Subdue: Compression-Based Frequent Pattern Discovery in Graph Data

Nikhil S. Ketkar
University of Texas at Arlington
ketkar@cse.uta.edu

Lawrence B. Holder
University of Texas at Arlington
holder@cse.uta.edu

Diane J. Cook
University of Texas at Arlington
cook@cse.uta.edu

ABSTRACT
A majority of the existing algorithms which mine graph datasets target complete, frequent sub-graph discovery. We describe the graph-based data mining system Subdue which focuses on the discovery of sub-graphs which are not only frequent but also compress the graph dataset, using a heuristic algorithm. The rationale behind the use of a compression-based methodology for frequent pattern discovery is to produce a fewer number of highly interesting patterns than to generate a large number of patterns from which interesting patterns need to be identified. We perform an experimental comparison of Subdue with the graph mining systems gSpan and FSG on the Chemical Toxicity and the Chemical Compounds datasets that are provided with gSpan. We present results on the performance on the Subdue system on the Mutagenesis and the KDD 2003 Citation Graph dataset. An analysis of the results indicates that Subdue can efficiently discover best-compressing frequent patterns which are fewer in number but can be of higher interest.

1. INTRODUCTION
Recently, an increasing body of research has focused on developing algorithms to mine graph datasets. A graph representation provides a natural way to express relationships within data. Graph-based data mining expresses data in the form of graphs, and focuses on the the discovery of interesting sub-graph patterns.

Graph-based data mining has been successfully applied to various application domains including protein analysis[19], chemical compound analysis[1], link analysis[13] and web searching[16]. A number of varied techniques and methodologies have been applied to mining interesting subgraph patterns from graph datasets. These include mathematical graph theory based approaches like FSG[10] and gSpan[20], greedy search based approaches like Subdue [2] or GBI[12], inductive logic programming (ILP) approaches like WARMR[3], inductive database approaches like MolFea[15] and kernel function based approaches[8].

Mathematical graph theory based approaches mine a complete set of subgraphs mainly using a support or frequency measure. The initial work in this area was the AGM[6] system which uses the Apriori level-wise approach. FSG takes a similar approach and further optimizes the algorithm for improved running times. gFSG [9] is a variant of FSG which enumerates all geometric subgraphs from the database. gSpan uses DFS codes for canonical labeling and is much more memory and computationally efficient than previous approaches. Instead of mining all subgraphs, CloseGraph[21] only mines closed subgraphs. A graph G is closed in a dataset if there exists no supergraph of G that has the same support as G. Gaston [14] efficiently mines graph datasets by first considering frequent paths which are transformed to trees which are further transformed to graphs. FFSM [5] is a graph mining system which uses an algebric graph framework to address the underlying problem of subgraph isomorphism. In comparison to mathematical graph theory based approaches which are complete, greedy search based approaches use heuristics to evaluate the solution. The two pioneering works in the field are Subdue and GBI. Subdue uses MDL-based compression heuristics, and GBI uses an empirical graph size-based heuristic. The empirical graph size definition depends on the size of the extracted patterns and the size of the compressed graph. Another methodology in this field is that of inductive logic programming which has the advantage of the extensive descriptive power of first-order logic. The first graph-based system to combine the ILP method with Apriori-like level-wise search was WARMR. The major advantage of these approaches is their high representation power. WARMR was used on carcinogenesis prediction of chemical compounds [7].

Another promising direction in the field of graph-based data mining is that of inductive databases which are a new generation of databases that are not only capable of dealing with data but also with patterns or regularities within the data. Data mining in such a framework is an interactive querying process. The inductive database framework is especially interesting for bioinformatics and chemoinformatics, because of the large and complex databases that exist in these domains, and the lack of methods to gain scientific knowledge from them. The pioneer work in this field was the MolFea system, which is based on the level-wise version space algorithm. MolFea is the Molecular Feature miner that mines for linear fragments in chemical compounds. Lastly, the kernel function based approaches have been used to a certain extent for mining graph datasets. The kernel function defines a similarity between two graphs. When high dimensional data is represented in linear space, the function to learn is difficult in that space. We can map the linear data to nonlinear space and the problem of learning in that high dimensional space becomes learning scalar products. Kernel functions make computation of such scalar products very efficient. Learning in this high dimensional space becomes difficult. The key is finding efficient mapping functions and good feature vectors. The pioneering approach that applied kernel functions to graph structures is the diffusion kernel [8].

