1. Introduction

Component trees [1]:
- Represent the hierarchy of connected components (CCs) of the level sets of a grayscale image;
- Efficiently represented by max-trees and widely used;

Component-hypertrees [2]:
- Extension of component trees using increasing connectivities;
- Efficient ways of updating hypertrees from previous connectivities have been presented [3];
- Goal: an efficient way of representing hypertrees.

2. Proposed Method

Suppose a grayscale image f and a sequence \((A_1, \ldots, A_n)\) of increasing sets of neighboring pixels are given. Then, the proposed algorithm follows the template below:
1. Initialize parent (array representing a max-tree);
2. Initialize the hypertree (empty at the beginning);
3. For \(1 \leq i \leq n:\)
   (a) For \((p, q)\) neighbors in \(A_i:\)
      i. Update parent by connecting \(p\) and \(q\) [4].
         Track changes by marking nodes and arcs;
   (b) Update the hypertree by allocating marked nodes and arcs.

3. Minimal Component-Hypertree

Minimal component-hypertree is the smallest graph satisfying:
1. No repeated nodes;
2. All inclusion relations;
3. Pixels stored only once;
4. Nodes do not depend on nodes with higher connectivity index to be reconstructed.

4. Results

Time consumption:
- Updating the max-tree is the most time consuming step;
- Only 3% to 6% of time used to allocate structures.

Number of nodes and arcs compared to other representations:

5. Conclusion

- An efficient way of computing and storing hypertree was presented;
- Computation of attributes in this structure will be presented in a later date.

6. Acknowledgements

- This study was financed in part by the CAPES - Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (Finance Code 001);
- FAPESP - Fundação de Amparo a Pesquisa do Estado de São Paulo (Proc. 2018/15652-7);
- CNPq - Conselho Nacional de Desenvolvimento Científico e Tecnológico (Proc. 428720/2018-8).

7. References