Multi-label segmentation of images with partition trees

Emmanuel Maggiori, Yuliya Tarabalka, Guillaume Charpiat

Multi-label segmentation of images with partition trees

Emmanuel Maggiori, Yuliya Tarabalka, Guillaume Charpiat

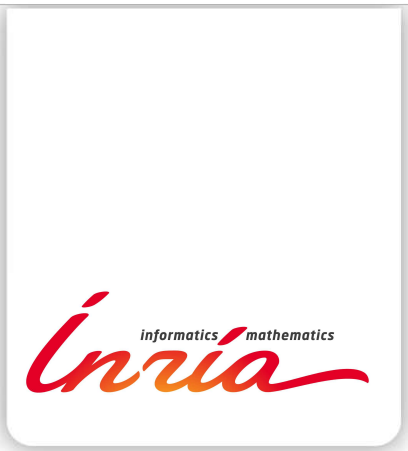
RESEARCH REPORT

N° 1

November 12th, 2014

Project-Teams AYIN and STARS

ISSN 0249-6399 ISRN INRIA/RR--1--FR+ENG



Multi-label segmentation of images with partition trees

Emmanuel Maggiori, Yuliya Tarabalka, Guillaume Charpiat

Project-Teams AYIN and STARS

Research Report n° 1 — November 12th, 2014 — 21 pages

Abstract: We propose a new framework for multi-class image segmentation with shape priors using a binary partition tree. In the literature, such trees are used to represent hierarchical partitions of images, and are usually computed in a bottom-up manner based on color similarities, then analyzed to detect objects with a known shape prior. However, not considering shape priors during the construction phase induces mistakes in the later segmentation. This paper proposes a method which uses both color distribution and shape priors to optimize the trees for image segmentation. The method consists in pruning and regrafting tree branches in order to minimize the energy of the best segmentation that can be extracted from the tree. Theoretical guarantees help reducing the search space and make the optimization efficient. Our experiments show that the optimization approach succeeds in incorporating shape information into multi-label segmentation, outperforming the state-of-the-art.

Key-words: Partition trees, multi-class segmentation, shape priors, graph cuts

Email addresses of the authors are {firstname.lastname@inria.fr}

RESEARCH CENTRE
SOPHIA ANTIPOLIS – MÉDITERRANÉE

2004 route des Lucioles - BP 93
06902 Sophia Antipolis Cedex

Segmentation d'images avec des arbres de partition

Résumé : Les arbres de partition constituent une représentation hiérarchique d'images dont des segmentations peuvent être extraites efficacement. Les arbres peuvent inclure des descripteurs de la forme des régions, ce qui peut améliorer les segmentations que l'on en obtient. Le processus traditionnel de construction de ces structures est *bottom-up*, ce qui a comme effet la propagation et amplification des erreurs commises aux échelles inférieures. Ce travail présente une méthode pour optimiser les arbres en les ébranchant et les greffant, permettant ainsi d'accomplir la segmentation efficace des images avec des contraintes sur les couleurs ainsi que sur la forme des objets.

Mots-clés : Arbres de partition, segmentation, contraintes de forme, coupe de graph

Contents

1 Introduction and related work

The multi-label segmentation of images is one of the great challenges in computer vision. It consists in the simultaneous partitioning of an image into regions and the assignment of labels to each of the segments. The problem can be posed as the minimization of an energy with respect to a set of variables which can take one of multiple labels. Throughout the years, several efforts have been done in the design of algorithms that minimize such energies [24].

The introduction of shape descriptors into segmentation significantly improves its quality [6]. However, it is difficult to optimize energies that involve shape priors because of their non-local nature [18]. The state-of-the art methods require either the design of an optimizer specific to the particular shape prior [13, 30], or a complex way of incorporating it, when possible, into energies minimizable by standard techniques [7, 39].

Hierarchical segmentations of images are of particular interest, as objects are often easily characterizable by their parts. Binary partition trees (BPTs) [27] are a particularly efficient structure for this purpose, both in space and computational complexities. Segmentations can be extracted from such a tree by selecting scales in different branches.

In this work we exploit the idea of iteratively optimizing the structure of BPTs to produce better partitions with shape constraints. This approach is powerful in the sense that modifications can be performed at multiple scales, possibly involving many pixels at the same time.

1.1 Shape and optimization

In the literature regarding shape features in segmentation, we can distinguish contributions that focus on explicit shape models in a *template matching* manner and others geared at incorporating discriminative features (*e.g.,* , convexity, compactness).

The problem of template matching has been studied in the active contours framework [5, 19]. The evolution of the curves is usually slow and prone to get trapped in poor local minima.

Graph-based methods can optimize only specific families of energies, which makes shape priors hard to express [18]. A first family of works propose iterative schemes to tackle shape features. To include an elliptical prior, the authors of [31] proposed to fit ellipses on minimal cuts. In [10], a method to match templates was proposed, which must be run repeatedly to account for any non-rigid deformations. To iteratively estimate the contours, deformable models [14] and *layered pictorial structures* [15] have been coupled with Markov random fields. The computational complexity of these approaches is high: every loop contains expensive operations.

A second family of contributions is aimed at guaranteeing globally optimal solutions. To perform template matching with deformations, in [9], dynamic programming was used on a specially constructed graph that exploits the properties of a triangulation of the template. The authors of [28] proposed to look for minimal ratio cycles in the product graph of the input image and the shape template. These methods find a globally optimal segmentation in polynomial time on the graphs. The authors of [18] allow the user to provide an arbitrary number of prototypical shapes, which should be clustered hierarchically to enjoy the benefits of a branch-and-bound optimization.

In the context of discriminative shape features, star-shaped [39], compactness [7, 11] or convexity [13] priors can be introduced in energies minimizable by traditional s-t cuts. Such approaches rely either on the ability to express the shape prior in the pairwise interaction terms of a Markov random field, or on the design of a specific optimizer for it. However the way of incorporating shape priors in graph-based methods tends to be algorithmically complex (*e.g.,* evaluating a rotating discrete line around a used-defined point [39]). Moreover it is unfeasible to combine several shape priors under the same scheme.

1.2 Hierarchical trees

A BPT [27] is a data structure to represent images as a hierarchy of regions. Multi-label segmentations can be extracted efficiently by performing horizontal cuts on such trees. Recent works have explored the use of shape descriptors during the cuts [37, 40].

BPTs are constructed by successively merging regions with similar colors. Even though BPTs can constitute a good hierarchical approximation to the underlying structure of an image, the bottom-up approach propagates and amplifies the errors produced at the lower scales. As a result, it is very likely that nodes in the BPT will not represent complete significant objects [22, 40]. Previous works mitigated this problem by including additional penalties, like the growth of perimeters [40], the elongation of the regions [17] or edge information [34]. The authors of [40] fitted templates on partially detected objects. These approaches can only alleviate the effect.

As another consequence of the bottom-up approach, shape information cannot be used during construction: the ultimate shape of an object in a branch cannot be predicted by a portion of it. This leads to using different criteria at the construction and at the processing of the trees (as in [37, 40]), which limits the feasibility of the tool *as it is* to perform segmentation with shape prior.

