An Parallel Computing Project Abstract On

Connected Component Labelling

Submitted by

Divija Nagaraju-14IT112

Mukta Kulkarni-14IT220

Pooja Soundalgekar-14IT230

V Sem B.Tech (IT)

in partial fulfillment for the award of the degree

of

Bachelor of Technology

In

Information Technology

At

Department of Information Technology

National Institute of Technology Karnataka, Surathkal.

September 2016

Abstract

Connected-component labeling (alternatively called region extraction) is an algorithmic application of graph theory, where subsets of connected components are uniquely labeled based on a given heuristic. It is used in computer vision to detect connected regions in binary digital images, although color images and data with higher dimensionality can also be processed.

Connected component labeling methods are classified into 4 different categories :

- Repeat pass,one component at a time
- Two Pass way
- Hierarchical Tree structure
- mesh and hypercube parallel processors.

This project is a parallel implementation of Suzuki's and Two pass algorithm i.e. the concepts mentioned in the first two categories.

SUZUKI'S Algorithm

The Suzuki algorithm scans through the image in forward and backward directions using 2 separate masks and assigns a provisional label to each pixel. These are stored in a separate one dimensional array. For the first scan these label values are assigned based on their 8-connected neighbourhood. In the next scans, the forward and backward scans are performed repeatedly and alternately for the entire image.

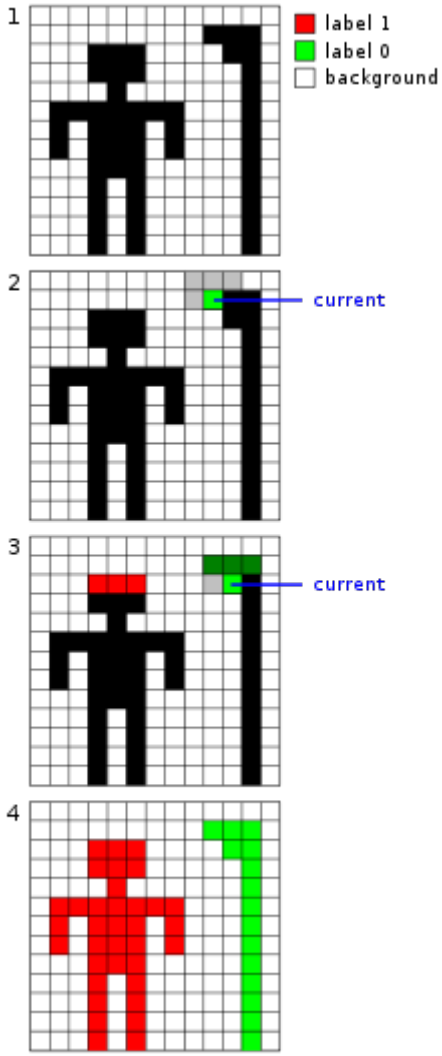
The following is a pseudo code of the Suzuki algorithm:

```
algorithm Suzuki(data)
    First pass
    for row in data
        for column in row
            assign a label to data[row][col] using Forward Mask
    Next passes
    while there is a change in labels do
        for row in data
            for column in row
                update the labels[row][column] using Backward Mask

        for row in data
            for column in row
                update the labels[row][column] using Forward Mask

return labels
```

Charachteristics of the algorithm :

Suzuki’s algorithm promises a linear time complexity. Suzuki et al. show that maximum of four scans is sufficient for the images with complex geometrical shapes.
The Suzuki algorithm is sequential in nature since the result of the previous iterations is used in the next one.
Hence, to exploit parallelism irrespective of the data dependencies a method similar to pipelining is used. This implies that each thread continues iterations of the row of the previous thread while that thread moves on to the next row. For this synchronisation to take place, an additional 2 dimensional array is maintained which stores the result of the condition checks.

The following figures show the components analysed in a binary image.



Two Pass Algorithm

The two pass algorithms for labeling connected components use a standard Union Find Algorithm for storing label equivalences and a decision tree for scanning. The usage of the decision tree reduces the number of neighbours to be examined for each pixel.The pseudo code for the algorithm is:

```
function TwoPass1(image)
        ScanTwoPass1 (image) -->Processes image lines using forward mask
        flatten(p, count) -->Smallest equivalent label is assigned
        for row in image do
                for col in row do
                        label(e) ← p[label(e)]
end function
```

It processes 2 lines at a time and 2 passes at a time, therefore uses a different mask.

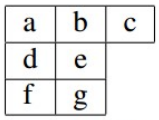


Figure shows the mask used for scanning the image

Both e and g are assigned labels simultaneously. If both e and g are background pixels, then nothing needs to be done. If e is a foreground pixel and there is no foreground pixel in the mask, we assign a new provisional label to e and if g is a foreground pixel, we will assign the label of e to g. If there are foreground pixels in the mask, then we assign e any label assigned to foreground pixels. In this case, if there is only one connected component in the mask then there is no need for label equivalence. Otherwise, if there are more than one connected components in the mask and as they are connected to e, all the labels of the connected components are equivalent labels and hence need to be merged.

Parallel Implementation

In the parallel implementation of the algorithm, the image is divided row-wise into chunks of equal size and given to the threads. A simple lock mechanism is used while merging to resolve dependencies.

The algorithms are to be implemented in C/C++ with OpenMP directives.