from graph data. *Principles of Data Mining and Knowledge Discovery* 13–23.

Kuramochi, M., and Karypis, G. 2001. Frequent subgraph discovery. In *Data Mining, 2001. ICDM 2001, Proceedings IEEE International Conference on*, 313–320. IEEE.

Mookiah, L.; Eberle, W.; and Mondal, M. 2016. Detecting change in news feeds using a context based graph. In *Proceedings of the International Conference on Data Mining (DMIN)*, 161. The Steering Committee of The World Congress in Computer Science, Computer Engineering and Applied Computing (WorldComp).

Park, H.-M., and Chung, C.-W. 2013. An efficient mapreduce algorithm for counting triangles in a very large graph. In *Proceedings of the 22nd ACM international conference on Information & Knowledge Management*, 539–548. ACM.

Quinlan, J. R., and Rivest, R. L. 1989. Inferring decision trees using the minimum description lenght principle. *Information and computation* 80(3):227–248.

Rissanen, J. 1984. Universal coding, information, prediction, and estimation. *IEEE Transactions on Information theory* 30(4):629–636.

van Leeuwen, M., and Vreeken, J. 2014. Mining and using sets of patterns through compression. In *Frequent Pattern Mining*. Springer. 165–198.

Yan, X., and Han, J. 2002. gspan: Graph-based substructure pattern mining. In *Data Mining, 2002. ICDM 2003. Proceedings. 2002 IEEE International Conference on*, 721–724. IEEE.

Zhou, F. 2015. Graph compression. *Department of Computer Science and Helsinki Institute for Information Technology HIIT* 1–12.