1.3 Optimization of hierarchical trees

To our best knowledge, there are no previous contributions addressing the optimization of BPT structures. Possibly the most related work has been done in the area of computational phylogenetics, on the construction of *phylogenetic trees*. These trees represent the evolutionary relationships among species [20]. Several traditional optimization algorithms (hill climbing, simulated annealing, genetic algorithms) have been applied on the tree structures [12]. The standard moves (known as *branch-swapping* or *swappers*) are nearest neighbor interchange, subtree pruning and regrafting (SPR) and tree bisection and reconnection (TBR). SPR is the pruning/paste of a subtree into another location, while TBR rearranges the subtree before pasting. The most common objective is to maximize the *parsimony* of the tree, *i.e.* to explain the observed data with the least evolutionary change. We have incorporated the idea of regrafting tree branches. Our optimization objective is however different, and our context requires to define moves that preserve the parent, child and spatial adjacency relations of BPTs.

1.4 Contributions

Our **main contributions** are:

- the optimization of partition trees for multi-class image segmentation,
- a theoretical study leading to the reduction of the space of possible moves, making this optimization efficient,
- the design of efficiently computable shape features.

In Section 2 we pose segmentation as an energy optimization problem, based on probability distributions of color and shape features. For this, we propose efficiently computable shape features. In Section 3 we describe how to build an initial BPT and how to find the best partition representable by this tree. In Section 4 we describe a family of moves to modify the tree. An optimization algorithm is designed on these moves to lower down the energy of the possible partitions. In particular, theoretical properties help increasing the speed of the optimization by reducing the space of moves to be explored. We experimentally show in Section 5 that the

optimization algorithm substantially improves the trees and efficiently produces segmentations that take into account the shape priors.

2 Segmentation as an optimization problem

Let $I = (I_j)_{1 \leqslant j \leqslant n}$ be an input image containing n pixels. We suppose we are given a set of possible object classes, as well as priors for each class. Multi-label segmentation consists in an exhaustive partitioning of the pixels into a non-overlapping set of regions $\mathcal{R} = (R_i)$, together with associated class labels $\mathcal{L} = (L_i)$. It can be stated as an optimization problem: minimize

$$E(\mathcal{R}, \mathcal{L}) = E_C(I, \mathcal{R}, \mathcal{L}) + \sum_{i=1}^{|\mathcal{R}|} E_S(R_i, L_i), \tag{1}$$

where E_C expresses the color prior (quantifying how the segmentation fits the image colors), and E_S, the shape prior.

2.1 Color prior

For each object class, we suppose we are given training examples, from which the color distribution can be estimated and used as a prior. One way to do this is to train classifiers based on the samples' colors, using support vector machines (SVM), and to extend them to output probability estimates $P(L_j|I_j)$ as usual in classification problems [41].

Given a candidate segmentation $(\mathcal{R}, \mathcal{L})$, let us denote by $i(j)$ the region containing a pixel j. The color prior is then:

$$E_C(I, \mathcal{R}, \mathcal{L}) = \sum_{j=1}^{n} -\log P(L_{i(j)}|I_j). \tag{2}$$

2.2 Low-complexity shape features in BPTs

Similarly, the shape prior term is defined as follows:

$$E_S(R_i, L_i) = -|R_i| \log P(L_i|\mathbf{S}_i), \tag{3}$$

$|R_i|$ being the area of region R_i, and $P(L_i|\mathbf{S}_i)$ being the probability of assigning the label L_i to the region R_i, given a vector $\mathbf{S}_i$ of shape features of that region. Traditional regularization (such as boundary length [24]) can be incorporated as part of this term. The weight on the area makes the per-pixel contribution of the color prior and the per-region contribution of the shape prior equally important.

We wish to augment the nodes of BPTs by including shape information of the corresponding regions. Given that the optimization of the trees will involve recomputing region descriptors, we must design a pool of features that can be computed efficiently from children nodes. In addition, the errors in estimating the shape descriptors at the finer levels should not be amplified in the upper ones.

Area can be efficiently computed by adding the areas of the children. **Rectangularity** and **elongatedness** shape descriptors have been used in the context of hierarchical methods [17, 33, 37], though no details on how to efficiently implement these features have been given. A rectangle B of dimensions $h \times w$ ($h \geqslant w$), is said to be the **minimum area enclosing rectangle** (MAR) of a region R if it is the rectangle of minimal area that entirely contains

R. Rectangularity measures the resemblance to a rectangle, and is computed as $|R|/(h \cdot w)$. Elongatedness measures the resemblance to a line, and is defined as w/h.

We propose to store the **convex hull** of the region at every node. When two regions are merged, the convex hull of the new region can be computed by merging the convex hulls of its children. The operation of merging convex hulls can be executed in linear time on the size of the input polygons by using *rotating calipers* [35]. The MAR can be efficiently computed from the convex hull by using a similar algorithm [35], and then rectangularity, elongatedness and other descriptors like **solidity** [42] are directly derived from it. In a balanced tree, the convex hulls can incur in an $O(n \log(n))$ increase of the storage required. Their computation for all the nodes in a tree does not increase the complexity of its construction (proofs in appendix).

In a discrete environment, the errors of computing the area converge as the areas go larger, so do the convex hulls. As a consequence, the previous features tend to be more precise in the upper levels of the trees.

Another useful shape descriptor is **compactness** [23], related to the resemblance to a circle. It is typically defined as $\delta R^2/(4\pi|R|)$, where δR is the perimeter of R. However, this formulation imposes difficulties in a discrete environment [23] due to the fact that the error in the estimation of δR does not converge. In [21] the authors proved the robustness of computing compactness as $|R|^2/(2\pi I_g)$, I_g being the moment of inertia of the shape with respect to its centroid. Given that the centroid and moment of inertia of a region can be computed in constant time from the children, we propose this method to measure compactness in BPTs.

Let us suppose a probability density function $p(S|L)$ is available for every feature and class. These densities can be obtained by smoothing histograms of training samples [29]. Assuming independence of the features $\mathbf{S} = s_1, ..., s_m$,

$$P(L|\mathbf{S}) = \prod_{i=1}^{m} P(L|s_i). \tag{4}$$

Bayes' theorem and the density functions can be used to compute the posterior probabilities required in (4):

$$P(L|s_i) = \frac{p(s_i|L)P(L)}{\sum_{L_j \in \mathcal{L}} p(s_i|L_j)P(L_j)}. \tag{5}$$

Combining Eq. (1), (2) and (3), the energy criterion to minimize is formulated as $E(\mathcal{R}, \mathcal{L}) :=$

$$\begin{aligned} &-\sum_{i=1}^{|\mathcal{R}|} \Big(\sum_{j \in R_i} \log P(L_i|I_j) + |R_i| \log P(L_i|\mathbf{S}_i) \Big) \\ &= -\sum_{j=1}^{n} \Big(\log P(L_{i(j)}|I_j) + \log P(L_{i(j)}|\mathbf{S}_{i(j)}) \Big). \end{aligned} \tag{6}$$

The first form of the energy expresses it as an independent sum over regions, while the second one emphasizes the cost per pixel (which depends on the region containing it).

3 Building and processing a partition tree

In this section we show how to build a first candidate solution to the problem (6), in a bottom-up manner, before presenting our optimization approach (Section 4).

