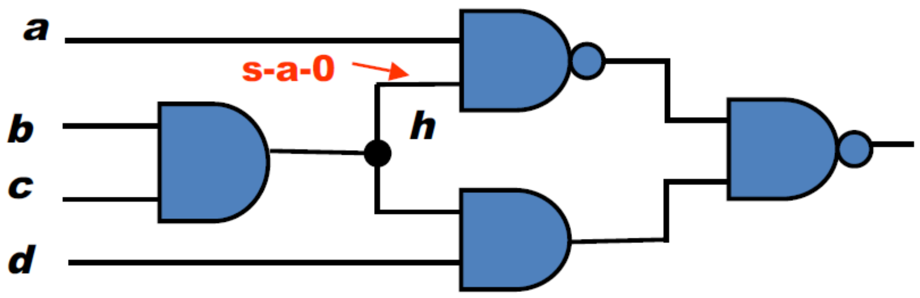
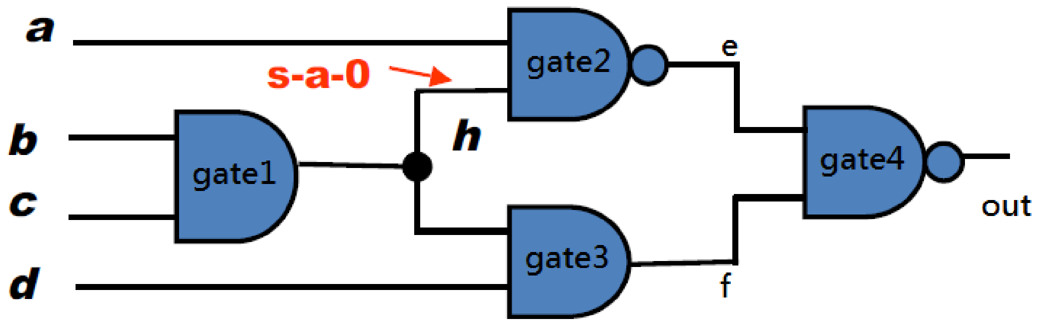
1. **(10 points)** Please go through the steps of D-algorithm to find an input vector that can detect stuck-at-0 fault at node h.



I rename nodes and gates as following figure.



Step1: Test for s-a-0 at h, set h= D

Step2: Make gate3 as D-Frontier.

Here, I pass D signal form the following path. if choosing gate2 and gate3 both as D-Frontiers, the output of gate4 can't get any D representation. (e = D', f = D, so out should be 1.)

Step3: [Sensitize gate3] set d = 1 to pass D to f.

And make gate4 as next D-Frontier.

In addition, adds gate1 to J-Frontier.

Step4: [Sensitize gate4] set e= 1 to pass D to out, and then out is D'

