Calculating Treedepth on a GPU- introduction

Dymitr Lubczyk Wojciech Replin Bartosz Różański

Contents

1 Introduction

Tree-depth definition

In graph theory, the tree-depth is defined for undirected graphs. Intuitively, we may think of it as parameter that says how similar are G and a star, the lower tree-depth is the more "starlike" G is. Its value is between 1 and |G|. To explain the most common tree-depth definition, it is necessary to introduce term tree-depth decomposition. So tree-depth decomposition of graph G is a forest F with the following property:

If there is a edge uv in G then verticies u,v have ancestor-descendant relationship to each other in F.

So the tree-depth of G is a depth of the forest with minimal depth among all tree-depth decomposition of G.

There is also recursive definition, that is worth to mention because of dynamic algorithm, that we are going to present in futher part of this documentation. The definition looks as follows:

$$td(G) = \begin{cases} 1 & \text{if } |G|=1 \\ 1 + min_{v \in V} td(G - v) & \text{if G is connected} \\ max_i td(G_i) & \text{otherwise} \end{cases} \tag{1}$$

Examples

In this paragraph I will provide some obvious facts and examples of tree-depth decomposition in order to give reader a better intuition.

Only connected graph with tree-depth equal to 1 is K_1 and stars are only connected graphs with tree-depth equal to 2 and for these graphs graph and its best tree-depth decomposition is the same thing.

Antoher thing is that for every graph G a tree-depth depcomposition of G is a path P(where |P|=|G|) rooted in its beginning. It is true, because for such a tree every pair of vertices have ancestor-descendant relationship to each other.

The last fact, that I will point out is that if G is connected then its tree-depth decomposition F is also connected. It is true, because if F had two compositions, then there would be two groups of vertices in G without edges between each other, which is contradictiory to G being connected.

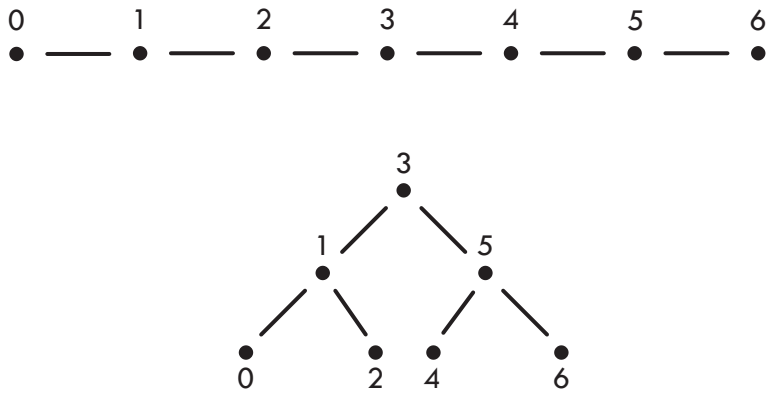

Figure 1: P_7 and its tree-depth decomposition

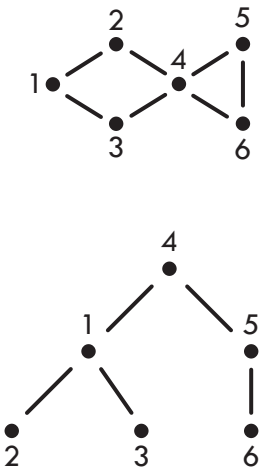

Figure 2: Fish and its tree-depth decomposition

2 Typical approaches

3 Our approach

4 Proposed modifications

5 Expected results

General thoughts

As we mentioned before we are going to try a few variants with diffrent degree of complexity and in this point we are not sure if we manage to accomplish all of them. The most uncertain variant is bound and branch algorith on GPU as this alogorithm seems to be very hard to be made parallel. As a result it may not be possible to provide GPU implementation for hybrid algorithm, but it is not sure, we will try to figure it out.

Dynamic

We most likely will manage to do dynamic algorithm both GPU and CPU implementation and the GPU implementation of dynamic algorithm will probably perform very well as long as memory will allow it to do so. We predict that the limits on memory will be reached for graphs with around 30 vertices. The exact value depends on union-find structure implementation.

Bound and branch

This algorithm has higher complexity than dynamic algorithm and most likely it will perform worse than dynamic algorithm, however it does not have limits on memory as its memory complexity is polinomial. It is very possible that we will manage to provide CPU implementation, but because of its recursive nature it may cause a lot of trouble to accomplish GPU one.

Hybrid

This algorithm has similar complexity to Bound and branch algorith as Bound and branch is main part of it, but we assume that it will perform way better than its basic version because of *endings* precalculated by dynamic algorithm. Most likely we will provide CPU implementation. Unfortunately, we think that the GPU implementation is very unlikly to be accomplished.