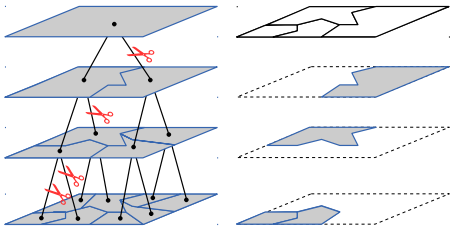


Figure 1: BPT and segmentation by pruning.

3.1 Building a partition tree

Our initialization involves the construction of a BPT, without any introduction of shape information. BPTs are built by using a bottom-up region merging approach [2]. At each iteration, the two most similar regions are merged into a bigger one and a node representing the new region is added to the BPT, connected to its two corresponding children (see Fig. 1). The final tree contains exactly $2n-1$ nodes, where the root node will represent the whole image, the following level the subdivision of the image into two disjoint regions, and so on.

Throughout the BPT construction, a region adjacency graph (RAG) is maintained [27]. The edges in the RAG are labeled with region dissimilarity values and a priority queue is constructed on them. Whenever two regions are merged, the labels of the adjacent edges on the RAG must be updated. The overall complexity is $O(n\log(n)M)$ [16], n being the initial number of nodes and M the maximum degree of a RAG node during the execution. Given that usually $M \ll n$ or even $M < K$ depending on the dissimilarity function, the algorithm is quasilinear in practice.

Region merging requires to provide a *model* to represent regions and a *dissimilarity* function to compare the regions. Non-parametric models, *i.e.* color histograms, are known to be superior to parametric models (*e.g.,* region mean intensities) because they take into account the possible variability of a color into the same class [36]. A large number of dissimilarity functions have been proposed in the literature [4]. The authors of [36] showed the advantages of using a cross-bin measure. For this purpose, we propose to use the mean of Earth Mover's Distances [25] among histograms of every color channel, denoted by $\mathrm{EMD}(R_i, R_j)$.

In the methods that cut BPTs based on classification (*e.g.,* [36]), the dissimilarity measure used to construct the trees is purely based on the comparison of color histograms (optionally, area or growth constraints). Non-parametric models allow to represent and compare inhomogeneous regions, which is useful for textures and light gradients. However, internal class variability (*e.g.,* an object composed by areas of different colors) is not at all considered. In our context, to take class variability into account during the BPT construction, we have included a term that acts as a clustering force for similarly classified regions. We then propose to define the dissimilarity between regions R_i and R_j as follows: $D(i,j) =:$

$$\min(|R_i|,|R_j|)(-\log P(L_i = L_j) + \lambda\,\mathrm{EMD}(R_i, R_j)). \tag{7}$$

The first factor is a typical weight on the area that enforces the balancing of the trees [4]. Parameter λ controls the influence of each term. The first term involves the probability of assigning the same class to the two regions:

$$P(L_i = L_j) = \sum_{L_k \in \mathcal{L}} P(L_k|R_i)P(L_k|R_j), \tag{8}$$

where $P(L_k|R_i)$ can be estimated either by majority voting [32] or by computing the probability of simultaneously assigning label L_k to all pixels in the region, conditioned by the fact that

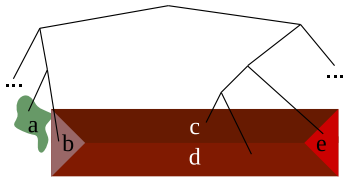


Figure 2: Case where no node represents the object ($bcde$).

every pixel must be labeled equally. In our experiments, the use of (7) has already improved the performance of the standard non-optimized BPTs.

3.2 Best segmentation representable by a given tree

Given that a BPT represents a hierarchy of possible segmentations, we now need to find for any BPT the best possible segmentation that it contains w.r.t. criterion (6).

The traditional processing on BPTs consists in selecting the highest or lowest branches satisfying a given condition [22, 36]. However, some contributions have posed problems in terms of energy-minimization [26, 27]. In particular, [26] has interpreted segmentation as a horizontal *cut* on the tree (see Fig. 1), *i.e.* with a source at every leaf and a sink at the root. Let us denote τ a tree and $C(\tau)$ the energy of the cut on τ that minimizes (6). Our task is to find such a cut.

Considering that the branches in the trees are independent, the globally optimal cut can be found by means of a dynamic programming algorithm [26]. Let us denote $\mathcal{E}(R) = \min_{L \in \mathcal{L}} E(\{R\}, \{L\})$ the lowest possible energy of R. The tree is traversed in a bottom-up manner. Whenever a region R is visited, the following property is evaluated:

$$\mathcal{E}(R) \leqslant C(R_{left}) + C(R_{right}). \tag{9}$$

If the property does not stand, we set $C(R) = C(R_{left}) + C(R_{right})$ and keep the best cuts of both children. Otherwise, we set $C(R) = \mathcal{E}(R)$ and replace the cuts by R with label L. This is performed in linear time on the size of the image, since only one traversal of the BPT is required.

3.3 Issues with unoptimized trees

Even though the globally optimal cut on a BPT can be found efficiently, the organization of the nodes in the tree structure restricts the possible cuts that can be done on them. In Fig. 2 a toy example illustrates this issue. Let us suppose that an aerial shot of a city captures a house with a non-uniform roof. During the construction of the tree, a and b are merged together because they feature the lower dissimilarity among every pair of regions. It is a typical situation that parts of a roof might be more contrasting among themselves than with other objects [1]. We must point out that at the moment a and b were merged, it was impossible to know that b would eventually form a more significant object under a different sequence of merges. The resulting tree does not allow to perform any cut that would include the whole building into the same object, even using strong shape priors, given that it is split through different branches.

4 Optimizing the tree for better segmentation

To optimize the BPTs we follow a local search approach, in which a solution is iteratively modified by performing local transformations on the trees.

4.1 Moves and associated updates

We propose a *move* that performs a *prune-and-paste* of a branch into another part of the tree. The pruning and paste locations must represent spatially adjacent regions. Fig. 3 illustrates such a move: α is the paste place and β is the pruning place. We denote by $\mathrm{LCA}(\alpha, \beta)$ their lowest common ancestor. The move creates a new node $\alpha\beta$ in the paste side that comprises α and β. In the pruned side, the tree is collapsed after β is removed. The number of possible moves is bounded by $O(n \log(n))$ in a balanced tree (*c.f.* appendix for the proof).

We store at each node R the branch cost $C(R)$ of the best possible cut within its branch. When applying a move as depicted in Fig. 3, it is necessary to recompute the branch cost C at the ancestry of α and β only. The rest of the branches are unaffected, as observed in (9). Among the ancestry, only the nodes below $\mathrm{LCA}(\alpha, \beta)$ require to recompute their models (shape and color features), given that further up the regions represented by the nodes do not change. Thus we recompute the features (and thus $\mathcal{E}(R)$ and $C(R)$) only in that part of the tree, and for the nodes in the tree above $\mathrm{LCA}(\alpha, \beta)$ we recompute only their branch cost $C(R)$, which simply involves reassessing (9) without reevaluating $\mathcal{E}(R)$ nor their features.