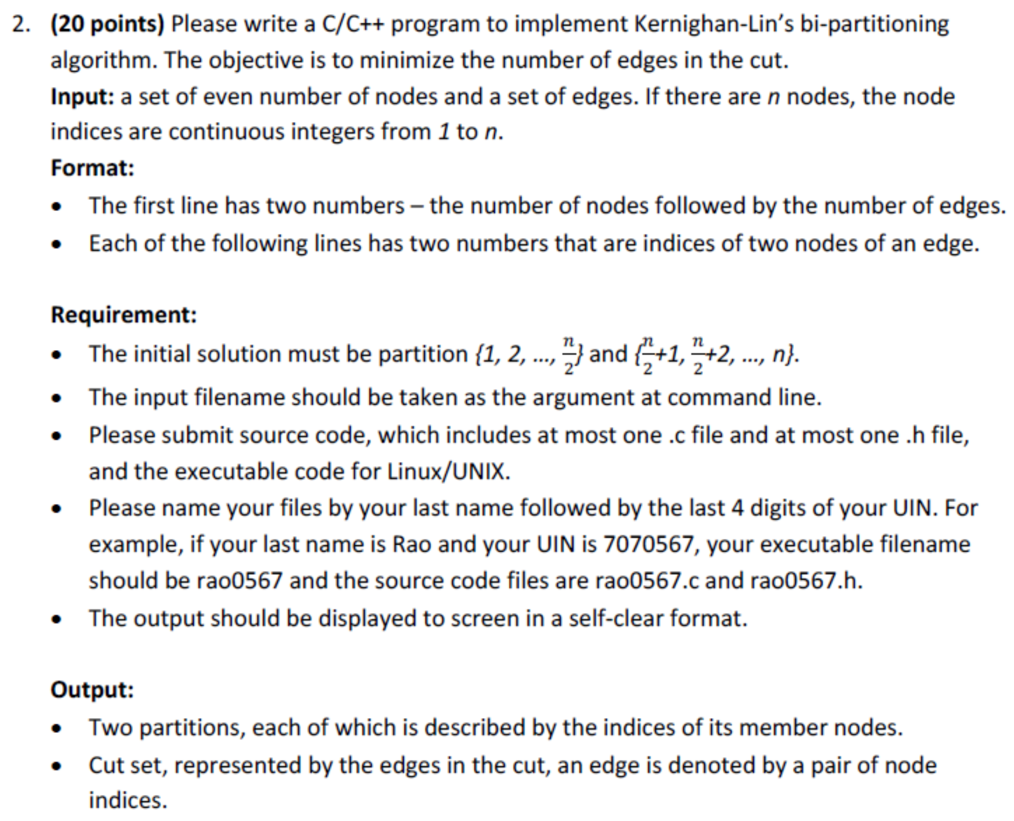
adds gate2 to J-Frontier.

Step5: [justification for gate 1], b = c = 1. Then, h can be D to test s-a-0 case.

Step6: [justification for gate 2], a = 0. Then, e can be 1.

Therefore, I found vector [a, b, c, d] = [0,1,1,1] to test this s-a-0 case at h.

If out (D') is 0 means this circuit (node h) is OK. Otherwise, h is stuck at 0.



2. **(20 points)** Please write a C/C++ program to implement Kernighan-Lin's bi-partitioning algorithm. The objective is to minimize the number of edges in the cut.
   Input: a set of even number of nodes and a set of edges. If there are *n* nodes, the node indices are continuous integers from *1* to *n*.
   Format:
   - The first line has two numbers – the number of nodes followed by the number of edges.
   - Each of the following lines has two numbers that are indices of two nodes of an edge.

   Requirement:
   - The initial solution must be partition $\{1, 2, \ldots, \frac{n}{2}\}$ and $\{\frac{n}{2}+1, \frac{n}{2}+2, \ldots, n\}$.
   - The input filename should be taken as the argument at command line.
   - Please submit source code, which includes at most one .c file and at most one .h file, and the executable code for Linux/UNIX.
   - Please name your files by your last name followed by the last 4 digits of your UIN. For example, if your last name is Rao and your UIN is 7070567, your executable filename should be rao0567 and the source code files are rao0567.c and rao0567.h.
   - The output should be displayed to screen in a self-clear format.

   Output:
   - Two partitions, each of which is described by the indices of its member nodes.
   - Cut set, represented by the edges in the cut, an edge is denoted by a pair of node indices.

Use such command to compile my code. (Only one cpp file, hsu1725.cpp)

```
g++ -std=c++11 Path/hsu1725.cpp -o Path/hsu1725.out
```

For example, my Path: /home/grads/w/**NetID**/ece687

How to test, put hsu1725.out and test file in the same folder.
The filename input by command line.

```
./hsu1725.out Your_Test_FileName
```

