Technische Universität München
Fakultät für Informatik
Prof. N. Navab, Ph.D.

SS 2013
Exercise Sheet 8
June 27, 2013

Exercises in Computer Aided Medical Procedures II

Exercise 1 (P) Random Walks for Image Segmentation

In this exercise you will implement the Random Walks for Image Segmentation algorithm [1]. The main function has been provided in the file `runRandomWalks.m`. The code for manual/predefined seed placement is given, as well as the functions to visualize the results. You will focus on the implementation of the core algorithm. For this you need to create the missing function `solveRw.m`, following the lecture slides and the indications bellow.

a) Compute the edge weights of the graph using a 4-neighborhoud connectivity: (**Hint:** `MATLAB` functions ind2sub and sub2ind might be of help)

- Calculate first the weights as the absolute value of the intensity difference for every pair of connected neighbours.

$$w_{ij} = |A(p_i) - A(p_j)|$$

where A is the image, and p_i and p_j are two neighbour pixels. Save the results in a **sparse** matrix W (**Hint:** use the `sparse` function of `MATLAB`).

- Normalize the weights in W such that their values are between 0 and 1

$$W = (W - \min(W(:)))./(\max(W(:)) - \min(W(:)) + \texttt{eps});$$

where `eps` is the matlab function determining the floating point spacing

- Compute the final exponential weights

$$w_{ij} = ((\exp(-\beta * w_{ij})) + \varepsilon);$$

with $\varepsilon = 10^{-6}$ and β provided by the user (set to 90 in `runRandomWalks.m`)

b) Compute the graph Laplacian matrix from the final exponential weight matrix. **Hint:** L should be a sparse matrix

c) Assemble L_u by selecting rows and columns corresponding to unmarked nodes in the Graph-Laplacian (see Fig.1) **Hint:** L_u should be a sparse matrix.

d) Assemble M by creating a matrix, with one column per label (here 2 columns: label 1 = foreground, label 2 = background) and one row per seed node. **Hint:** M should be a sparse matrix.

- In the first column, all the seeds corresponding to label 1 will be filled with a 1, and all the seeds with label 2 will have value 0.
- In the second column, the roles are inverted, all the seeds with label 1 get assigned a 0 and those with label 2 will have value 1.

See lecture slides and Fig.1 for more details.

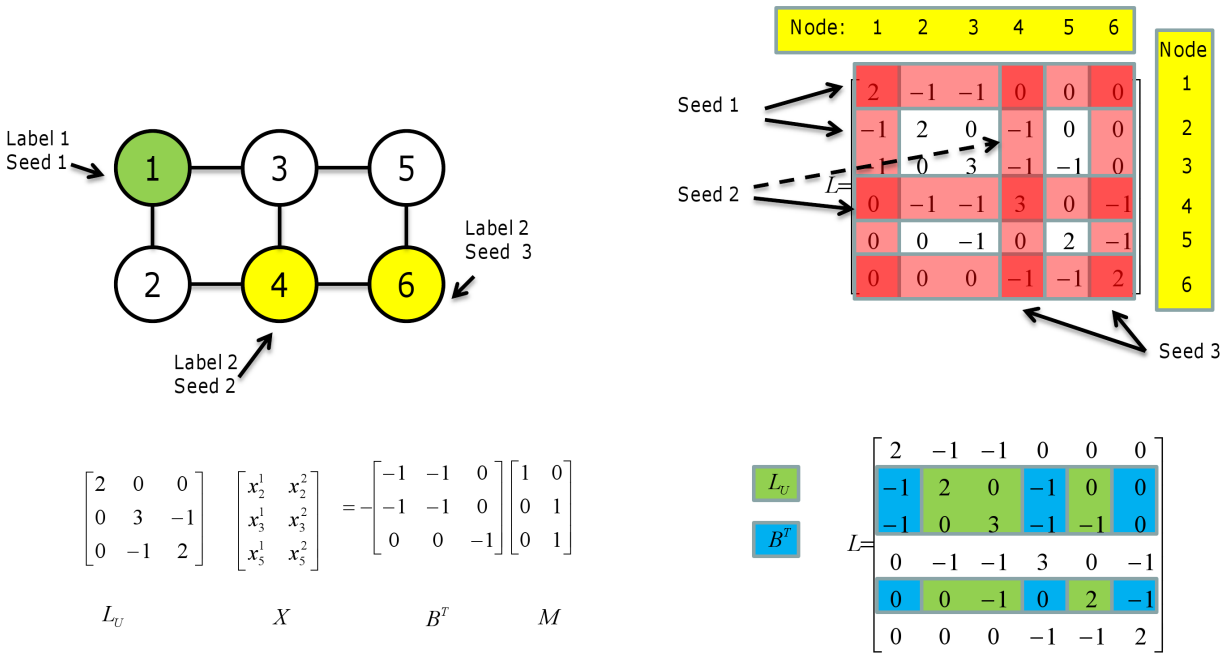

Figure 1: Instead of decomposing L into marked and unmarked nodes one can extract directly all the necessary matrices from L. This figure demonstrates how to assemble L_u and B^T directly from L by selecting the corresponding matrix elements. Note that the rows and columns of L correspond to the graph nodes.

e) Assemble the system of linear equations for solving the Random Walks problem (see lecture slides and Fig.1). **Hint:** All the necessary matrices should at this point be available and **sparse**.

f) Solve the linear system using `MATLAB` "\" operator . **Hint:** For our 2-label (fore-/background) example we need to solve only for one label (see lecture slides).

g) As a result the `solveRw.m` function will return the computed probabilities of each pixel to belonging to each label, as well as a mask. The mask will be an image of the same size as the original but where each pixel takes one of two possible values (1=foreground, 2=background).

For the attestation prepare to visualise the results both for the manual positioning of the seeds, as well as for the predefined seeds in files `CTAbdomenBackground.png` and `CTAbdomenForeground.png`

References

[1] L. Grady. Random walks for image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(11):1768–1783, 2006.