In a balanced tree there are at most $O(\log(n))$ ancestors of α and β and, as stated before, for some of these ancestors the model of the regions must be updated. If we suppose that the complexity of updating a model (*i.e.* merging two children) is k, then the computation of the new costs C is bounded by $O(k \log(n))$. Usually $k \ll n$, therefore the time is $O(\log(n))$ in practice.

The adjacency relations in a BPT can be derived from the children by observing that the ancestors of two adjacent nodes are also adjacent, until the LCA. At the finest level the adjacency is usually known (*e.g.,* a 4-connected grid).

4.2 Properties of the moves

We will now explore some properties of the *prune-and-paste* move. In particular, we will show that to find all possible good moves we do not need to exhaustively evaluate the energy gain of every possible move.

Proposition 1. *Given a tree τ, suppose a node R_m is pasted at $\tau_i < \tau_1$ leading to a new tree φ. Let us consider an alternative move that pastes R_m at τ_j, with $\tau_i < \tau_j < \tau_1$, producing a tree ψ. In the cases where either $C(\varphi_1) - C(\tau_1) \leqslant 0$ or $C(R_m) \geqslant C(\varphi_1) - C(\tau_1)$, then $C(\psi_1) \geqslant C(\varphi_1)$.*

Intuitively, if we paste very low, the entire branch already contains the new pasted subtree. The proposition states that, if a move reduces the energy in the branch, a higher paste place will not do better. Under certain assumptions, if the move increases the energy, pasting higher will not be favorable either. The proof is developed in the appendix.

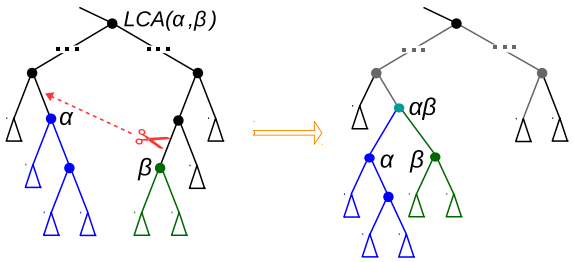


Figure 3: *Prune-and-paste* move.

Proposition 2. *Let us consider a case where Prop. 1 hypotheses do not apply. There might then exist a higher paste place τ_α so that $C(\psi_1) < C(\varphi_1)$. Let us suppose that instead of pasting at τ_α we paste at τ_β, with $\tau_\alpha < \tau_\beta < \tau_1$, leading to a tree ρ. Then $C(\rho_1)$ would monotonously decrease as the paste place τ_β is located higher.*

When the hypothesis of Prop. 1 do not apply, there might exist a favorable paste place higher in the branch. However, we know that the higher it is, the most beneficial it can be (proof in appendix). As a result, we can just consider the highest possible paste place (right below the LCA). However, any paste place between the location of the original cut and the LCA would lead to the same energy (proof in appendix). These cases happen when it is preferable to cut the pruned node apart. The higher we paste R_m, the less we condition the way the rest of the tree must be cut.

Following these properties, an exhaustive search of good moves can be done as follows: for every possible pruning place we check the gain of the moves for only the lowest paste places. In some cases, an additional check with the paste place at some point between the cut and the LCA might be needed. Now that a constant number of paste places is verified, a factor in the number of moves is gained.

The lowest scale both on the pruning and paste side can be set to the pixel level. A coarser scale on the pruning side can be used to adjust the precision of the moves, depending on the application. If a coarser scale is set on the paste side, we limit the minimum object size of the partitions, since we ignore moves that might lower the cut below a certain level. This can be adjusted by observing the density functions of the area feature.

4.3 Optimization approach

We propose the following optimization scheme, which must be iterated:

1) Construct a heap of all moves according to the branch
   cost variation ΔC.
2) a) Either apply the best move, or
   b) apply the best k moves (when still appropriate).

The first step involves exploring the whole search space, featuring an $O(n \log^2(n))$ complexity. Moves are tried to measure the energy gain but are not applied. Propositions (1-2) can be used to reduce the execution time of this step, and the evaluations of energy gain can be done in parallel.

In the second step, good moves are applied. As soon as a move is applied, the tree is restructured and the effect of some other moves might be altered. New good moves might also arise. 2a) just applies the best move and reiterates. As a shortcut, we can just update the entries in the heap of the moves that might have been affected. This option is still costly because ΔC must be recomputed for any move that could have been possibly affected, even though in practice this might be the case for just a few of them. This approach guarantees to apply the best move each time. Considering the fact that there might be many unrelated good moves in the tree, 2b) proposes to apply a number k of best moves, but verifying for each move that ΔC did not increase as a result of the previous transformations done on the tree. This approach will apply a number of independent moves first, ignoring the fact that some new good moves might arise, which will be dealt with in the next iteration. The loop stops when there are no more moves whose associated ΔC is negative.

5 Experiments

The proposed method was tested on satellite images of urban scenes extracted from Google Maps[1]. Figs. 4(a) and 5(a) show two color images acquired over New York City and the area of Brest, with spatial dimensions of 225×180 and 450×850 pixels, respectively. For both images, manual segmentation (Figs. 4(b) and 5(b)) was performed by combining visual inspection with cadastral records available through OpenStreetMap[2]. The list of the considered object classes is given in Fig. 4(b) (no instance of the *internal road* class is present in the Brest image). Note that in the particular case of buildings, cadastral information was used to delineate every object independently even when they are spatially adjacent (see different nuances of red in Fig. 5(b)).

To evaluate the performance of the proposed method, we use two criteria:

1) Overall accuracy $\mathcal{A}$, defined as the proportion of correctly classified pixels.

2) Building's overlap $\mathcal{D}$. For every building in the manually segmented image, we search for the most overlapping *building* region in the segmentation map in terms of Dice's coefficient [8]. The criterion $\mathcal{D}$ is estimated by averaging the computed coefficients.

In the experiments, the parameters of the SVM with a Gaussian radial basis function kernel (penalty and spread of the kernel) were tuned by tenfold cross-validation [38]. BPT construction was performed with 30 bins per histogram and $\lambda = 1$ to make the terms in (7) equally relevant. The optimization was executed on tree branches with a size of at least 10 pixels. The criterion (6) involved area, rectangularity and elongatedness shape descriptors. The distributions were trained on a set of sample objects from an adjacent image. In the area covered by these objects, 100 random pixels per class were selected to train the SVM.

We compared the performance of the proposed approach with the following methods: 1) SVM; 2) graph cut with α-expansion [3] (GC); 3) cut on the BPT, regularized by the number of regions without using shape priors (TC) [26]; 4) cut on the same BPT with our shape formulation (6), but without tree optimization (TSC). Figs. 4(c)-(i) illustrate the output of these techniques for the image of New York, together with the results obtained by applying our optimization method with different values of k (see Sec. 4.3). The SVM classification exhibits many issues, notably the assignment of some roof parts to the wrong class. GC and TC smooth the results though do not correct the main mistakes in the classification. In the initial cut with shape priors on the unoptimized tree (TSC), some regions are enhanced but some others are significantly deteriorated with respect to the previous methods. This is due to the faulty tree construction that does not represent the entire objects in unique nodes. Figs. 4(g)–(h) show that the optimization of the tree copes with these issues, not only enhancing the initial cut on the tree but also outperforming the other techniques.