For example, my Test_FileName: test_input1

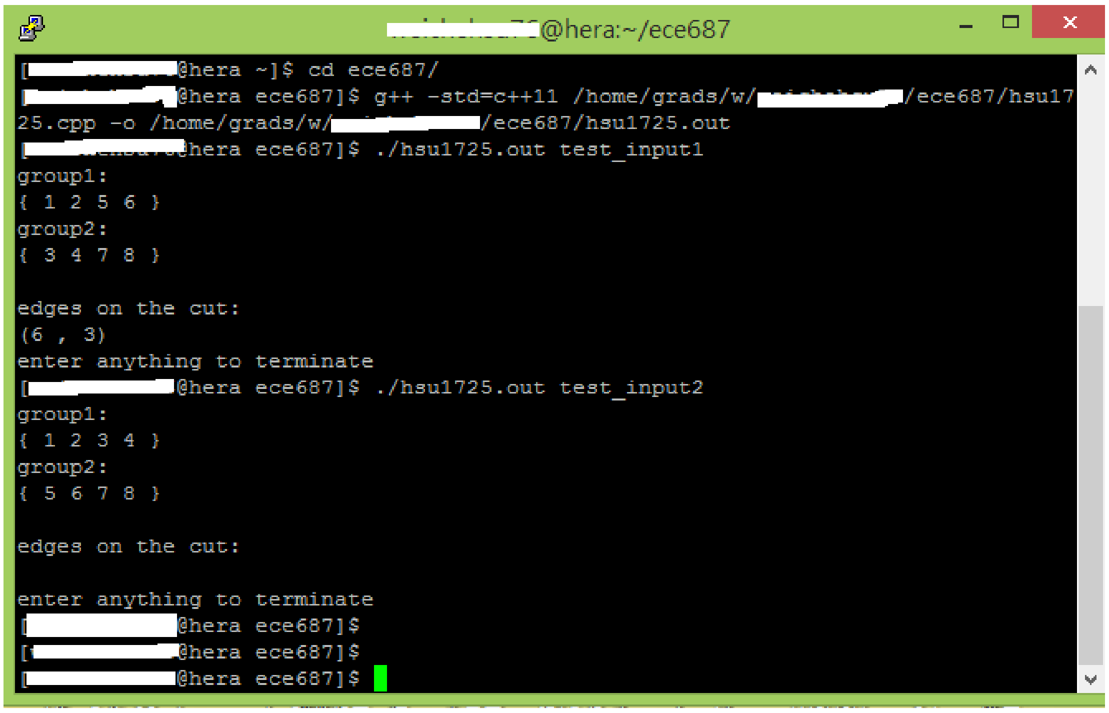
[Following command I used for the following Figure]

```
g++ -std=c++11 /home/grads/w/NetID/ece687/hsu1725.cpp
-o /home/grads/w/NetID/ece687/hsu1725.out
./hsu1725.out test_input1
./hsu1725.out test_input2
```

[Test case1] test_input1 is the same example in the lecture 13 note, page24.
[Test case2] test_input2 is one example, and the node 1~4 connect into one group, and node 5~8 connect into the other group. Thus, no edge on the cut, and it does not need to swap any node.

I compile and run my program on ECE server hera.

The result will show the “2 partition groups” and “all edges on the cut”.



[redacted]@hera:~/ece687

```
[[redacted]@hera ~]$ cd ece687/
[[redacted]@hera ece687]$ g++ -std=c++11 /home/grads/w/[redacted]/ece687/hsu17
25.cpp -o /home/grads/w/[redacted]/ece687/hsu1725.out
[[redacted]@hera ece687]$ ./hsu1725.out test_input1
group1:
{ 1 2 5 6 }
group2:
{ 3 4 7 8 }

edges on the cut:
(6 , 3)
enter anything to terminate
[[redacted]@hera ece687]$ ./hsu1725.out test_input2
group1:
{ 1 2 3 4 }
group2:
{ 5 6 7 8 }

edges on the cut:

enter anything to terminate
[[redacted]@hera ece687]$
[[redacted]@hera ece687]$
[[redacted]@hera ece687]$
```