In this paper we present the graph-based data mining system Subdue which approaches the problem of mining interesting substructures in graph datasets by finding subgraphs which are not only frequent but also compress the graph dataset. The driving principle of Subdue's approach is the use of the minimum description length (MDL) principle to evaluate the interestingness of a substructure. The use of MDL rather than frequency to evaluate a substructure distinguishes Subdue from all other frequent subgraph discovery systems. Subdue typically produces a fewer number of substructures which best compress the graph dataset. These few substructures which compress the input dataset can provide important insights about the domain than a large number of substructures, each of which is frequent, in the graph dataset. The rest of the paper is organized as follows. In Section 2 we discuss the Subdue system. In Section 3 we discuss the suitability of a compression-based methodology to frequent subgraph discovery. In Section 4 we perform a comparison of Subdue with the graph-based data mining systems FSG and gSpan on the Chemical Toxicity and the Chemical Compounds datasets that are provided with gSpan. In Section 4 we present Subdue's results on Mutagenesis and the KDD 2003 Citation Graph dataset. Conclusions and future work are presented in Section 6.

2. SUBDUE

Subdue is a graph-based relational learning system. The work on Subdue is one of the pioneering works in the field of graph-based data mining. Inputs to the Subdue system can be a single graph or a set of graphs. The graphs can be labeled or unlabeled. Subdue outputs substructures that best compress the input dataset according to the Minimum Description Length (MDL) [17] principle. Subdue performs a computationally-constrained beam search which begins from substructures consisting of all vertices with unique labels. The substructures are extended by one vertex and one edge or one edge in all possible ways, as guided by the example graphs, to generate candidate substructures. Subdue maintains the instances of substructures in the examples and uses graph isomorphism to determine the instances of the candidate substructure in the examples. Substructures are then evaluated according to how well they compresses the Description Length (DL) of the dataset. The DL of the input dataset G using substructure S can be calculated using the following formula,

$$I(S) + I(G|S)$$

where S is the substructure used to compress the dataset G. I(S) and I(G|S) represent the number of bits required to encode S and dataset G after S compresses G. The length of the search beam defines the number of substructures retained for further expansion. This procedure repeats until all substructures are considered or user-imposed computational constraints are exceeded. At the end of the procedure Subdue reports the best compressing substructures. The following is the algorithm for Subdue's discovery process,

```
SUBDUE(Graph, BeamWidth, MaxBest, MaxSubSize, Limit)
 1  ParentList = NULL;
 2  ChildList = NULL;
 3  BestList = NULL;
 4  ProcessedSubs = 0;
 5  Create a substructure from each unique vertex label
    and its single-vertex instances;
 6  Insert the resulting substructures in ParentList;
 7  while ProcessedSubs less than or equal to Limit
    and ParentList not empty
 8        do
 9           while ParentList is not empty
10               do
11                  Parent = RemoveHead(ParentList);
12                  Extend each instance of Parent
                    in all possible ways;
13                  Group the extended instances
                    into Child substructures;
14           for each Child
15               do
16                  if SizeOf(Child) less
                    than MaxSubSize
17                     then
18                          Evaluate the Child;
19                          Insert Child in ChildList in
                            order by value;
20                          if BeamWidth less than
                            Length(ChildList)
21                             then
22                                  Destroy
                                    substructure
                                    at end of
                                    ChildList;
23           Increment ProcessedSubs;
24           Insert Parent in BestList
             in order by value;
25           if MaxBest less than Length(BestList)
26              then
27                  Destroy substructure
                    at end of BestList;
28           Switch ParentList and ChildList;
29  return BestList;
```

The Subdue system provides a user-defined option of performing an inexact graph match in the discovery procedure described above. This option is useful in the case where substructure instances can appear in different forms throughout the graph dataset. In this approach, the user can also assign a distortion cost to each graph distortion. A distortion consists of basic transformations such as insertion, deletion and substitution of vertices and edges. By determining the distortion costs, the user can bias the match for or against particular types of distortions.