The evolution of the energy (6), the accuracy $\mathcal{A}$ and the building's overlap $\mathcal{D}$ with respect to the number of iterations is depicted in Figs. 4(j)-(l). As expected, the energy curve becomes less smooth as k goes larger. For values of k small enough, the segmentation maps are comparable, though for larger k the performance starts to degrade. This validates the fact that many branch moves are independent and can be applied prior to reconstructing the heap.

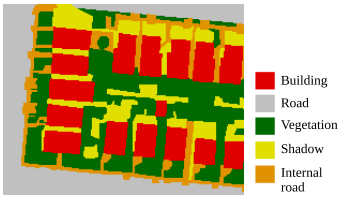
The BPT construction time for this image on an 8-CPU 2.7 GHz processor is 1.25 seconds. The optimization time, summarized in Fig. 4(m), is considerably faster when Props. 1-2 are taken into account.

Fig. 5 illustrates experimental results for the image of Brest. Our method was executed with $k = 600$. Fig. 5(c) shows the obtained segmentation map. Two fragments of the map (boxed in Fig. 5(b)) are zoomed in for comparison with the other methods in Figs. 5(d)-(g). These results

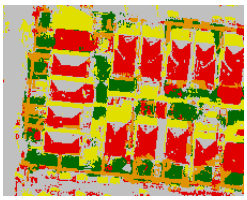
[1] `http://maps.google.com/`

[2] `http://www.openstreetmap.org/`

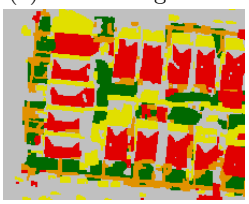
(a) Color image.



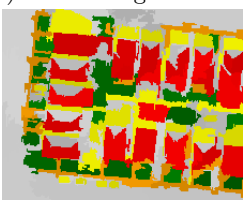
(b) Manual segmentation.



(c) SVM. $\mathcal{A} = 0.64$, $\mathcal{D} = 0.8$.

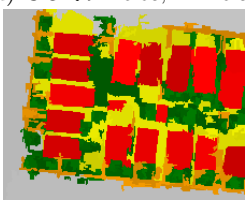


(d) GC. $\mathcal{A} = 0.68$, $\mathcal{D} = 0.81$.

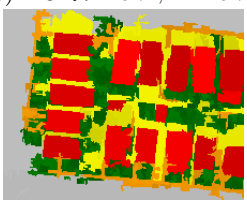


(e) TC. $\mathcal{A} = 0.71$, $\mathcal{D} = 0.71$.

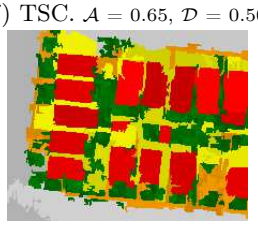
(f) TSC. $\mathcal{A} = 0.65$, $\mathcal{D} = 0.50$.



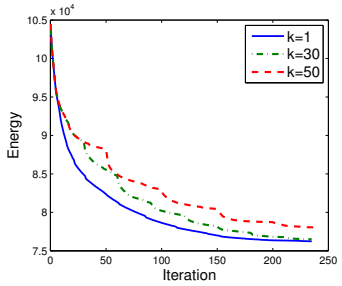
(g) Opt_1. $\mathcal{A} = 0.79$, $\mathcal{D} = 0.89$.



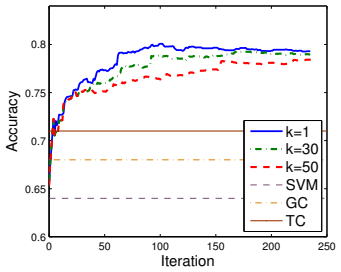
(h) Opt_{30}. $\mathcal{A} = 0.79$, $\mathcal{D} = 0.90$.



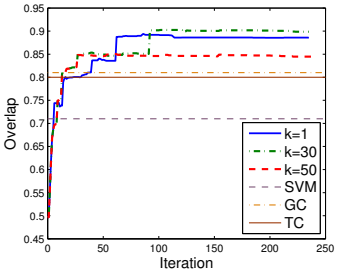
(i) Opt_{50}. $\mathcal{A} = 0.78$, $\mathcal{D} = 0.84$.



(j) Evolution of the energy.



(k) Evolution of overall accuracy.



(l) Evolution building's overlap.

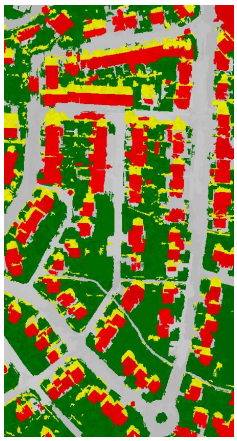
k	Time[s]	Time[s] (Props. 1-2)
1	707	334
30	23.08	9.02
50	12.47	4.69

(m) Execution time.

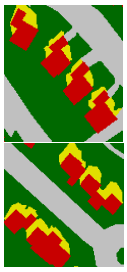
Figure 4: Experimental results for the satellite image over New York City (see text for details).

(a) Color image.

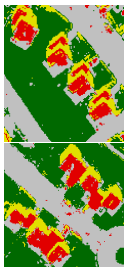
(b) Manual segmentation



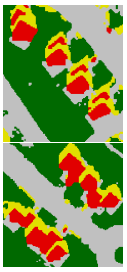
(c) BPT optimization.



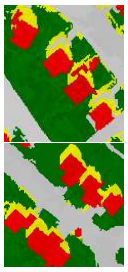
(d) Manual.



(e) SVM.



(f) GC.



(g) BPT Opt.

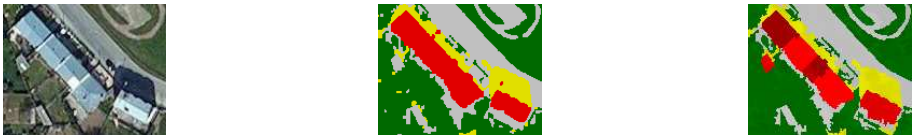
	$\mathcal{A}$	$\mathcal{D}$
SVM	0.764	0.50
GC	0.799	0.49
TC	0.804	0.47
Opt.	0.813	0.69

(h) Performance.

Time [s]	Time [s] (Props. 1-2)
214	84

(i) Execution time.

Figure 5: Experimental results for the satellite image over Brest (see text for details).



(a) Color image. (b) Graph cuts. (c) BPT opt.

Figure 6: BPT optimization detects individual buildings.

validate the previous observations. The BPT construction takes 13 seconds and the optimization time is improved by using Props. 1-2 (see Fig. 5(i)).

We highlight in another satellite image over Brest (Fig. 6) that our method is able to separate an entire building blob into rectangles. In addition, the use of shape features permits to "switch" a small building to the correct class, even though its color was assigned to *road* by the classifier.

6 Conclusion

In this paper, we have presented a novel BPT-based approach for multi-class image segmentation with shape priors. The segmentation is formulated as an *energy optimization* task, using the probability distributions of color and shape features. We construct a BPT of the image and search for a horizontal cut on the tree, which minimizes the proposed energy function. One of the key innovations of the proposed method is that *instead of using the standard static BPTs* (as in [26]), *we optimize them by pruning and regrafting their branches at different scales* in order to minimize the energy of the best segmentation extracted from the tree. This work can be interpreted as a generalization of superpixel techniques, with the addition of shape priors, given that we perform segmentation by combining sets of pixels of possibly different sizes.

