Reduced graphs for min-cut/max-flow approaches in image segmentation

Nicolas Lermé,Lucas Létocart,François Malgouyres [1,2,3]

LAGA UMR CNRS 7539 and LIPN UMR CNRS 7030
Université Paris 13 –Avenue J.B. Clément
93430 Villetaneuse - France

Abstract

In few years, min-cut/max-flow approach has become a leading method for solving a wide range of problems in computer vision. However, min-cut/max-flow approaches involve the construction of huge graphs which sometimes do not fit in memory. Currently, most of the max-flow algorithms are impracticable to solve such large scale problems. In this paper, we introduce a new strategy for reducing exactly graphs in the image segmentation context. During the creation of the graph, we test if the node is really useful to the max-flow computation. Numerical experiments validate the relevance of this technique to segment large scale images.

Keywords: image segmentation, min-cut/max-flow, graph reduction.

1 Introduction

Graph cuts provide a global optimization method based on max-flow/min-cut for solving a wide range of problems encountered in computer vision. Since

[1] Email: `nicolas.lerme@lipn.univ-paris13.fr`
[2] Email: `lucas.letocart@lipn.univ-paris13.fr`
[3] Email: `malgouy@math.univ-paris13.fr`

pioneer work of Greig *et al.* [5], the graph cuts have recently known a quick development with the arrival of a fast max-flow algorithm [2].

At the same time, the resolution of images acquired by digital devices increase constantly. In biomedical imaging, high-resolution data can involve massive graphs containing millions of nodes, which do not fit in memory. For these instances, global optimization methods such as graph cuts are impractical due to memory requirements, even using implicit graph representation.

To overcome this problem, a parallelized max-flow algorithm yielding near-linear speedup with the number of processors has been proposed [4]. This algorithm is able to segment large volumes while keeping optimality on solutions but remains less effective than standard graph cuts on small graphs. On the other side, some authors have also proposed heuristics based on multi-resolution schemes [8,9]. These algorithms reduce drastically speed and memory usage but fail to recover thin structures in images. Other heuristics [7,3] use adjacency graphs. Results highly depend both on the image structure and the low-level segmentation algorithm.

In the present work, we propose an algorithm for reducing exactly graphs. In section 2, we review the graph cuts framework. Next, our approach is detailed in section 3 and compared to standard graph cuts in section 4.

2 Relashionship between images and min-cut/max-flow problem

An image can be defined by a pair $(\mathcal{P}, I)$ consisting of a finite discrete set $\mathcal{P} \subset \mathbb{Z}^d$ $(d > 0)$ of points (pixels in $\mathbb{Z}^2$, voxels in $\mathbb{Z}^3$, etc.) and a function I that maps each point $p \in \mathcal{P}$ to a value $I(p)$ in some value space. Usually, $\mathcal{P}$ corresponds to a rectangle. For an image, we can construct the associated directed weighted graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, c)$ consisting of a set of nodes $\mathcal{V} = \mathcal{P} \cup \{s, t\}$, a set of edges $\mathcal{E}$ and a positive weighting function $c : \mathcal{V}^2 \to \mathbb{R}^+$ defining the edge capacity.

We distinguish two special nodes of $\mathcal{V}$: the source node s specifying the "object" terminal and the sink node t specifying the "background" terminal. Furthermore, we split the set of edges $\mathcal{E}$ in two disjoint sets $\mathcal{E}_n$ and $\mathcal{E}_t$ denoting respectively n-links (neighborhood links) and t-links (terminal links). Next, we associate a neighborhood $\mathcal{N}(p)$ to any point $p \in \mathcal{P}$. In this setting, we will use the following neighborhoods:

$$\begin{aligned} \mathcal{N}_0(p) &= \{q : \textstyle\sum_{i=1}^{d} |q_i - p_i| = 1\} \qquad \forall p \in \mathcal{P}, \\ \mathcal{N}_1(p) &= \{q : |q_i - p_i| \leq 1 \ \forall 1 \leq i \leq d\} \ \forall p \in \mathcal{P}, \end{aligned}$$

where p_i denote the i^{th} coordinate of the point p. For instance, each pixel has 4 and 8 neighbors in 2D, 6 and 26 neighbors in 3D and finally 8 and 80 neighbors in 4D [4]. In the sequel, the terms "connectivity 0" and "connectivity 1" will correspond respectively to the use of a $\mathcal{N}_0$ and $\mathcal{N}_1$ neighborhood.

In [1], Boykov and Jolly showed that the image segmentation problem can be efficiently solved by minimizing a Markov Random Field of the form:

$$E(u) = \beta \cdot \sum_{p \in \mathcal{P}} E_p(u_p) + \sum_{\substack{p,q \in \mathcal{P} \\ q \in \mathcal{N}(p)}} E_{p,q}(u_p, u_q), \tag{1}$$

where $u \in \{0,1\}^{\mathcal{P}}$. As usual, the data fidelity term $E_p(.)$ forces u_p to fit the input data while the smoothness term $E_{p,q}(.)$ penalize neighboring pixels p and q if they have different labels. According to [6], the minimizer of the energy (1) corresponds to a min-cut in a graph and can be efficiently computed by the algorithm described in [2] [5].

3 Reducing graphs

As we have seen before, the memory usage for segmenting high-resolution data by graph cuts can be prohibitive. Nevertheless, we can observe that most of the nodes are useless because they are not traversed by any flow. Clearly, only a small part of nodes is used during the max-flow computation. When reducing such a graph, one would like to extract the smallest possible graph $\mathcal{G}' = (\mathcal{V}', \mathcal{E}', c)$ from $\mathcal{G}$ while keeping a solution u' identical or very close to u. Ideally, we want to maximize the reduction rate $\rho = 1 - \frac{|\mathcal{V}'|}{|\mathcal{V}|}$ s.t. $u \simeq u'$. However, the method for determining $\mathcal{G}'$ also needs to be fast and this rules out the resolution of such an optimization problem. Before describing our method for building $\mathcal{G}'$, let us introduce some terminology. In accordance with the graph construction given in [6], we consider (without loss of generality) that a node is linked to at most one terminal, i.e:

$$(s,p) \in \mathcal{E}_t \Rightarrow (p,t) \notin \mathcal{E}_t, \qquad \forall p \in \mathcal{P}.$$

Also, we summarize the capacities of the t-links at any node $p \in \mathcal{P}$ by $c(p) = c(s,p) - c(p,t)$. For any $B \subset \mathbb{Z}^d$ and $p \in \mathcal{P}$, we denote by $\widetilde{B}_p$ the set translation

[4] Typically, larger neighborhood systems yield better results but increase running times and memory consumptions.

[5] An implementation of the max-flow algorithm is freely available at http://www.cs.cornell.edu/People/vnk/software.html

of B by the point p: $\widetilde{B}_p = \{b + p \mid b \in B\}$. Moreover, for $Z \subset \mathcal{P}$ and $B \subset \mathbb{Z}^d$, we define the dilation of Z by B as:

$$\widetilde{Z}_B = \{z + b \mid b \in B, z \in Z\} = \bigcup_{z \in Z} \widetilde{B}_z .$$

We also define, for any $Z \subset \mathcal{P}$, the maximal amount of flow coming in and out through the n-links by

$$P_{in}(Z) = \textstyle\sum_{\substack{p \notin Z, q \in Z \\ q \in \mathcal{N}(p)}} c(p,q), \quad P_{out}(Z) = \textstyle\sum_{\substack{p \in Z, q \notin Z \\ q \in \mathcal{N}(p)}} c(p,q).$$

Finally, we define the maximum amount of flow passing through the t-links and the flow orientation by

$$A(Z) = \sum_{p \in Z} |c(p)|, \qquad O(Z) = \sum_{p \in Z} \mathrm{sign}(c(p)),$$

where sign$(t) = 1$ if $t > 0$, 0 if $t = 0$ and -1 otherwise.

Let $B \subset \mathbb{Z}^d$, in order to build $\mathcal{G}'$, we remove from the nodes of $\mathcal{G}$ any $Z \subset \mathcal{P}$ such that either

$$\begin{aligned} O(\widetilde{Z}_B) = +|\widetilde{Z}_B| \text{ and } A(\widetilde{Z}_B \setminus Z) \geq P_{out}(\widetilde{Z}_B), \text{ or} \\ O(\widetilde{Z}_B) = -|\widetilde{Z}_B| \text{ and } A(\widetilde{Z}_B \setminus Z) \geq P_{in}(\widetilde{Z}_B). \end{aligned} \tag{2}$$

As an illustration of those conditions, notice for instance that the last condition implies that all the flow that might come in the region $\widetilde{Z}_B$ comes from its boundary and can be absorbed by the band $\widetilde{Z}_B \setminus Z$. Building such sets Z is done by testing each individual pixels of Z. In order to do so, we establish (in a forthcoming paper) that the conjunction of conditions (2) for every $z \in Z$ implies (2) for Z. Considering B, a square window of size $(2r+1)$ $(r > 0)$ centered at the origin, a more conservative test for $z \in Z$ is

$$\begin{cases} c(q) \geq +\delta & \forall q \in \widetilde{B}_z \qquad \text{or} \\ c(q) \leq -\delta & \forall q \in \widetilde{B}_z, \end{cases} \tag{3}$$

where $\delta = \frac{P(B)}{(2r+1)^2 - 1}$, with

$$\begin{aligned} P(B) = \max(&|\{(p,q), p \in Z, q \notin Z \text{ and } p \in \mathcal{N}(q)\}|, \\ &|\{(p,q), p \in Z, q \notin Z \text{ and } q \in \mathcal{N}(p)\}|). \end{aligned}$$

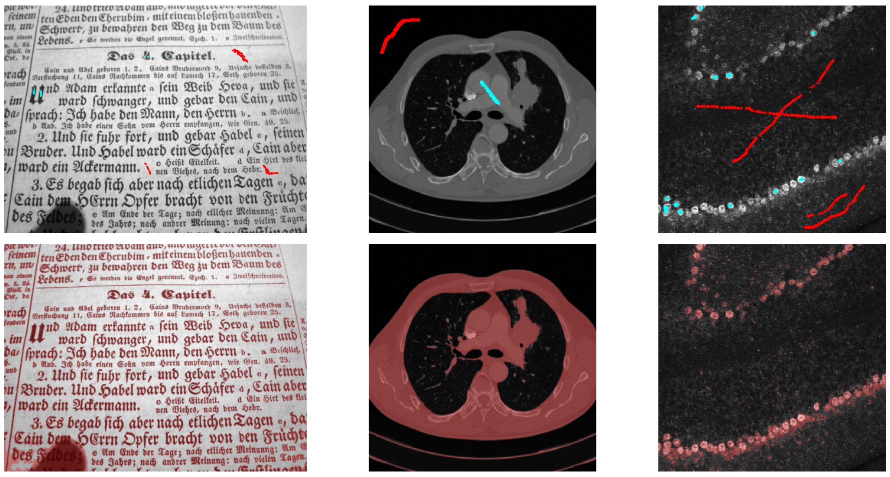
If all the capacities of the n-links are smaller than 1 (which is true for most interesting energies) and (3) holds, the inegality (2) holds for $Z = \{z\}$. Then, $\mathcal{G}'$

is determined by the set of nodes $\mathcal{V}' = \{p \in \mathcal{P}$ not satisfying (3)$\} \cup \{s, t\}$. We have theoretical and empirical evidence suggesting that this graph reduction provides an exact solution. Morever, the condition (3) is simple and a straightforward implementation has a worst-case complexity of $O(|B|)$. Decomposing this test along the d dimensions yields an algorithm with complexity $O(1)$, except for image borders.

4 Experimental results

This section compares the performance of standard graph cuts and our method in terms of speed and memory with the Boykov/Jolly [1] energy model. Experiments are performed on an Athlon Dual Core 6000+ 3GHz with 2GB RAM for segmenting 2D/ 3D images in connectivity 1. Times are averaged over 10 runs.

Introduced in [1], this model has quickly become a standard in applications. From a user viewpoint, it consists of marking some parts of the image as "object" and "background". For more information, we refer the reader to [1].



Text – 3072 × 2304 CT thorax – 358 × 358 × 221 Cells – 358 × 358 × 88

	Standard graph cuts		Our method	
Image	Time	Memory	Time	Memory
Text	7.59	1.23 Gb	4.69	154 Mb
CT thorax	/	14.35 Gb	45.67	1.52 Gb
Cells	/	5.69 Gb	21.29	1.48 Gb

These results compare time and memory usage between standard graph cuts and our method for segmenting 3 real images. The second image represents an abdominal CT with a pulmonary tumor while the third image shows brain cells.

In these experiments, the model's parameters are optimized for better visualization. The window radius is chosen such that memory usage is minimized. For all images, the amount of allocated memory for the graph is reduced by a factor ranging from 3.8x to 9.4x. For the first image, our algorithm is 1.6x faster and require 8x less memory while getting exactly the same result. Moreover, altough the graphs induced by the volumes "ct-thorax" and "cells" do not fit in memory (estimated values) when no reduction is performed, we observe that our algorithm is able to segment them in less than 1 minute. We can notice that we obtain the same conclusions for all the images tested (more than 100).

References

[1] Boykov, Y. and Jolly, M-P., *Interactive Graph Cuts for Optimal Boundary and Region Segmentation of Objects in N-D Images*, ICCV **1** (2001), 105–112.

[2] Boykov, Y. and Kolmogorov, V., *An Experimental Comparison of Min-cut/Max-flow Algorithms for Energy Minimization in Vision*, IEEE Transactions on PAMI **26 (9)** (2004), 1124–1137.

[3] Cigla, C. and Alatan, A.A., *Region-based image segmentation via graph cuts*, ICIP (2008), 2272–2275.

[4] Delong, A. and Boykov, Y., *A Scalable Graph-Cut Algorithm for N-D Grids*, CVPR (2008), 1–8.

[5] Greig, D. M. and Porteous, B. T. and Seheult, A. H., *Exact Maximum A Posteriori Estimation for Binary Images*, Journal of the Royal Statistical Society **51 (2)** (1989), 271–279.

[6] Kolmogorov, V. and Zabih, R., *What Energy Functions Can Be Minimized Via Graph Cuts?*, IEEE Transactions on PAMI **26 (2)** (2004), 147–159.

[7] Li, Yin. and Sun, Jian. and Tang, Chi-Keung. and Shum, Heung-Yeung., *Lazy Snapping*, ACM Transactions on Graphics **23 (3)** (2004), 303–308.

[8] Lombaert, H. and Sun, Y.Y. and Grady, L. and Xu, C.Y., *A Multilevel Banded Graph Cuts Method for Fast Image Segmentation*, ICCV **1** (2005), 259–265.

[9] Sinop, A.K. and Grady, L., *Accurate Banded Graph Cut Segmentation of Thin Structures Using Laplacian Pyramids*, MICCAI **9 (2)** (2006), 896–903.