The Subdue system also provides a user-defined option of using the best substructure found in a discovery step to compress the input graph by replacing those instances of the substructure by a single vertex and performing the discovery process on the compressed graph. The process of finding substructures and compressing the graph can continue until the whole graph is represented by one vertex or the user imposed constraints are exceeded. This feature generates a hierarchical description of the graph dataset at various levels of abstraction in terms of the discovered substructures.

The Subdue knowledge discovery system comes with several auxiliary programs. Some important programs are Inexact graph matcher, Minimum Description Length, distributed MPI Subdue[4] and Subgraph Isomorphism.

3. COMPRESSION-BASED METHODOLOGY

A common characteristic of a majority of the graph-based data mining methodologies described above is that they focus on complete, frequent sub-graph discovery. Complete, frequent sub-graph discovery algorithms such as FSG and gSpan are guaranteed to find all subgraphs that satisfy the user specified constraints. Although completeness is a fundamental and desirable property, one cannot ignore the fact that these systems typically generate a large number of substructures, which by themselves provide relatively less insight about the domain. Typically, interesting substructures have to be identified from the large set of substructures either by domain experts or by other automated methods so as to achieve insights into this domain. Subdue typically produces a smaller number of substructures which best compress the graph dataset. These few substructures which compress the input dataset can provide important insights about the domain.

We demonstrate this advantage of Subdue's compression-based methodology by performing an experimental comparison of Subdue with the graph-based data mining systems FSG and gSpan on a number of artificial datasets. We generate graph datasets with 500, 1000, 1500 and 2000 transactions by embedding one of the six substructures shown in Figure 1. Each of the generated twenty-four datasets (six different embedded substructures, each with 500, 1000, 1500 and 2000 transactions) have the following properties,

1. 60% of the transactions have the embedded substructure, rest of the transactions are generated randomly.
2. For all the transactions that contain the embedded substructure, 60% of the transaction is the embedded substructure, that is, coverage of the embedded substructure is 60%.

Since each of the twenty-four datasets have the properties 1 and 2 discussed above, it is clear that the embedded substructure is the most interesting substructure in each of the datasets. Using these datasets we now compare the performance of Subdue, FSG and gSpan. For each of the twenty-four datasets, Subdue was run with the default parameters and FSG and gSpan were run at a 10% support level. Table

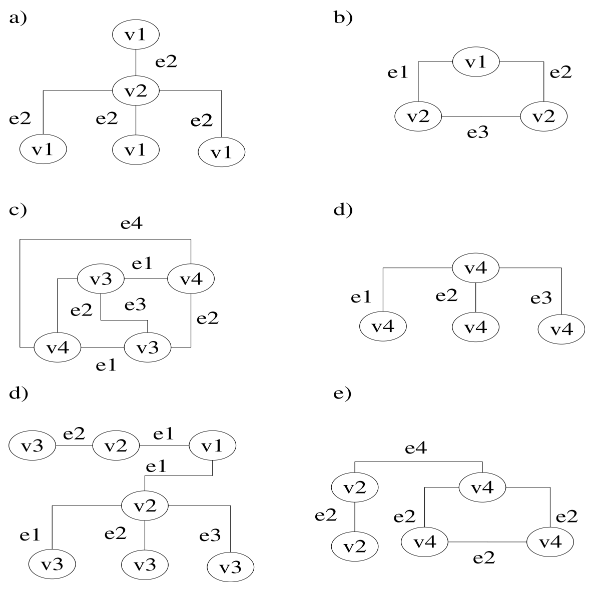

Figure 1: Substructures embedded in the artificial graph datasets. Datasets are generated by embedding one of the six substructures.

Table 1: Results (Average) on 6 artificial datasets each with a different embedded substructure.

Number of transactions	Percent times embedded substructure reported by Subdue as best	Number of substructures generated by FSG/gSpan
500	66%	233495
1000	83%	224174
1500	83%	217477
2000	78%	217479
Average	79%	223156

Table 2: Runtimes (secs.) on 6 artificial datasets each with a different embedded substructure.

