

DAIA: a decompose and improve algorithm for treedepth decomposition

Stéphane Grandcolas

Laboratoire d'Informatique et des Systèmes, Aix-Marseille université
stephane.grandcolas@lis-lab.fr

Abstract

DAIA is a two-phases heuristic algorithm that searches for good treedepth decompositions of graphs. First it builds a treedepth decomposition partitionning recursively the vertices. Then it modifies the resulting tree in order to reduce its height.

Keywords and phrases Treedepth

Funding *Stéphane Grandcolas*: <https://www.lis-lab.fr/>

DOI 10.5281/zenodo.3884054

<https://gitlab.lis-lab.fr/stephane.grandcolas/pace-2020>

1 Approach

Given a connected graph G the problem is to build a rooted tree whose nodes correspond to the vertices of G and such that, for each edge (u, v) of G , the nodes corresponding to u and v are in the same branch (that is either u is an ancestor of v or v is an ancestor of u). The objective is to minimize the height of the tree. We propose a method that consists in first building recursively a treedepth decomposition, then in improving the tree relocating subtrees. No preprocessing is applied to simplify the graph or to detect particular structures. The process can be reiterated as long as a given time limit is not reached.

2 Initial decomposition

The process is based on the partitioning of the vertices of subgraphs of G . If $G' = (V', E')$ is a subgraph of G , *separation* consists in partitioning V' in three sets A , B and S , such that there is no edge of E' between A and B . S is called the *separator*. A treedepth decomposition of G' is obtained decomposing recursively the subgraphs induced by A and B , and connecting the resulting trees as subtrees of a path-branch built with the vertices of S . H. Althoby et al. [1] describe a simple greedy heuristic to solve the vertex separation problem (VSP). The idea is to transfer the vertices from S or B to A , starting with all the vertices in B but one, which is in S . At each step a vertex of S with as few neighbors in B as possible is chosen and moved to A , and its neighbors which are in B are moved to S . For the VSP the objective is to minimize the size of the separator. For treedepth decomposition the goal is also that the decomposition of the subgraphs induced by A and B give low trees, preferably of the same heights. We have developed the greedy heuristic as follows:

- several *flushes* are made, moving the vertices from B to A , then from A to B , and so on,
- during the process the separation is evaluated and the best one is memorized.

We use two evaluations. The first one estimates roughly the height of the tree:

$$evalCG(S, A, B) = size(S) + \sqrt{\max\{\#Edges(A), \#Edges(B)\}}$$

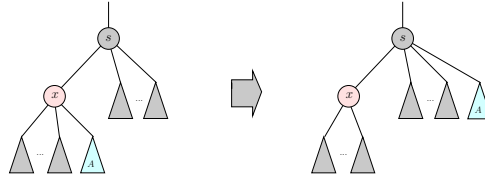
The other aims at generating sets of similar sizes and a small separator:

$$evalSV(S, A, B) = \frac{card(S) + 1}{\min\{size(A), size(B)\} + 1}$$

Before each partitioning an evaluation function is chosen at random. Then a number of separations are performed, each with a number of flushes and the same evaluation function.

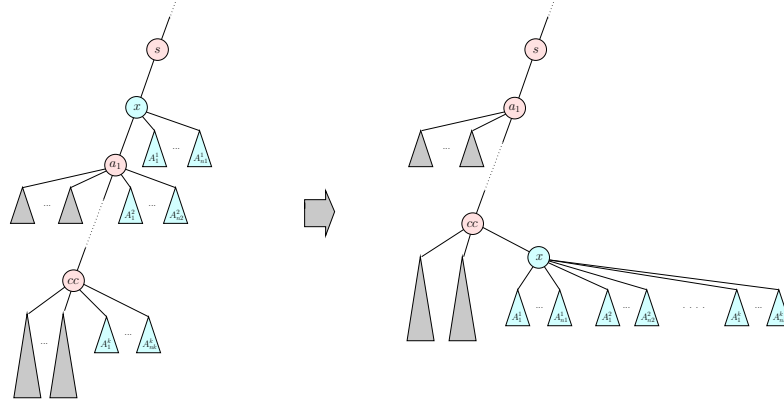
3 Improvement

Two operations are implemented to make the tree lower. Both concern *critical nodes*, that is nodes whose subtrees contain all the nodes of the tree which are at the maximal depth (thus removing a critical node gives a lower tree). The first operation consists in pulling up subtrees that are *independent* of their nearest ancestor, see figure 1 (a subtree is *independent* of a given node if it contains no neighbor of the node).



■ **Figure 1** Pullup: there is no neighbor of x in A , A is moved up.

The second operation consists in ejecting a critical node x to a lower position, carrying with it the subtrees with which it is dependent (figure 2, the blue subtrees are dependent of x). It requires to search x *critical correspondent* cc , the lowest critical node that has subtrees dependent of x . The height of the tree decreases if the maximal height of the subtrees dependent of x plus two is not greater than the height of the subtree rooted in cc before the operation.



■ **Figure 2** Ejection: node x is moved under its critical correspondent with its dependent subtrees

To improve a given treedepth decomposition, we propose first to pullup the subtrees whose nearest ancestor is a critical node while there are some, then to eject critical nodes as long as it reduces the height of the tree.

References

- 1 Haeder Y. Althoby, Mohamed [Didi Biha], and André Sesboüé. Exact and heuristic methods for the vertex separator problem. *Computers & Industrial Engineering*, 139:106135, 2020. URL: <http://www.sciencedirect.com/science/article/pii/S0360835219306047>, doi:<https://doi.org/10.1016/j.cie.2019.106135>.