A comparison of many max-tree computation algorithms

Edwin Carlinet, Thierry Géraud

EPITA Research and Development Laboratory (LRDE), France

edwin.carlinet@lrde.epita.fr, thierry.geraud@lrde.epita.fr

At a Glance

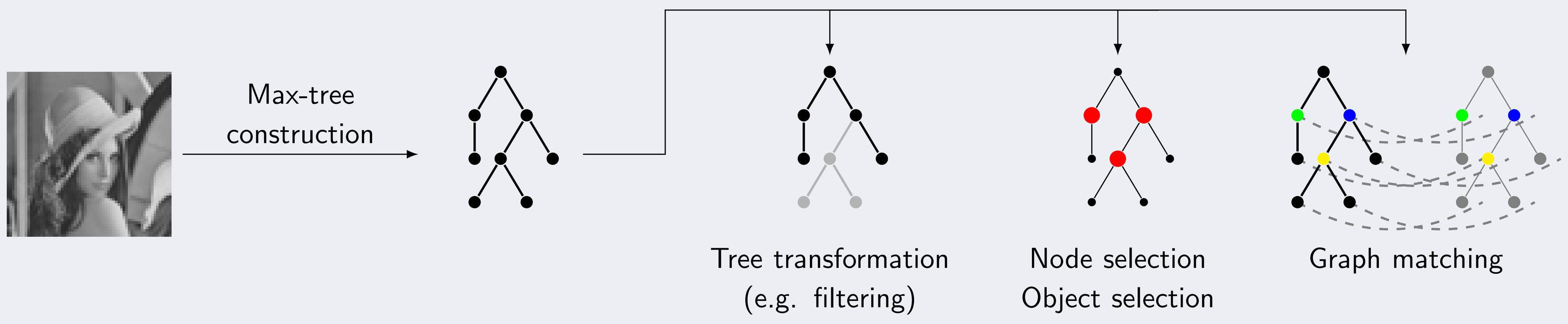
Issue Many max-tree algorithms, some of them being specific to a dedicated task (e.g. filtering)

Goal Comparisons of max-tree algorithms have been attempted, but they are partial.

Contribution Provide a full and fair comparison of 5 max-tree algorithms and some variations in a common framework, i.e., same hardware, same language (C++) and same outputs.

Code & Demo http://www.lrde.epita.fr/Olena/maxtree

Typical workflow



Desired properties

We aim at providing a "usable" tree, i.e.:

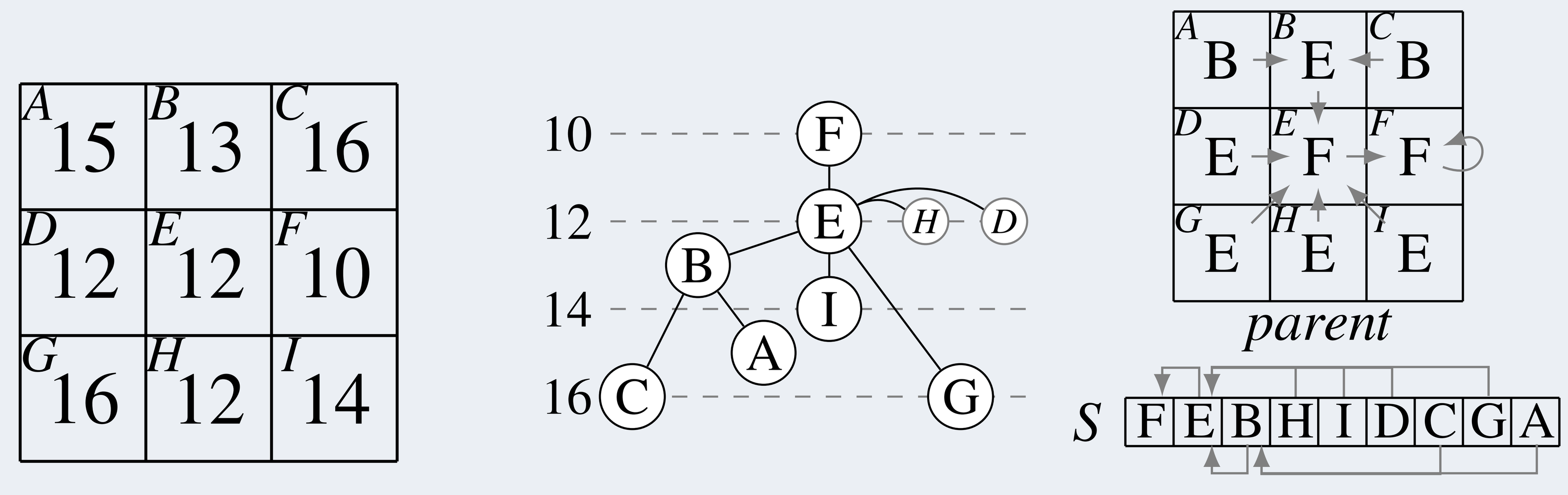
(1) Direct access to parent: Access any node's parent in constant time.