Our experimental evaluation showed that the proposed framework efficiently incorporates shape information into multi-class segmentation and outperforms the recently proposed state-of-the-art methods. Furthermore, we conducted a study to: a) *reduce the space of possible moves in BPTs*; b) *design low-complexity shape features*. The theoretical properties proved in Sec. 4.2 yielded a fast implementation of the proposed approach.

In the future, we intend to study possible extensions of the family of branch moves and their associated theoretical guarantees. In addition, we are interested in including interactions between objects and designing dedicated optimizers. We also plan to extend the number of shape features in order to incorporate further priors in the segmentation.

References

[1] Huseyin Gokhan Akcay and Selim Aksoy. Building detection using directional spatial constraints. In *IGARSS*, pages 1932–1935, 2010.

[2] J-M Beaulieu and Morris Goldberg. Hierarchy in picture segmentation: A stepwise optimization approach. *IEEE TPAMI*, 11(2):150–163, 1989.

[3] Yuri Boykov, Olga Veksler, and Ramin Zabih. Fast approximate energy minimization via graph cuts. *TPAMI, IEEE Transactions on*, 23(11):1222–1239, 2001.

[4] Felipe Calderero and Ferran Marques. Region merging techniques using information theory statistical measures. *IEEE TIP*, 19(6):1567–1586, 2010.

[5] Daniel Cremers, Stanley J Osher, and Stefano Soatto. Kernel density estimation and intrinsic alignment for shape priors in level set segmentation. *IJCV*, 69(3):335–351, 2006.

[6] Daniel Cremers, Florian Tischhäuser, Joachim Weickert, and Christoph Schnörr. Diffusion snakes: Introducing statistical shape knowledge into the Mumford-Shah functional. *IJCV*, 50(3):295–313, 2002.

[7] Piali Das, Olga Veksler, Vyacheslav Zavadsky, and Yuri Boykov. Semiautomatic segmentation with compact shape prior. *Image and Vision Computing*, 27(1):206–219, 2009.

[8] L. Dice. Measure of the amount of ecological association between species. *Ecology*, 26(3):297–302, 1945.

[9] Pedro F Felzenszwalb. Representation and detection of deformable shapes. *IEEE TPAMI*, 27(2):208–220, 2005.

[10] Daniel Freedman and Tao Zhang. Interactive graph cut based segmentation with shape priors. In *CVPR*, pages 755–762, 2005.

[11] Gareth Funka-Lea, Yuri Boykov, Charles Florin, M-P Jolly, Romain Moreau-Gobard, Rana Ramaraj, and Daniel Rinck. Automatic heart isolation for ct coronary visualization using graph-cuts. In *IEEE Int. Symp. Biomedical Imaging: Nano to Macro*, pages 614–617, 2006.

[12] Gonzalo Giribet. Efficient tree searches with available algorithms. *Evolutionary bioinformatics online*, 3:341, 2007.

[13] Lena Gorelick, Olga Veksler, Yuri Boykov, and Claudia Nieuwenhuis. Convexity shape prior for segmentation. In *ECCV*, pages 675–690, 2014.

[14] Rui Huang, Vladimir Pavlovic, and Dimitris N Metaxas. A graphical model framework for coupling MRFs and deformable models. In *CVPR*, pages 739–746, 2004.

[15] M Pawan Kumar, PHS Ton, and Andrew Zisserman. Obj cut. In *CVPR*, pages 18–25, 2005.

[16] T Kurita. An efficient agglomerative clustering algorithm for region growing. In *IAPR Workshop on Machine Vision Applications*, pages 210–213, 1994.

[17] Camille Kurtz, Nicolas Passat, Pierre Gancarski, and Anne Puissant. Extraction of complex patterns from multiresolution remote sensing images: A hierarchical top-down methodology. *Pattern Recognition*, 45(2):685–706, 2012.

[18] Victor Lempitsky, Andrew Blake, and Carsten Rother. Image segmentation by branch-and-mincut. In *ECCV*, pages 15–29, 2008.

[19] Michael E Leventon, W Eric L Grimson, and Olivier Faugeras. Statistical shape influence in geodesic active contours. In *CVPR*, pages 316–323, 2000.

[20] Fulu Li and Andrew Lippman. Random tree optimization for the construction of the most parsimonious phylogenetic trees. In *CISS*, pages 757–762, 2009.

[21] Wenwen Li, Michael F Goodchild, and Richard Church. An efficient measure of compactness for two-dimensional shapes and its application in regionalization problems. *Int. Journal of Geographical Information Science*, 27(6):1227–1250, 2013.

[22] Huihai Lu, John C Woods, and Mohammed Ghanbari. Binary partition tree for semantic object extraction and image segmentation. *IEEE Trans. Circuits and Systems for Video Technology*, 17(3):378–383, 2007.

[23] Raul S Montero and Ernesto Bribiesca. State of the art of compactness and circularity measures. *Int. Mathematical Forum*, 4(27):1305–1335, 2009.

[24] Claudia Nieuwenhuis, Eno Töppe, and Daniel Cremers. A survey and comparison of discrete and continuous multi-label optimization approaches for the Potts model. *IJCV*, 104(3):223–240, 2013.

[25] Yossi Rubner, Carlo Tomasi, and Leonidas J Guibas. A metric for distributions with applications to image databases. In *ICCV*, pages 59–66, 1998.

[26] Philippe Salembier, Samuel Foucher, and Carlos López-Martínez. Low-level processing of PolSAR images with binary partition trees. In *IGARSS*, 2014.

[27] Philippe Salembier and Luis Garrido. Binary partition tree as an efficient representation for image processing, segmentation, and information retrieval. *IEEE TIP*, 9(4):561–576, 2000.

[28] Thomas Schoenemann and Daniel Cremers. Globally optimal image segmentation with an elastic shape prior. In *ICCV*, pages 1–6, 2007.

[29] Bernard W Silverman. *Density estimation for statistics and data analysis*. CRC press, 1986.

[30] Ali Kemal Sinop and Leo Grady. Uninitialized, globally optimal, graph-based rectilinear shape segmentation-the opposing metrics method. In *ICCV*, pages 1–8, 2007.

[31] Greg Slabaugh and Gozde Unal. Graph cuts segmentation using an elliptical shape prior. In *ICIP 2005.*, pages II–1222. IEEE, 2005.

[32] Y. Tarabalka, J. Chanussot, and J. A. Benediktsson. Segmentation and classification of hyperspectral images using watershed transformation. *Pattern Recognition*, 43(7):2367–2379, 2010.

[33] Yuliya Tarabalka and James C Tilton. Improved hierarchical optimization-based classification of hyperspectral images using shape analysis. In *IGARSS*, pages 1409–1412, 2012.

[34] James Tilton and Edoardo Pasolli. Incorporating edge information into best merge region-growing segmentation. In *IGARSS*, 2014.

[35] Godfried T Toussaint. The rotating calipers: An efficient, multipurpose, computational tool. In *ICCTIM*, pages 215–225, 2014.

