CSC 323 Spring 2015: GraphColoring (C++)

Ravi Patel

April 28, 2015

**Algorithm steps:**

step 0: prepare and initialize everything

- load adjacentMatrix from the input pairs

- print the matrix

- set nodeColor array to zero

- newColor <-- 0

// get new new color (using 1, 2, 3, ... as the color scheme)

step 1: newColor ++

//use the newColor to color as many un-colored nodes as possible

Step 2: newNode <-- get an uncolor node

if newNode does not have any adjacent node that are already colored with the newColor

nodeColor[newNode} <-- newColor

Step 3: repeat step 2 until all uncolored nodes are checked.

// for debugging purpose, ...

Step 4: print (to save paper)

(a) print the newColor, one text line

(b) print the index of nodeColor array in one text line;

(c) print the nodeColor lineup with the index above, in one text line.

Step 5: repeat steps 1 to 3 until all uncolored nodes are checked.

Step 6: copy nodeColor[i] to adjacnetMatrix[i][i]

Step 7: print newColor

print adjecantMatrix

Source Code:

#include <iostream>

#include <fstream>

using namespace std;

bool checkColor(int\* nodeColor, int NumberOfNodes){

for (int row=1; row < NumberOfNodes ; row++) {

if(nodeColor[row]==0){

return true;

}

}

return false;

}

void printNodeColor(int\* nodeColor, int NumberOfNodes, ostream& outputFile){

for (int row=1; row < NumberOfNodes ; row++) {

outputFile<<nodeColor[row];

}

outputFile<<endl;

}

int main( int argc , char\* argv[]){

ifstream inputFile;

inputFile.open(argv[1]);

ofstream outputFile;

outputFile.open(argv[2]);

int NumberOfNodes, from, to;

inputFile>>NumberOfNodes;

cout<<NumberOfNodes<<endl;

int\*\* matrix;

int\* nodeColor;

NumberOfNodes++;

matrix = new int\* [NumberOfNodes];

nodeColor = new int [NumberOfNodes];

int newColor = 0;

int newNode=1;

for(int row=0; row<NumberOfNodes; row++){

matrix[row] = new int[NumberOfNodes];

for(int col=0; col<NumberOfNodes; col++){

matrix[row][col]=0;

}

}

for(int row=0; row < NumberOfNodes; row++){

nodeColor[row]=0;

}

while(inputFile>>from && inputFile>>to){

for (int row=0; row < NumberOfNodes; row++) {

for(int col=0; col < NumberOfNodes; col++){

if(row==from && col==to){

matrix[from][to]=1;

}

}

}

}

while(checkColor(nodeColor, NumberOfNodes)){

newColor++;

for(int row=1; row < NumberOfNodes; row++){

if(nodeColor[row]==0){

newNode = row;

}

for (int col=1; col < NumberOfNodes; col++) {

if (matrix[row][col] == 0 || matrix[row][col] == newColor ) {

break;

}

nodeColor[newNode] = newColor;

}

}

outputFile<<"New Color: "<<newColor<<endl;

outputFile<<"Node Color Array: ";

printNodeColor(nodeColor, NumberOfNodes, outputFile);

}

for(int row=0; row<NumberOfNodes; row++){

for(int col=0; col<NumberOfNodes; col++){

if(row == col){

matrix[row][col]=nodeColor[row];

}

outputFile<<matrix[row][col]<<" ";

}

outputFile<<endl;

}

}

Input1:

Output1:

Input2:

Output2:

Input3:

Output3: