Top-Down Evolutionary Image Segmentation using a Hierarchical Social Metaheuristic

Abraham Duarte[1], Ángel Sánchez[1], Felipe Fernández[2], Antonio S. Montemayor[1] and Juan J. Pantrigo[1]

[1] ESCET-URJC, Campus de Móstoles, 28933, Madrid Spain
{a.duarte, an.sanchez, a.sanz, j.j.pantrigo}@escet.urjc.es
[2] Dept. Tecnología Fotónica, FI-UPM, Campus de Montegancedo, 28860, Madrid, Spain
Felipe.Fernandez@es.bosch.com

Abstract. This paper presents an application of a hierarchical social (HS) metaheuristic to region-based segmentation. The original image is modelled as a simplified image graph, which is successively partitioned into two regions, corresponding to the most significant components of the actual image, until a termination condition is met. The graph-partitioning task is solved as a variant of the min-cut problem (normalized cut) using an HS metaheuristic. The computational efficiency of the proposed algorithm for the normalized cut computation improves the performance of a standard genetic algorithm. We applied the HS approach to brightness segmentation on various synthetic and real images, with stimulating trade-off results between execution time and segmentation quality.

1 Introduction

In general, image segmentation is one of the most difficult tasks in image analysis. The problem consists in subdividing an image into its constituent regions or objects [7]. The level of subdivision depends on the specific problem being solved. The segmentation result is the labelling of the pixels in the image with a small number of labels. This partition is accomplished in such a way that the pixels belonging to homogeneous regions with respect to one or more features (i.e. brightness, texture or colour) share the same label, and regions of pixels with significantly different features have different labels. According to Ho et al [8], four objectives must be considered for developing an efficient generalized segmentation algorithm: continuous closed contours, non-oversegmentation, independence of threshold setting and short computation time.

Many segmentation approaches have been proposed in the literature [7][12][15]. Roughly speaking, they can be classified as edge-based, thresholding-based and region-based methods. Our proposed method can be considered as region-based and pursuits a high-level extraction of the image structures. As a result, the method produces a hierarchical top-down region-based decomposition of the scene. A way to solve the segmentation problem is as a pixel classification task, where each pixel is assigned to a class or region by considering only local information [7]. We take into account this pixel classification approach by representing the image as a weighted graph where nodes are the pixels in the original image and the arcs together with their associated weights are defined using as local information the distance among pixels and their corresponding brightness values. An optimal bipartition that minimizes the normalized cut value for the image graph is computed. This process is successively repeated for each of the two resulting regions (image subgraphs) after bipartitioning using a binary splitting schema. The application of a hierarchical social metaheuristic to efficiently solve the normalized cut problem is the core of the proposed method.

Many optimization problems are too difficult to be solved exactly in a reasonable amount of time. Due to the complexity of these problems, efficient approximate solutions may be preferable in practical applications. Heuristic algorithms are proposed in this direction. Examples of heuristics are many local search procedures that are problem specific and do not guarantee the optimality. Metaheuristics are high-level general strategies for designing heuristics procedures [6][11][17]. The relevance of metaheuristics is reflected in their application for acceptably solving many different real-world complex problems, mainly combinatorial. Since the initial proposal of Glover about tabu search in 1986, many metaheuristics have emerged to design good general heuristics methods for solving different domain application problems. Genetic programming, GRASP, simulated annealing or ant colony optimization are other well-known examples of metaheuristics. Their relevance is reflected in the publication of many books and papers on this research area [6].

The applications of evolutionary techniques to Image Processing and Computer Vision problems have increased mainly due to the robustness of the approach [12]. Many image analysis tasks like image enhancement, feature selection and image segmentation have been effectively solved using genetic programming [13]. Among these tasks, segmentation is in general the most difficult one. Usually the standard linear segmentation methods are insufficient for a reliable object classification. The usage of some non-linear approaches like neural networks or mathematical morphology methods has provided better results [15]. However, the inherent complexity of many scenes (i.e. images with non-controlled illumination conditions or textured images) makes very difficult to achieve an optimal pixel classification into regions, due to the combinatorial nature of the task. Evolutionary image segmentation [8][13][19] has reported a good performance in relation to more classical segmentation methods. Our approach of modelling and solving image segmentation as a graph-partitioning problem is related to Shi and Malik's work [14]. They use a computational technique based on a generalized eigenvalue problem for computing the segmentation regions. Instead, we found that very acceptable results can be obtained, when applying a hierarchical social metaheuristic for image segmentation through a normalized cut solution.

2 Normalized Cut Problem

An important graph bipartition problem is the Min-Cut problem [1][5] defined for a weighted undirected graph $S = (V, E, W)$, where V is the ordered set of vertices or nodes, E is the ordered set of undirected arcs or edges and W is the ordered set of weights associated with each edge of the graph. This Min-Cut optimization problem consists in finding a bipartition G of the set of nodes of the graph: G=(C, C') such that the sum of the weights of edges with endpoints in different subset is minimized. Every partition of vertices V into C and C' is usually called a cut or cutset and the sum of the weights of the edges is called the weight of the cut or *similarity* (s) between C and C'. For the considered Min-Cut optimization problem, the cut or similarity between C and C' given by

$$w(C,C') = s(C,C') = \sum_{v\in C, u\in C'} w_{vu} \tag{1}$$

is minimized. In [9] is demonstrated that the decision version of Max-Cut (dual version of Min-Cut problem) is NP-Complete. This way, we need to use approximate algorithms for finding the solution in a reasonable time. In this paper we propose a new hierarchical social (HS) metaheuristic for finding an approximate solution to a variant of the Min-Cut problem called Normalized Cut problem [14].
The Min-Cut formulation has been used to Wu and Leahy [18] as a clustering method and applied to image segmentation. These authors look for a partition of the graph into k subgraphs such that the similarity (min-cut) among subgraphs is minimized. They pointed out that although in some images the segmentation is acceptable, in general this method produces an over-segmentation because small regions are favoured. To avoid this fact, in [14] other functions that try to minimize the effect of this problem are proposed. The optimization function called min-max cut is:

$$cut(G) = \frac{\sum_{v\in C, u\in C'} w_{vu}}{\sum_{v\in C, u\in C} w_{vu}} + \frac{\sum_{v\in C, u\in C'} w_{vu}}{\sum_{v\in C', u\in C'} w_{vu}} = \frac{s(C,C')}{s(C,C)} + \frac{s(C,C')}{s(C',C')} \tag{2}$$

where the numerators of this expression are the similarity $s(C,C')$ and the denominators is the sum of the arc weights belonging to C or C', respectively. It is important to remark that in an image segmentation framework, it is necessary to minimize the similarity between C and C' (numerators of eq. 2) and maximize the similarity inside C, and inside C′ (denominators of eq. 2). In this case, the sum of arcs between C and C' is minimized, and simultaneously the sums of weights inside of each subset are maximized. Reference [14] proposes an alternative cut value called *normalized cut*, which in general gives better results in image segmentation.

$$Ncut(G) = \frac{\sum_{v\in C, u\in C'} w_{vu}}{\sum_{v\in C, u\in C\cup C'} w_{vu}} + \frac{\sum_{v\in C, u\in C'} w_{vu}}{\sum_{v\in C', u\in C\cup C'} w_{vu}} = \frac{s(C,C')}{s(C,G)} + \frac{s(C,C')}{s(C',G)} \tag{3}$$

where $G=C\cup C'$. The normalized cut is characterized for maximizing the similarity inside each subset and minimizing the dissimilarity between subsets.

3 Hierarchical Social (HS) Algorithms.

This section presents the general features of a new metaheuristic called hierarchical social (HS) algorithms. In order to get a more general description of this metaheuristic, the reader is referred to [2][3][4]. This metaheuristic have been successfully applied to several problems as critical circuit computation [3], scheduling with unlimited resources [4] and MAX-CUT problem [2].

HS algorithms are inspired in the hierarchical social behaviour observed in a great diversity of human organizations and biological systems. The key idea of HS algorithms consists in a simultaneous optimization of a set of feasible solutions. Each group of a society contains a feasible solution and these groups are initially randomly distributed to produce a partition of the solution space. Using evolution strategies, where each group tries to improve its objective function or competes with the neighbour groups, better solutions are obtained through the corresponding social evolution. In this social evolution, the groups with lower quality tend to disappear. As a result, the rest of group objective functions are optimized. The process typically ends with only one group that contain the best solution found.

3.1 Metaheuristic structure

For the image segmentation problem, the feasible society is modelled by the specified undirected weighted graph $S=(V,E,W)$ also called feasible society graph. The set of individuals are modelled by nodes of the graph V and the set of feasible relations are modelled by edges E of the specified graph. The set of similarity relations are described by the weights W. Notice that the graph also models an image, where nodes model image pixels of the image and edges model the similarity between pixels.

Figure 1.a shows an example of the feasible society graph for a particular normalized cut problem.

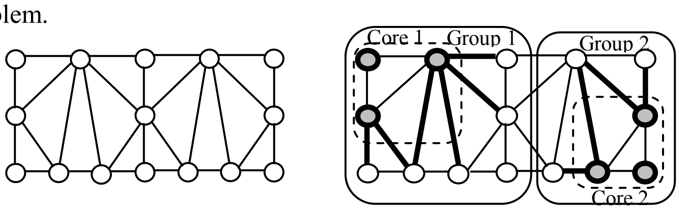

Fig. 1. (a) Feasible society graph. (b) Society partition and groups partition

The state of a society is modelled by a hierarchical policy graph [3][4]. This graph also specifies a society partition composed by a disjoint set of groups $\Pi=\{g_1,g_2,\ldots,g_g\}$, where each individual or node is assigned to a group. Each group $g_i \subset S$ is composed by a set of individuals and active relations, which are constrained by the feasible society. The individuals of all groups cover the individuals of the whole society. Notice that each group exactly contains one solution.

The specification of the hierarchical policy graph is problem dependent. The initial society partition determines an arbitrary number of groups and assigns individuals to groups. Figure 1.b is shows a society partition example formed by two groups.

A society partition can be classified into *monopoly partition*, in which there is only one group, and *competition partition*, in which there is more than one group. The example shown in Figure 1.b shows a competition partition. Each individual of a society has two objective functions: individual objective function $f1$ and the group objective function $f2$ that is shared by all individuals of a group. Furthermore each group g_i is divided into two disjoint parts: core and periphery. The core determines the value of the corresponding group objective function *f2* and the periphery defines the local search region of the involved group.

In the image segmentation problem framework, the set of nodes G_i of each group g_i is divided into two disjoint parts: $G_i=(C_i,C'_i)$ where C_i represents the core or group of nodes belonging to the considered cutset and C_i' is the complementary group of nodes. The core edges are the arcs that have their endpoints in C_i and C_i'. Figure 1.b also shows an example of core for the previous considered partition. The core edges or cutset edges of each group are shown in this figure by thick lines. For each group of nodes $g_i = (C_i, C'_i)$, the group objective function *f2*(*i*) is given by the corresponding normalized cut *Ncut*(*i*) referred to the involved group g_i:

$$f2(i) = NCut(i) = \frac{\sum_{v\in C_i, u\in C_i'} w_{vu}}{\sum_{v\in C_i, u\in C_i\cup C_i'} w_{vu}} + \frac{\sum_{v\in C_i, u\in C_i'} w_{vu}}{\sum_{v\in C_i', u\in C_i\cup C_i'} w_{vu}} = \frac{s(C_i,C_i')}{s(C_i,g_i)} + \frac{s(C_i,C_i')}{s(C_i',g_i)} \qquad (4)$$

$$\forall v\in g_i \quad f2(v,i) = f2(i) = NCut(i)$$

where $Gi=Ci\cup Ci'$ and the weights w_{vu} are supposed to be null for the edges that do not belong to the specified graph.

For each individual or node *v*, the individual objective function *f1*(*v*,*i*) relative to each group $g_i = (Ci, Ci')$ is specified by a function that computes the increment in the group objective function when an individual make a movement. There are two types of movements: Intra-group movement and inter-group movement. In the intra-group movement there are to possibilities: the first one consists in a movement from ***C_i* to *C_i'***, the second one is the reverse movement (***C'_i* to *C_i***). The inter-group movement is accomplished by individual v that belong to generic group X $(X = \Pi \setminus g_i)$ that want to move from X to g_i. There are two possibilities: the first one consists in a movement form ***X* to *C_i***, the second one consists in a movement from ***X* to *C'_i***. The next formula shows the movement $C_i \rightarrow C'_i$.(described by the function *C_to_C'(v,i)*)

$$f1(v,i) = C\_to\_C'(v,i) = \frac{s(C_i,C_i') - \sigma'(i) + \sigma(i)}{s(C_i,gi) - \sigma'(i)} + \frac{s(C_i,C_i') - \sigma'(v,i) + \sigma(v,i)}{s(C_i',g_i) + \sigma(v,i)} \qquad (5)$$

$$\text{where} \quad \sigma(v,i) = \sum_{u\in C_i} w_{vu} \;\; and \;\; \sigma'(v,i) = \sum_{u\in C_i'} w_{vu}$$

This function allows to select for each individual *v*, the group which achieves the corresponding minimum value. The other movements ($C'_i \rightarrow C_i$, $X \rightarrow C_i$,$X \rightarrow C'_i$) have similar functions and expressions.

The HS algorithms here considered, try to optimize one of their objective functions (*f1* or *f2*) depending on the operation phase. During an autonomous or winner phase, each group g_i aims to improve independently the group objective function *f2*. During a loser phase, each individual tries to improve the individual objective function *f1*, the original groups cohesion disappeared and the graph partition is modified in order to optimize the corresponding individual objective function. The strategy followed in a loser phase could be considered inspired in Adams Smith's "invisible hand" economic society mechanism described in his book "An Inquiry into the Nature and Causes of the Wealth of Nations".

3.2 Metaheuristic process

The algorithm starts from a random set of feasible solutions. Additionally for each group, an initial random cutset is derived. The groups are successively transformed through a set of evolution strategies. For each group, there are two main strategies: *winner or autonomous strategy* and *loser or competition strategy*. The groups named *winner groups*, which apply the winner strategy, are that which have the lowest group objective function *f2*. The rest of groups apply loser strategy and are named *loser groups*. During optional *autonomous phases* in between competition phases, all groups behave like winner groups. These optional autonomous phases improve the capability of the search procedure.

The winner strategy can be considered as a local search procedure in which the quality of the solution contained in each group is improved by autonomously working with the individuals that belong to this group.

The loser strategy is oriented to let the interchange of individuals among groups. In this way the groups with lower quality tend to disappear because their individuals move from this group to another group with higher quality.

Winner and loser strategies are the basic search tools of HS algorithms. Individuals of loser groups, which have higher group objective functions, can change their groups during a loser strategy for improving their respective individual objective function. This way, the loser groups tend to disappear in the corresponding evolution process. Individuals of winner groups, which have lowest group objective functions, can move from the core to periphery or inversely, in order to improve the group objective function. These strategies produce dynamical groups populations, where group annexations and extinctions are possible.

3.3 High-level pseudo-code

The general high-level description of an HS metaheuristic for the image segmentation problem is shown in Figure 2.

```
Procedure Hierarchical_Social_Algorithm(S)
Var
 S=(V,E,W):Initial_Society_graph;
 GS={gi}:Groups_structure;
 F1={f1}:Individuals_objective_function_structure;
 F2={f2}:Groups_objective_function_structure;
 i,k:1..Number_of_groups /*Group indices*/
Begin                                /* Begin social evolution */
 GS=Get_initial_random_partition_and_groups_structure();
 Repeat  /* group evolution*/
  Compute_F1()_and_F2(); /*Objective functions*/
  For each gi in GS
   If f2(i)=min{f2(k)∀k} Or Autonomous_phase
    Then Winner_Strategy(i) Else Loser_Strategy(i);
  End For
  Update_groups_structure(GS);
 Until termination_condition_met;    /*End of social evolution*/
 Return(GS); /* Approximate optimal solution */
End Hierarchical_Social_Algorithm
```

Fig. 2. High-level pseudo-code for Hierarchical Social Algorithms

The Winner Strategy is oriented to improve the normalized cut value of a group. For a given group of nodes g_i, an interchange of nodes between core and periphery allows to optimize the group objective function. The interchange is accomplished if there is an improvement (minimization) of the normalized cut weight of the corresponding group g_i. We allow to interchange the nodes between C_i and C'_i in parallel and simultaneously on a single iteration. The only restriction is the following: if one node v is gone out from C_i (or C'_i), none of its adjacent nodes can change their position in the same iteration. This restriction avoids node cycling in the corresponding procedure.

The winner strategy is easily specified in the pseudo-code of Figure 3, taking into account the particularization of group objective function of one generic individual v to movements $C_i \rightarrow C'_i$ and $C'_i \rightarrow C_i$ restricted to the group $gi=(C_i,C_i')$.

```
Procedure Winner_Strategy(i)/*For Normal_Min_Cut problem*/
Var Gi=(Ci,Ci´):Nodes_partition_of_considered_group_gi;
Begin
For v = 1 to Number_of_nodes_of_considered_group_gi
  If v∈Ci and C_to_C′(v,i) < f2(v,i) Then Ci′= Ci′∪{v}; Ci= Ci\{v};
  If v∈Ci´and C′_to_C(v,i) < f2(v,i) Then Ci′= Ci′\{v}; Ci= Ci∪{v};
End For
End Winner_Strategy
```

Fig. 3. Pseudo-code of Winner Strategy

The Loser Strategy is oriented to allow group changing. The individuals that belong to groups with lower cut value can change their group in order to increase the individual objective function *f1*. Each node searches for the group that gives the best improvement in its individual objective function. Figure 4 shows the pseudo-code of Loser Strategy considering the functions $\sigma n(v,i)$ and $\sigma n'(v,i)$ previously defined (5).

```
Procedure Loser_Strategy(i) /*For Normal_Min_Cut problem*/
Var
 gi= (Ci,Ci´) :Nodes_partition_of_the_considered_group_gi
 G= {g1,..gk} :Groups_partition_of_the_graph
 i,j,k:1..Number_of_groups /*Group indices*/
Begin
For v = 1 to Number_of_nodes_of_considered_group
  j = arg min {X_to_C(v,k), X_to_C′(v,k)} ,∀k} }; /*j= host group*/
  If (v∈Ci) and (j≠i) Then Ci=Ci\{v}
   Else If (v∈C′i) and (j≠i) Ci´=Ci´\{v}; /*Remove v from gi*/
  If X_to_C (v,j)< X_to_C′(v,j) Then Cj′=Cj′∪{v}
   Else Cj=Cj∪{v}; /*Add v to gj*/
End For
End Loser_Strategy
```

Fig. 4. Pseudo-code of Loser Strategy.

4 Proposed HS Brightness Image Segmentation Method

Given an image to be segmented, we first construct its simplified undirected image weighted graph $S = (V, E, W)$. This graph is defined with all the image pixels as nodes and setting the edge weights with a measure of spatial and grey level difference distances (similarity) between the corresponding endpoints. We can define the graph edge weight w_{ij} connecting the two nodes i and j by the conditional function:

$$If\ (abs(x_i - x_i) < r_x)\ Then\ w_{ij} = e^{\frac{-(I_i - I_j)^2}{\sigma_I^2}} \cdot e^{\frac{-(x_i - x_j)^2}{\sigma_x^2}}\ Else\ w_{ij} = 0 \quad (6)$$

where r_x is an experimental threshold value, I_i is the grey level intensity of one pixel i, and x_i is the spatial location this pixel. The values of σ_I, σ_x and r_x are adjusted experimentally and in general they depend on the characteristic of the image. Non-significant weighted edges, according to defined similarity criteria, are removed from the image graph. Using this graph, a high-level pseudocode of the proposed HS segmentation algorithm is described by 4 main steps in Figure 5.

```
1.  Construct the related image weighted graph as described above.
2.  Compute an approximate optimal solution for the NCut of graph using
    the proposed HS algorithm obtaining a two-region graph partition
    /*Components correspond to the most significant regions in image*/
3.  If the current partition is not an acceptable segmentation result,
    then apply step 2 to each region of the obtained bipartition.
4.  Convert the union of all intermediate graph partitions into the
    segmented image solution.
```

Fig. 5. High-level pseudo-code of HS algorithm for the segmentation problem

5 Experimental Results

The computational experiments were evaluated in a Intel Pentium 4, at 1.7 GHz, with 256 MB of RAM. All algorithms were coded by the same programmer in C++ without code optimization. We compared the performance of HS metaheuristic applied to normalized cut problem with an adapted solution for the same problem using a standard genetic algorithm as proposed by Dolezal et al [1] for the Max-Cut problem. Some main details of the metaheuristics implementation are the following:

1. For the proposed *HS algorithm*, the number of groups was *nodes*/100 and number of autonomous iterations was 20.
2. For the implemented *genetic algorithm*, the initial population was 50 individuals, the maximum number of generations was 100, the probability of crossover and mutation were 0,6 and 1/*nodes*, respectively.

Both approaches were tested on several real and synthetic images. The segmentation results for two real images are respectively shown in Figures 6 and 7.

Table 1. Comparison between GA and HSA for de first NCut bipartition value for 4 images.

Image graph			Parameters			GA		HSA	
Image	Nodes	Arcs	σ^2_I	σ^2_X	r_X	NCut	Time(s)	NCut	Time(s)
Pout97x80	7760	251924	0.050	5	10	0.08921	2845	0.03149	716
Hurricane100x80	8000	1103388	0.007	15	15	0.16823	12221	0.02155	3168
Sint1_20x20	400	7414	0.030	10	4	0.08132	43	0.02886	< 1
Sint2_20x20	400	7414	0.030	10	4	0.09125	51	0.01802	< 1

Table 1 represents the comparative results between the genetic and HS algorithms for several real image (rows 1 and 2) and synthetic images (rows 3 and 4).

First and second rows of Table 1 shows the experimental results for images of figures 6 and 7. Its columns respectively represent the name and size of images, their corresponding image graphs (number of nodes and edges), the parameters that define the edges weights (σ_I, σ_x, r_x), and the results for the respective genetic and HS algorithms. For both algorithms, we show a segmentation quality result (for the first value of the *NCut*) and a computational result (execution time in seconds).

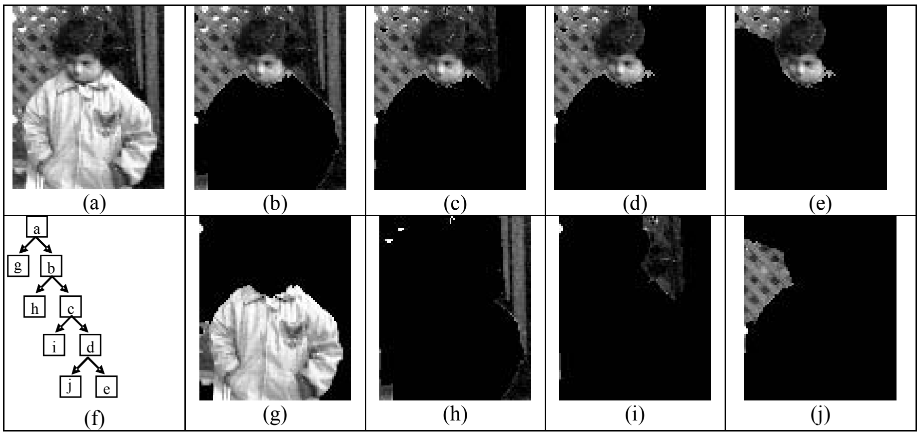

Fig. 6. Segmentation results for image *Pout*: (a) Initial image. (f) Structure of the segmentation tree. (b)..(j) Resulting segmented regions according to the segmentation tree.

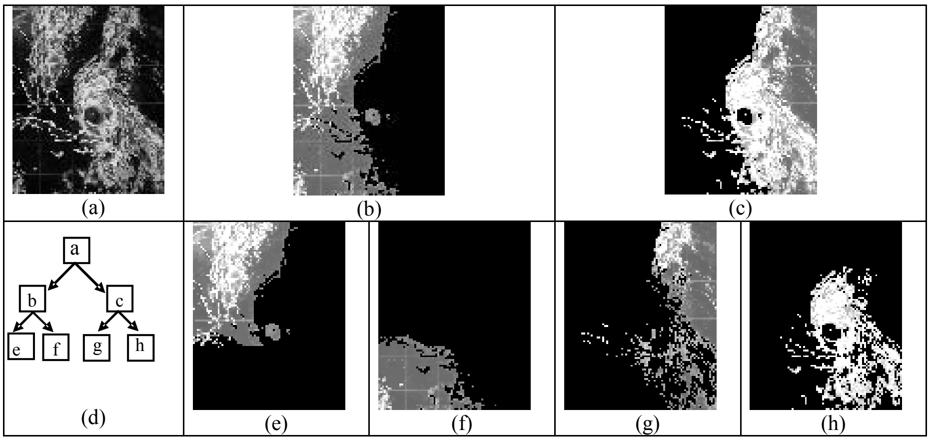

Fig. 7. Segmentation results for image *Hurricane*: (a) Initial image. (d) Structure of the segmentation tree. (b)..(h) Resulting segmented regions according to the segmentation tree.

6 Conclusions

This paper has introduced an HS algorithm to efficiently solve the region-based image segmentation problem. First, the image problem has been transformed into a normalized cut problem. The HS algorithms were introduced to exploit the power of competition and cooperation among different groups in order to explore the solution space. We have experimentally shown that the proposed HS algorithms provide high quality qualitative segmentation solutions with lower computation times.

The major advantage of using a normalized cut as group objective function is that the quality of the segmentation region is very high. However, the capability of the algorithm can be improved segmenting the image in several regions instead of two regions. In this case the normalized cut is not an adequate function, because it is not defined for several cuts. We propose as a future work the use of other group objective functions (such as K-means) in order to exploit all the potentiality of the HS metaheuristic.

7 References

1. O. Dolezal, T. Hofmeister, and H. Lefmann, H, "A comparison of approximation algorithms for the MAXCUT-problem", *Reihe CI 57/99*, Universität Dortmund, 1999.
2. A. Duarte, F. Fernández, A. Sánchez, A. S. Montemayor, "A Hierarchical Social Metaheuristic for the Max-Cut Problem", To appear in Proc. of EvoCOP '04.
3. F. Fernández, A. Duarte and A. Sánchez, "A Software Pipelining Method based on a Hierarchical Social Algorithm", *Proc. 1st* MISTA 2003, pp. 382-385, 2003.
4. F. Fernández, Software Pipelining using HS Metaheuristic, *Tech. Rep. URJC*, Spain, 2003.
5. P. Festa, P.M. Pardalos, M.G. Resende and C.C. Ribeiro, "Randomized Heuristics for the Max-Cut Problem", *Optimization Methods and Software*, Vol. 7, pp. 1033-1058, 2002.
6. F. Glover and G.A. Kochenberger (eds.), *Handbook of Metaheuristics*, Kluwer, 2002.
7. R.C. Gonzalez and R. Woods, *Digital Image Processing*, 2nd Edition, Prentice Hall, 2002.
8. S.Y. Ho and K.Z. Lee, "Design and Analysis of an Efficient Evolutionary Image Segmentation Algorithm, *J. VLSI Signal Processing*, Vol 35, pp. 29-42, 2003.
9. R.M. Karp, Reducibility among Combinatorial Problems, R. Miller and J. Thatcher (eds.): *Complexity of Computer Computations*, Plenum Press, pp. 85-103, 1972.
10. Michalewicz, *Genetic Algorithms+Data Structures=Evolution Programs*, Springer, 1996.
11. Z. Michalewicz, D.B. Fogel, *How to Solve It: Modern Heuristics*, Springer, 2nd Ed, 2000.
12. J. R. Parker, *Algorithms for Image Processing and Computer Vision*, John Wiley, 1996.
13. R. Poli, "Genetic programming for image analysis", J. Koza (ed): *Genetic Progr.,* 1996.
14. J. Shi and J. Malik, "Normalized Cuts and Image Segmentation", *IEEE Trans. Pattern Analysis and Machine Intelligence*, Vol. 22, no. 8, pp. 888-905, Aug. 2000.
15. M. Sonka et al. *Image Processing, Analysis and Machine Vision*, 2nd Ed., PWS, 1999.
16. W.M. Spears, *Evolutionary Algorithms*, Springer, 1998.
17. S. Voss, "Meta-heuristics: The State of the Art", A. Nareyek (ed.): *Local Search for Planning and Scheduling*, LNAI 2148, Springer, pp. 1-23, 2001.
18. Z. Wu et al, "Optimal Graph Theoretic Approach to Data Clustering: Theory and its Application to Image Segmentation", *IEEE Trans. PAMI*, V. 15, n. 11, pp. 1101-1113, 1993.
19. M. Yoshimura and S. Oe, "Evolutionary Segmentation of Texture Image using Genetic Algorithms", *Pattern Recognition*, Vol. 32, pp. 2041-2054, 1999.