[36] Silvia Valero, Philippe Salembier, and Jocelyn Chanussot. Hyperspectral image representation and processing with binary partition trees. *IEEE TIP*, 22(4):1430–1443, 2013.

[37] Silvia Valero, Philippe Salembier, and Jocelyn Chanussot. Object recognition in urban hyperspectral images using binary partition tree representation. In *IGARSS*, pages 4098–4101, 2013.

[38] Vlamimir Vapnik. *Statistical learning theory*. Wiley New York, 1998.

[39] Olga Veksler. Star shape prior for graph-cut image segmentation. In *ECCV*, pages 454–467, 2008.

[40] Veronica Vilaplana, Ferran Marques, and Philippe Salembier. Binary partition trees for object detection. *IEEE TIP*, 17(11):2201–2216, 2008.

[41] Ting-Fan Wu, Chih-Jen Lin, and Ruby C Weng. Probability estimates for multi-class classification by pairwise coupling. *JMLR*, 5:975–1005, 2004.

[42] Mingqiang Yang, Kidiyo Kpalma, Joseph Ronsin, et al. A survey of shape feature extraction techniques. *Pattern recognition*, pages 43–90, 2008.

A Proofs

Proposition 1. *Given a tree τ, suppose a node R_m is pasted at $\tau_i < \tau_1$ leading to a new tree φ. Let us consider an alternative move that pastes R_m at τ_j, with $\tau_i < \tau_j < \tau_1$, producing a tree ψ. In the cases where either $C(\varphi_1) - C(\tau_1) \leqslant 0$ or $C(R_m) \geqslant C(\varphi_1) - C(\tau_1)$, then $C(\psi_1) \geqslant C(\varphi_1)$.*

Proof. Let us abbreviate $\mathcal{E}(\tau_i)$ as e_i^τ and $C(\tau_i)$ as c_i^τ. Following (9), $c_1^\tau = min(e_1^\tau, \overline{c_2^\tau} + min(e_2^\tau, \overline{c_3^\tau} + min(...min(e_{i-2}^\tau, \overline{c_{i-1}^\tau} + min(e_{i-1}^\tau, \overline{c_i^\tau} + c_i^\tau))...))))$, where $\overline{c_i^\tau}$ denotes the sibling of c_i^τ (see Fig. **??**). This implies: $\underbrace{\underset{k=1}{\overset{i-1}{\forall}} \left(e_k^\tau \geqslant c_1^\tau - \Sigma_{j=2}^k \overline{c_j^\tau}\right)}_{\alpha}$ and $\left(c_i^\tau \geqslant c_1^\tau - \Sigma_{j=2}^i \overline{c_j^\tau}\right)$.

Let us now suppose that a node R_m is pasted at τ_i. Then $c_1^\varphi = min(e_1^\varphi, \overline{c_2^\tau} + min(e_2^\varphi, \overline{c_3^\tau} + min(...min(e_{i-2}^\varphi, \overline{c_{i-1}^\tau} + min(e_{i-1}^\varphi, \overline{c_i^\tau} + min(e_x^\varphi, c_i^\tau + c_m^R)))...))))$, which implies:

$$\underbrace{\underset{k=1}{\overset{i-1}{\forall}} \left(e_k^\varphi \geqslant c_1^\varphi - \Sigma_{j=2}^k \overline{c_j^\tau}\right)}_{\beta} \text{ and } \left(e_x^\varphi \geqslant c_1^\varphi - \Sigma_{j=2}^i \overline{c_j^\tau}\right) \text{ and } \underbrace{\left(c_m^R \geqslant c_1^\varphi - \Sigma_{j=2}^i \overline{c_j^\tau} - c_i^\tau\right)}_{\gamma}.$$

Let us now paste R_m one position upper than before. We wish to check if it is possible that this move will be better that the previous one ($c_1^\psi < c_1^\varphi$): $c_1^\psi = min(e_1^\varphi, \overline{c_2^\tau} + min(e_2^\varphi, \overline{c_3^\tau} + min(...min(e_{i-2}^\varphi, \overline{c_{i-1}^\tau} + min(e_y^\psi, c_m^R + min(e_{i-1}^\tau, \overline{c_i^\tau} + c_i^\tau)))...)) < c_1^\varphi$. This can be true if and only if:

$$\underbrace{\underset{k=1}{\overset{i-2}{\exists}} \left(e_k^\varphi < c_1^\varphi - \Sigma_{j=2}^k \overline{c_j^\tau}\right)}_{I} \text{ or } \underbrace{\left(e_y^\psi < c_1^\varphi - \Sigma_{j=2}^{i-1} \overline{c_j^\tau}\right)}_{II} \text{ or}$$

$$\underbrace{\left(e_{i-1}^\tau < c_1^\varphi - \Sigma_{j=2}^{i-1} \overline{c_j^\tau} - c_m^R\right)}_{III} \text{ or } \underbrace{\left(c_m^\tau < c_1^\varphi - \Sigma_{j=2}^i \overline{c_j^\tau} - c_i^\tau\right)}_{IV}.$$

In this expression, I contradicts β. Considering that $e_y^\psi = e_{i-1}^\varphi$ (see Fig. **??**), the term II also contradicts β. The term IV contradicts γ. We must now analyze III. By combining III and α: $c_1^\tau - \Sigma_{j=2}^{i-1} \overline{c_j^\tau} \leqslant e_{i-1}^\tau < c_1^\varphi - \Sigma_{j=2}^{i-1} - c_m^R \Rightarrow c_m^R < c_1^\varphi - c_1^\tau$.

If $c_1^\varphi - c_1^\tau \leqslant 0$, then c_m^R must be non-positive, which contradicts our hypothesis. If $c_1^\varphi - c_1^\tau > 0$: then it must be $c_m^R < c_1^\varphi - c_1^\tau$. As a conclusion, if the first move decreases C, III is contradicted, hence it must be $c_1^\psi \geqslant c_1^\varphi$. For a positive gain, III is contradicted unless $c_m^R < c_1^\varphi - c_1^\tau$.

□

Proposition 2. *Let us consider a case where Prop. 1 hypotheses do not apply. There might then exist a higher paste place τ_α so that $C(\psi_1) < C(\varphi_1)$. Let us suppose that instead of pasting at τ_α we paste at τ_β, with $\tau_\alpha < \tau_\beta < \tau_1$, leading to a tree ρ. Then $C(\rho_1)$ would monotonously decrease as the paste place τ_β is located higher.*

Proof. If Prop. 1 hypotheses do not apply, then the term III in its proof must be true. This term implies that when pasting at τ_j, the cut on the tree will be located at or below R_m. The cost c_1^ψ associated with this move will then be $c_1^\psi = \Sigma_{j=2}^i \overline{c_j^\tau} + c_i^\tau + c_m^R$, considering the location of the new cut. Analogously, the cost when pasting R_m k units up of i is: $c_1^\rho = \Sigma_{j=2}^{i-k} \overline{c_j^\tau} + c_{i-k}^\tau + c_m^R$.

Notice in the previous expression that we consider that the cut is still as low as R_m. The cut could not be higher, because if it were the case, then it would have already been cut there before.

