Proposal for the Master's Thesis Grammar-based Recompression with Adjacent Digrams

Matthias Dürksen

November 9, 2018

1 Motivation

Nowadays there is more and more data, some of it is stored in graphs. The graphs often become very large and therefore require a lot of memory. To solve this problem you can compress the graph and reduce the memory requirements. In this case a grammar-based graph compression is used. This procedure was already invented by me in a previous work [1].
This procedure [1] has already been extended in several papers. There is a recompression for already compressed graphs and also an efficient coding. For reasons of complexity, a compression possibility, the so-called adjacency digrams, was not supported in the further work, which significantly limits the compression efficiency. The overall aim of this work is to support adjacency digrams also in recompression and coding.

2 Fundamentals

Graph compression supports graphs with nodes and edge labels. With my approach, the graph is first transformed. The result of the transformation is a graph with only node labels and no edge labels, which facilitates compression.
After the transformation comes the actual transformation. Thereby patterns (which is called digram) are searched and the most common pattern/digram is replaced by a nonterminal. We distinguish between two types of digrams, the basic and the adjacency digrams. A basic digram (fig. 1) consists of two adjacent nodes and the edge that connects these nodes. Thus two adjacent nodes and the incident edge can be replaced by one node/nonterminal.
The adjacency digram (fig. 2) is formed by two knots which both have a common adjacency knot x. Thus, the adjacency digram contains the two nodes and the two edges that are incident to the node x. This digram can be replaced by a nonterminal and an

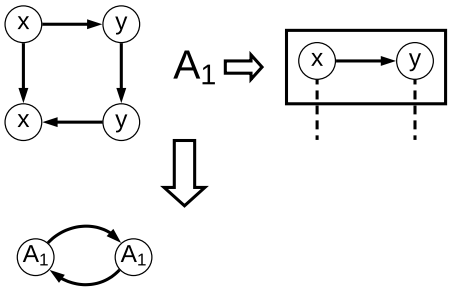

Figure 1: Application of a replacement of a basic digam

edge from the nonterminal to the node x. Thus the possibilities of replacement by the adjacency digram are significantly increased.

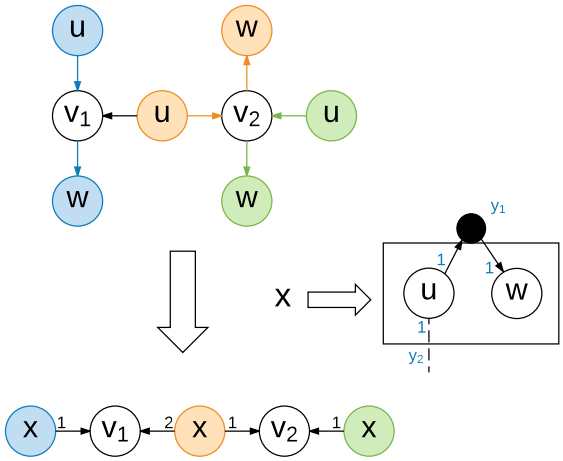

Figure 2: Application of a replacement of a adjacency digam

After the search for the most frequent digram, all overlap-free occurrences are replaced by the nonterminal. Then the search for the most frequent digram and the replacement of the occurrences is continued until there are no more digrams which occur frequently enough.

3 This Thesis

These compressed graphs and the corresponding grammar rules can be coded on the basis of Georgi's work [3] and thus save the graph in a memory efficient way. However, this coding is only designed for graphs where there have been no replacements based on the adjacency digrams. Therefore it is part of the work to extend this coding to support graphs where adjacency digrams have been replaced.
Furthermore, the works of Frerk [2] and Werneke [4] have developed an extension for the recompression of graphs. This method can be used if, for example, two subgraphs

were compressed individually and these graphs are compressed together. This method combines the two graphs and checks whether the grammar rules can be chosen more efficiently in the combination of the two graphs than to combine the previous rules. This method can also be used if an existing graph has been modified and thus has new nodes or old nodes have been removed. Thus it is checked again whether there is now an efficient compression.
Frerk's approach is based on the graph model and Werneke's methods more on memory level, so that Georgi's coding is considered here. However, it also only supports the basic digrams Therefore, the procedure of [4] is to be extended so that also here the adjacency digrams are supported.
However, Frerk has already discussed and supported the treatment in his work, so it is also necessary to examine which parts of the Frerk procedure can be applied to the other procedure. (Need for clarification with the supervisor)

References

[1] Matthias Duerksen. “Grammatik-basierte Graphkompression”. Bachelor’s Thesis. University of Paderborn, 2016.

[2] Philip Frerk. “Grammatik-basierte Rekompression angewendet auf Graphen”. Bachelor’s Thesis. University of Paderborn, 2017.

[3] David Georgi. “Kompakte Kodierung für grammatikbasierte Graphenkompression”. Bachelor’s Thesis. University of Paderborn, 2017.

[4] Lars Werneke. “Grammatikbasierte Rekompression von Graphen”. Bachelor’s Thesis. University of Paderborn, 2018.