(2) Direct access to nodes: Access any pixel's node in constant time.

(3) Downward and upward traversable: Get a processing order of the nodes.

Many algorithms output a tree structure that does not satisfy (2) or (3).

Max-tree representation



Original image ima — Max-tree of ima — Max-tree representation with a parent image and an array

- A node is represented by a single pixel *(canonical element)*
- A $parent$ image encodes the *parent* relationship of the tree.
- Every pixel points to a canonical element (ensures properties (1) and (2))
- An S vector stores the pixels ordered *downward* (ensures property (3))

Competitors

Immersion algorithms. Methods based on Tarjan's Union-Find.
Competitors: Berger et al. (2007), Najman and Couprie (2006), and 2 variations of the first one.

Flooding algorithms. Methods based on a depth-first propagation using priority or hierarchical queues.
Competitors: Salembier et al. (1998), Nistér and Stewénius (2008), Wilkinson (2011)

Merge-based algorithms. Methods that compute trees on sub-domains of the image and merge them in a map-reduce fashion.
Competitors: Matas et al. (2008), + any of the above algorithms.

Time and space complexities

Algorithm	Time complexity: Small int	Large int	Generic V	Auxiliary space requirement: Small int	Large int	Generic V
Berger (Berger et al., 2007)	$O(n\log n)$	$O(n\log n)$	$O(n\log n)$	$n+k+O(n)$	$2n+O(n)$	$n+O(n)$
Berger + rank	$O(n\,\alpha(n))$	$O(n\log\log n)$	$O(n\log n)$	$3n+k+O(n)$	$4n+O(n)$	$3n+O(n)$
Najman and Couprie (2006)	$O(n\,\alpha(n))$	$O(n\log\log n)$	$O(n\log n)$	$5n+k+O(n)$	$6n+O(n)$	$5n+O(n)$
Salembier et al. (1998)	$O(nk)$	$O(nk)\simeq O(n^2)$	N/A	$3k+n+O(n)$	$2k+n+O(n)$	N/A
Nistér and Stewénius (2008)	$O(nk)$	$O(nk)\simeq O(n^2)$	N/A	$2k+2n$	$2k+2n$	N/A
Wilkinson (2011)	$O(n\log n)$	$O(n\log n)$	$O(n\log n)$	$3n$	$3n$	$3n$
Salembier non-recursive	$O(nk)$	$O(n\log\log n)$	$O(n\log n)$	$2k+2n$	$3n$	$3n$
Map-reduce	$O(A(k,n))$	$O(A(k,n))+O(k\sqrt{n}\log n)$		$\ldots+n$	$\ldots+n$	$\ldots+n$
Matas et al. (2008)	$O(n)$	$O(n)+O(k\sqrt{n}(\log n)^2)$		$k+n$	$2n$	$2n$

With n the number of pixels, k the number of values.

Comparison of sequential algorithms

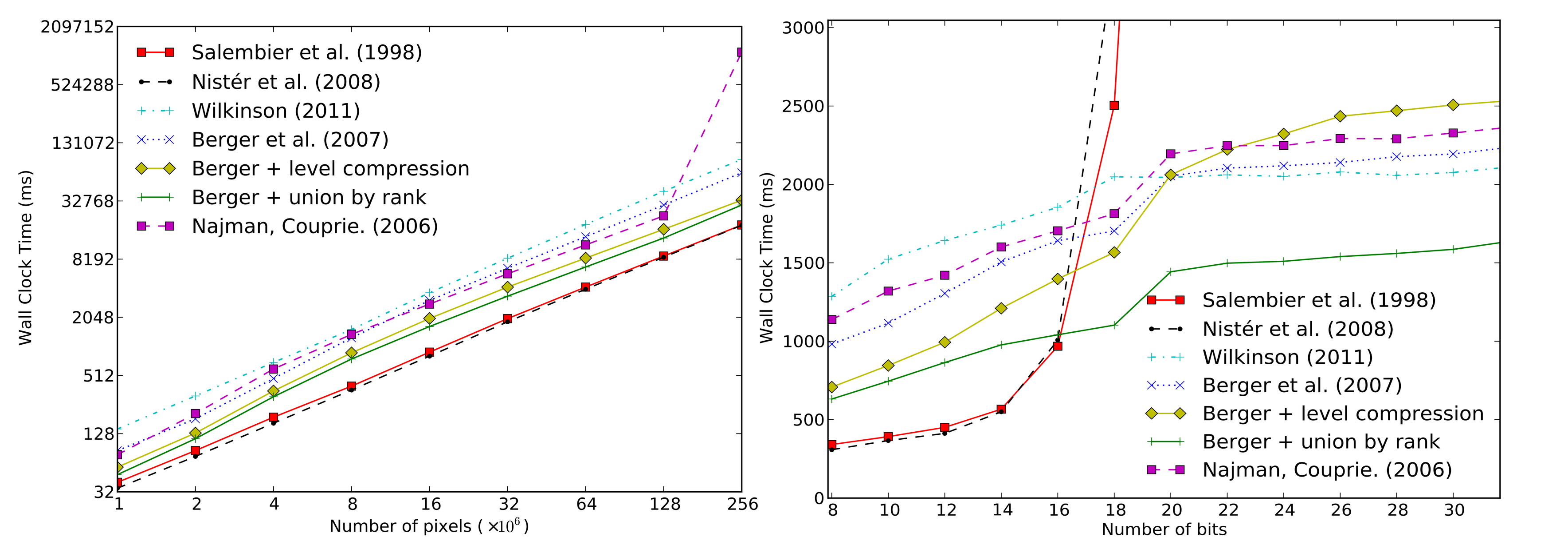


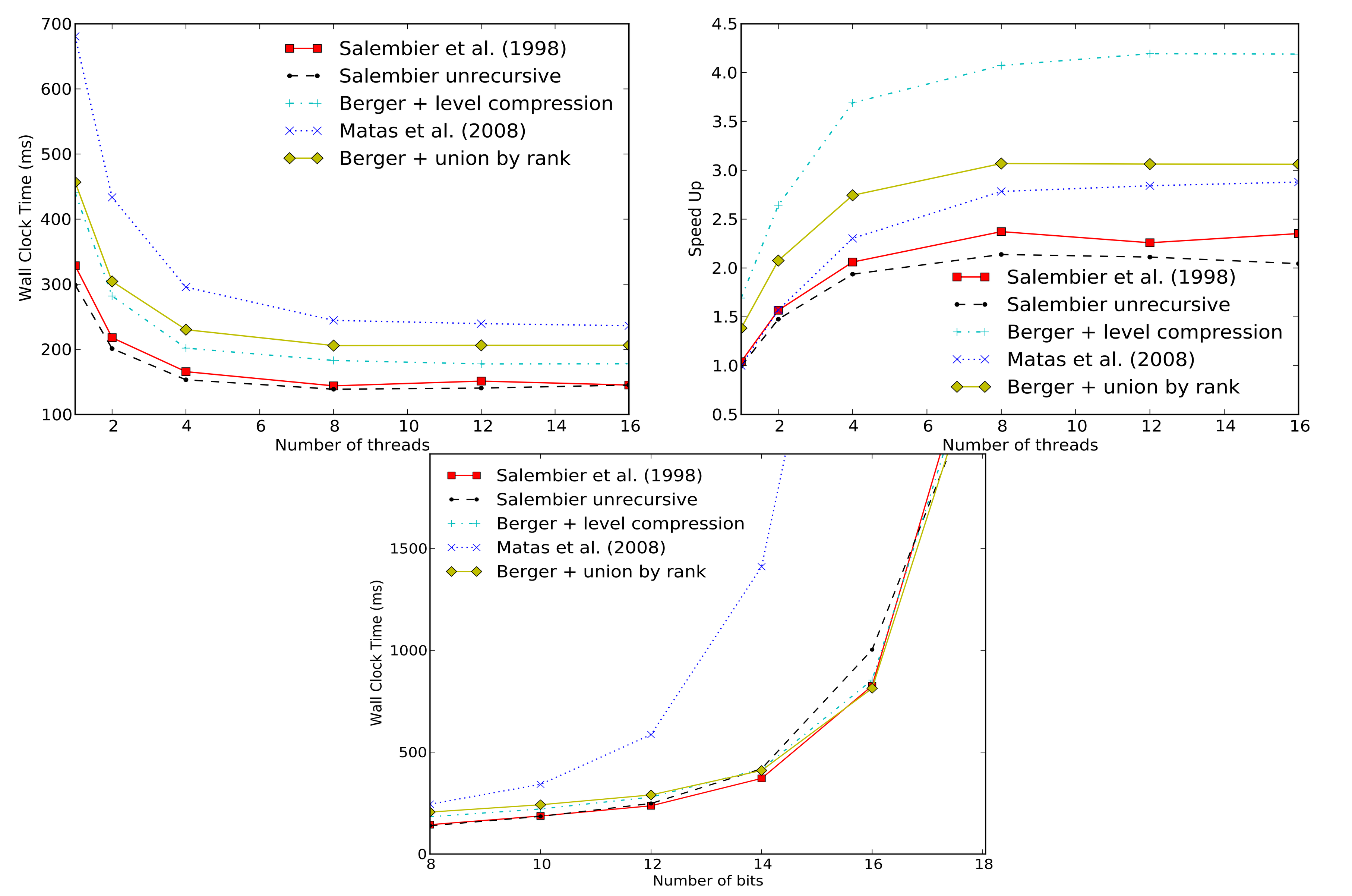
Figure: Algorithms comparison on a 8-bit image as a function of the size (left) and the quantization (right).

Decision tree

Embedded system ? (memory limitation)

- no → Low quantization ?
  - yes → Salembier et al. (1998) or Nistér and Stewénius (2008) (± parallelism)
  - no → Berger + rank
- yes → Low quantization ?
  - yes → Berger + level compression (± parallelism)
  - no → Berger et al. (2007) or Wilkinson (2011)

Comparison of parallel algorithms



Wall clock time (upper left) and speed up (upper right) of the parallelization w.r.t the number of threads. Bottom: algorithms comparison using 8 threads w.r.t the quantization.

References

Berger, C. et al. (2007). "Effective component tree computation with application to pattern recognition in astronomical imaging". In: *Proc. of ICIP*. Vol. 4, pp. IV–41.

Matas, P. et al. (2008). "Parallel algorithm for concurrent computation of connected component tree". In: *Adv. Concepts for Intelligent Vis. Sys.* Pp. 230–241.

Najman, L. and M. Couprie (2006). "Building the component tree in quasi-linear time". In: *IEEE Transactions on Image Processing* 15.11, pp. 3531–3539.

Nistér, D. and H. Stewénius (2008). "Linear time maximally stable extremal regions". In: *Proc. of European Conf. on Computer Vision*, pp. 183–196.

Salembier, P. et al. (1998). "Antiextensive connected operators for image and sequence processing". In: *IEEE Transactions on Image Processing* 7.4, pp. 555–570.

Wilkinson, Michael H. F. (2011). "A fast component-tree algorithm for high dynamic-range images and second generation connectivity". In: *ICIP*, pp. 1021–1024.

ISMM 2013, Uppsala, Sweden, May 27-29, 2013