Let us now see if the cost c_1^ρ could increase as k advances:

$\Sigma_{j=2}^{i-k}\overline{c_j^\tau} + c_{i-k}^\tau + c_m^R - (\Sigma_{j=2}^{i-k+1}\overline{c_j^\tau} + c_{i-k+1}^\tau + c_m^R)$
$= -\overline{c_{i-k+1}^\tau} + c_{i-k}^\tau - c_{i-k+1}^\tau > 0$
$\Leftrightarrow c_{i-k}^\tau > c_{i-k+1}^\tau + \overline{c_{i-k+1}^\tau}$, contradicting the algorithm to compute the cuts, hence the cost at R_1 must monotonously decrease.

□

Proposition 3. *Let us suppose we paste R_m at or over the initial cut of tree τ, leading to tree φ. Let us consider we paste higher instead, producing tree ψ. It must then be $c_1^\psi \geqslant c_1^\varphi$.*

Proof. Let us resume the proof of Proposition 2 in the paper. It was shown that $c_1^\psi < c_1^\varphi$ if and only if III was false. At that point we could not contradict III but show that under certain conditions it would be contradicted. Now we will show that the fact that we know the first cut was at or below τ_i will contradict III.

After III and γ we have:

$$\begin{aligned} & e_{i-1}^\tau + c_1^\varphi - \Sigma_{j=2}^{i}\overline{c_j^\tau} - c_i^\tau \leqslant e_{i-1}^\tau + c_m^R < c_1^\varphi - \Sigma_{j=2}^{i-1}\overline{c_j^\tau} \\ \Leftrightarrow & e_{i-1}\tau - \overline{c_i^\tau} - c_i^\tau \leqslant e_{i-1}^\tau + c_m^R < 0. \end{aligned} \tag{10}$$

If we now add the knowledge about the cut being below τ_{i-1}, it must be $e_{i-1}^\tau > c_i^\tau + \overline{c_i^\tau}$, then

$$c_i^\tau + \overline{c_i^\tau} - \overline{c_i^\tau} - c_i^\tau < e_{i-1}^\tau - \overline{c_i}^\tau - c_i^\tau \leqslant e_{i-1}^\tau + c_m^R < 0 \Leftrightarrow 0 < 0. \tag{11}$$

Therefore, III cannot be true, which proves the proposition.

□

Corollary: Combining Proposition 2 in the paper, where C monotonously decreases ($\leqslant$) as the paste location gets higher, and Proposition 3, where C cannot decrease ($\geqslant$), it becomes evident that pasting a node anywhere between the initial cut and the lowest common ancestor produces the same effect on the energy.

Proposition 4. *The storage space required to add the convex hull to every node of a balanced BPT in a discrete environment is bounded by $O(n\log(n))$.*

Proof. Let us call $CH(R)$ the convex hull of a region R. In the extreme case (the most compact region), $CH(R)$ can be as large as the perimeter δR of R which, in a discrete implementation (assuming 4-connectivity) does not contain more points than four times the area of the region:

$$CH(R) \leqslant \delta R \leqslant 4|R|. \tag{12}$$

As a consequence, the points of the convex hull of *all* regions in a tree $\mathcal{T}$ must be

$$\sum_{R_i \in \mathcal{T}} |CH(R_i)| \leqslant 4 \sum_{R_i \in \mathcal{T}} |R_i|. \tag{13}$$

If we observe that in a balanced tree

$$\sum_{R_i \in \mathcal{T}} |R_i| = \sum_{l=1}^{\#levels} \sum_{R_j \in l \in \mathcal{T}} |R_j| = \sum_{l=1}^{\#levels} n = n \sum_{l=1}^{\#levels} 1 = n \cdot \#levels = n\log(n), \tag{14}$$

then (13) is bounded by a factor of $n\log(n)$.

□

Proposition 5. *The complexity of computing the convex hull of every region represented in a balanced BPT in a discrete environment, is bounded by* $O(n\log(n))$.

Proof. Let us call $CH(R)$ the convex hull of a region R. In the extreme case (the most compact region), $CH(R)$ can be as large as the perimeter δR of R which, in a discrete implementation (assuming 4-connectivity) does not contain more points than four times the area of the region:

$$CH(R) \leqslant \delta R \leqslant 4|R|. \tag{15}$$

The time to compute $CH(R_i)$ is linear on the number of points in the polygons of the children (see [35] in the article):

$$O(|\delta LeftChild(R_i)| + |\delta RightChild(R_i)|). \tag{16}$$

The time to compute the convex hull of every node in the tree is then bounded by a factor of:

$$\begin{aligned} &\sum_{R_i \in \mathcal{T}} (|\delta LeftChild(R_i)| + |\delta RightChild(R_i)|) \\ \leqslant 4 &\sum_{R_i \in \mathcal{T}} (|LeftChild(R_i)| + |RightChild(R_i)|) = 4 \sum_{R_i \in \mathcal{T}} |R_i|. \end{aligned} \tag{17}$$

Following (14), the execution time is then a factor of $n\log(n)$. □

Proposition 6. *The amount of spatially adjacent regions in a balanced BPT is bounded by* $O(n\log(n))$.

Proof. Let us call $\mathcal{N}_R$ the number of neighbors of the region R. At the lowest scale and in a discrete environment we can suppose that the number of neighbors is equal to its boundary length (δR). We are interested in knowing the number of neighbors at all scales. In a balanced tree it can be assumed that the number of neighbors at every scale is half the number at the following one. As a result:

$$\mathcal{N}_R = \delta R + \frac{1}{2}\delta R + \frac{1}{2^2}\delta R + \frac{1}{2^3}\delta R + ... < 2\delta R. \tag{18}$$

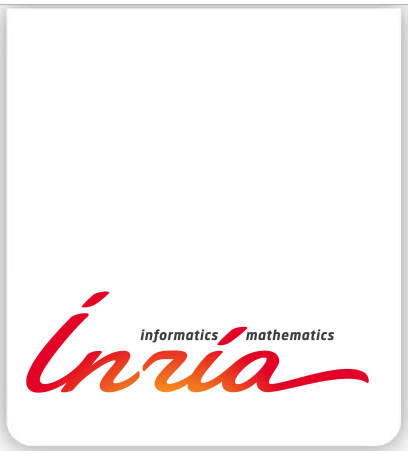
In a discrete implementation (assuming 4-connectivity):

$$\mathcal{N}_R < 2\delta R \leqslant 2 \cdot 4|R| = 8|R|. \tag{19}$$

The summation of the neighbors of *all* regions in a tree $\mathcal{T}$ is then

$$\sum_{R_i \in \mathcal{T}} \mathcal{N}_{R_i} < 8 \sum_{R_i \in \mathcal{T}} |R_i|. \tag{20}$$

Following (14), the total number of neighbors (the possible cut/paste moves) is a factor of $n\log(n)$. □



RESEARCH CENTRE
SOPHIA ANTIPOLIS – MÉDITERRANÉE

2004 route des Lucioles - BP 93
06902 Sophia Antipolis Cedex

Publisher
Inria
Domaine de Voluceau - Rocquencourt
BP 105 - 78153 Le Chesnay Cedex
inria.fr

ISSN 0249-6399