Number of transactions	FSG	Subdue	gSpan
500	734	51	61
1000	815	169	107
1500	882	139	182
2000	1112	696	249
Average	885	328	150

1 summarizes the results of the experiments and Table 2 the runtimes of Subdue, FSG and gSpan.

The results indicate that Subdue discovers the embedded substructure and reports it as the best substructure about 80% of the time. Both FSG and gSpan generate approximately 200,000 substructures among which there exists the embedded substructure. The runtime of Subdue is intermediate between FSG and gSpan. Subdue clearly discovers and reports fewer but more interesting substructures. Although it could be argued that setting a higher support value for FSG and gSpan can lead to fewer generated substructures, it should be noted that this can cause FSG and gSpan to miss the interesting pattern. We observed that the increase in run time for Subdue is non-linear when we increase the size of the dataset. Increase for FSG and gSpan was observed to be linear (largely because of various optimizations). The primary reason for this behavior is the less efficient implementation of graph isomorphism in Subdue than in FSG and gSpan. A more efficient approach for graph isomorphism, possibly based on canonical labeling, needs to be developed for Subdue.

4. COMPARISON OF SUBDUE WITH FSG AND GSPAN ON REAL-WORLD DATASETS

We performed an experimental comparison of Subdue with the graph mining systems gSpan and FSG on the chemical toxicity and the chemical compounds datasets that are provided with gSpan. We use these datasets rather than one of the several datasets studied by Subdue researchers in order to perform a fair comparison.

The chemical toxicity dataset has a total of 344 transactions. There are 66 different edge and vertex labels in the dataset. FSG and gSpan results were recorded at 5% support. We found that, if support is set to a lesser value, large numbers of random and insignificant patterns are generated. Table 3 summarizes the results of the experiment. Figure 2 shows the best compressing substructure discovered by Subdue. Subdue discovered best-compressing frequent patterns which were missed by FSG and gSpan. The best substructure by Subdue compressed 8% more than any of the substructures discovered by FSG and gSpan. It is also observed that the runtime of Subdue is much larger than that of FSG and gSpan.

The chemical compounds dataset has a total of 422 transactions. There are 21 different edge and vertex labels in the dataset. The results for FSG and gSpan were recorded at

Table 3: Results on the Chemical Toxicity Dataset.

Compression achieved by best substructure reported by Subdue	16%
Best compression achieved by any of the substructures reported by FSG/gSpan	8%
Number of substructures reported by FSG/gSpan	884
Runtime Subdue (secs.)	115
Runtime FSG (secs.)	8
Runtime gSpan (secs.)	7

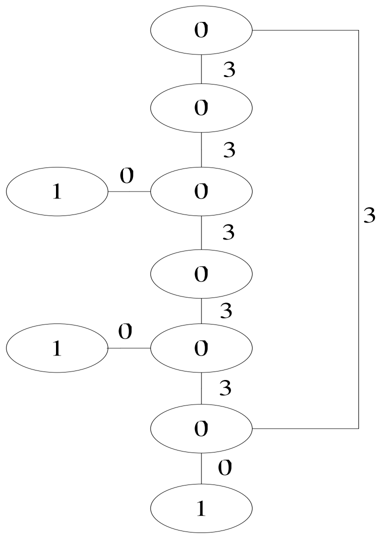

Figure 2: Best Compressing Substructure Discovered by Subdue on the Chemical Toxicity Dataset. The lables on the vertices and edges represent the atom names and bond names. Numbers have been used to represent atom names and bond names as gSpan only accepts numerical lables.

Table 4: Results on the Chemical Compounds Dataset

Compression achieved by best substructure reported by Subdue	19%
Best compression achieved by any of the substructures reported by FSG/gSpan	7%
Number of substructures reported by FSG/gSpan	15966
Runtime Subdue (secs.)	142
Runtime FSG (secs.)	21
Runtime gSpan (secs.)	4

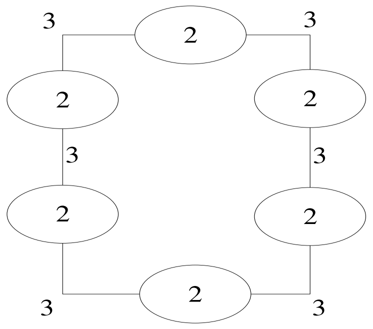

Figure 3: Best Compressing Substructure Discovered by Subdue on the Chemical Compounds Dataset. The lables on the vertices and edges represent the atom names and bond names. Numbers have been used to represent atom names and bond names as gSpan only accepts numerical lables.

10% support. We found that, if support is set to a lesser value, large numbers of random and insignificant patterns are generated. Table 4 summarizes the results of the experiment. Figure 3 shows the best compressing substructures discovered by Subdue. It is observed that Subdue discovered best-compressing frequent patterns which were missed by FSG and gSpan. The best substructure found by Subdue compressed 12% more than any of the substructures discovered by FSG and gSpan. It is also observed that the runtime of Subdue is much larger than that of FSG and gSpan.

5. EXPERIMENTS ON THE MUTAGENESIS AND KDD 2003 CITATION DATASET

The Mutagenesis dataset[18] has been collected to identify mutagenic activity in a compound based on its molecular structure and is considered to be a benchmark dataset for multi-relational data mining. The Mutagenesis dataset consists of the molecular structure of 230 compounds, of which 138 are labeled as mutagenic and 92 as non-mutagenic. The mutagenicity of the compounds has been determined by the Ames Test. The task is to distinguish mutagenic compounds from non-mutagenic ones based on their molecular structure. The Mutagenesis dataset basically consists of atoms, bonds, atom types, bond types and partial charges on atoms. We ran Subdue on the 138 positive examples in the Mutagenesis dataset. Figure 4 shows the best compressing substructure discovered by Subdue which compresses the input data by

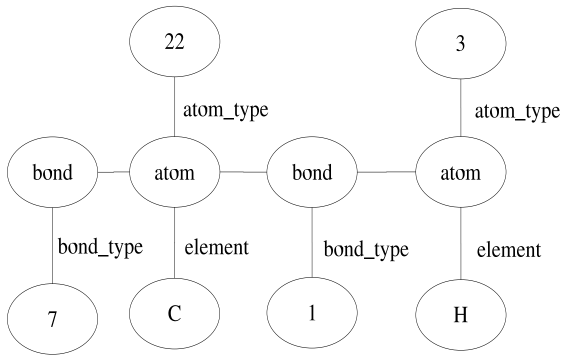

Figure 4: Best Compressing Substructure Discovered by Subdue on the Mutagenesis Dataset

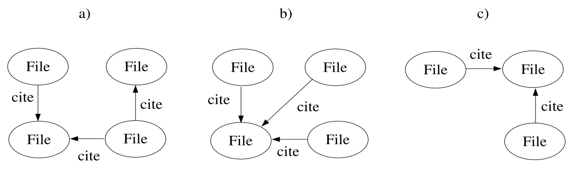

Figure 5: Best Compressing Substructures Discovered by Subdue on the KDD 2003 Citation Dataset

40%. The KDD 2003 Citation dataset consists of citation information from the e-print arXiv, which is the primary mode of research communication in physics. We constructed a citation graph from this data in which each paper was denoted by a single vertex labeled File and a citation between two papers was denoted by a directed edge. The citation graph comprised of a total of 29,014 vertices and 342,437 edges. We ran Subdue on this citation graph. Figure 5 shows the best compressing substructures discovered by Subdue each of which compresses the input data by 9%.

6. CONCLUSIONS AND FUTURE WORK

We presented the graph-based data mining system Subdue which uses a compression-based methodology to frequent subgraph discovery. Subdue can efficiently discover best-compressing frequent patterns which are fewer in number but can be of higher interest. In fact, Subdue may find high-compressing patterns that are less frequent, and therefore missed by the purely frequency-based approach. As a part of our future work we plan to develop better algorithms for graph isomorphism which will improve the run time of Subdue. We also plan to implement canonical labels which would allow implementation of several optimization as in FSG and gSpan. The use of pre-processing schemes (like elliminating vertices and edges with single instances) will also improve the run time of Subdue. Subdue's run time and quality of results is sensitive to the parameters beam and limit. Currently the default values for limit and beam are set based on the total size of the dataset. In certain cases this leads to an unsatisfactory performance as in [11]. Determining optimal values for limit and beam parameters by looking at the graph datasets would be an important direction of work. We also plan to compare Subdue with recent graph mining systems such as Gaston an FFSM.

The Subdue source code and related materials can be downloaded from http://ailab.uta.edu/subdue.

7. ACKNOWLEDGEMENTS

This research is sponsored by the Air Force Research Laboratory (AFRL) under contract F30602-01-2-0570. The views and conclusions contained in this document are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied, of AFRL or the United States Government.

8. ADDITIONAL AUTHORS

Additional authors: Rohan Shah, University of Texas at Arlington, email: shah@cse.uta.edu and Jeffrey Coble, University of Texas at Arlington, email: coble@cse.uta.edu

9. REFERENCES

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

[2] D. J. Cook and L. B. Holder. Substructure discovery using minimum description length and background knowledge. *J. Artif. Intell. Res. (JAIR)*, 1:231–255, 1994.

[3] L. Dehaspe and H. Toivonen. Discovery of frequent datalog patterns. *Data Min. Knowl. Discov.*, 3(1):7–36, 1999.

[4] G. Galal, D. J. Cook, and L. B. Holder. Exploiting parallelism in a structural scientific discovery system to improve scalability. *JASIS*, 50(1):65–73, 1999.

[5] J. Huan, W. Wang, and J. Prins. Efficient mining of frequent subgraphs in the presence of isomorphism. In *ICDM*, pages 549–552, 2003.

[6] A. Inokuchi, T. Washio, and H. Motoda. An apriori-based algorithm for mining frequent substructures from graph data. In *PKDD*, pages 13–23, 2000.

[7] R. D. King, A. Srinivasan, and L. Dehaspe. Warmr: a data mining tool for chemical data. *Journal of Computer-Aided Molecular Design*, 15(2):173–181, 2001.

[8] R. I. Kondor and J. D. Lafferty. Diffusion kernels on graphs and other discrete input spaces. In *ICML*, pages 315–322, 2002.

[9] M. Kuramochi and G. Karypis. Discovering frequent geometric subgraphs. In *ICDM*, pages 258–265, 2002.

[10] M. Kuramochi and G. Karypis. An efficient algorithm for discovering frequent subgraphs. *IEEE Trans. Knowl. Data Eng.*, 16(9):1038–1051, 2004.

[11] M. Kuramochi and G. Karypis. Finding frequent patterns in a large sparse graph. In *SDM*, 2004.

[12] T. Matsuda, T. Horiuchi, H. Motoda, and T. Washio. Extension of graph-based induction for general graph structured data. In *PAKDD*, pages 420–431, 2000.

[13] M. Mukherjee and L. B. Holder. Graph-based data mining for social network analysis. In *Proceedings of the ACM KDD Workshop on Link Analysis and Group Detection*, 2004.

[14] S. Nijssen and J. N. Kok. A quickstart in frequent structure mining can make a difference. In *KDD*, pages 647–652, 2004.

[15] L. D. Raedt and S. Kramer. The levelwise version space algorithm and its application to molecular fragment finding. In *IJCAI*, pages 853–862, 2001.

[16] A. Rakhshan, L. B. Holder, and D. J. Cook. Structural web search engine. In *FLAIRS Conference*, pages 319–324, 2003.

[17] J. Rissanen. *Sochastic Complexity in Statistical Inquiry*. Singapore: World Scientific Publishing, 1989.

[18] A. Srinivasan, S. Muggleton, M. J. E. Sternberg, and R. D. King. Theories for mutagenicity: A study in first-order and feature-based induction. *Artif. Intell.*, 85(1-2):277–299, 1996.

[19] S. Su, D. J. Cook, and L. B. Holder. Application of knowledge discovery to molecular biology: Identifying structural regularities in proteins. In *Pacific Symposium on Biocomputing*, pages 190–201, 1999.

[20] X. Yan and J. Han. gspan: Graph-based substructure pattern mining. In *ICDM*, pages 721–724, 2002.

[21] X. Yan and J. Han. Closegraph: mining closed frequent graph patterns. In *KDD*, pages 286–295, 2003.