Answer 1:

The purpose of generating suitable Benchmark Graphs for this problem is essentially to compare the results of a graph-partitioning heuristic under scrutiny with t

do the same with partition 2…

repeat the above steps until total cut-set reaches maximum…

when no further increase in cut-set is possible, *take the complement of the whole graph.* The maximum cut-set would (should?) then become the minimum cut-set of this new *sparse graph.*

}

(This alg

$g_{14} = -2$ $g_{15} = -3$ $g_{16} = -3$ $g_{24} = -1 - 1 - 2 = -4$ $g_{25} = -1 - 2 = -3$
$g_{26} = -1 - 2 = -3$ $g_{34} = -1 - 1 = -2$ $g_{35} = -1 - 2 = -3$ $1\ 2 = -3$ 3

$g_{35} = 1 + 2 – 0 = 3$

3, 5 are swapped.

Step 4: Update D Values:

A’ = {} B’ = {}

Since A’ an B’ are Empty, go to step 5

Step 5: Find maximum G:

$g_1 = -2$, $g_2 = -1$, $g_3 = 3$

Code for Part

```
void main (void)
{
	//declare variables
	int Time = 0, count, Current_Cost;
	int Sol_A[6], Sol_B[6];
	double Temp = T0, Cycle_Time = M;
	time_t t;
	FILE gm0.595903 0 Td (g 0.5)Tj 0.595903 0 Td ( ..5)Tj 0.595903 0 Td5i
```

```
FILE *Result;

// open file for writing
Result = fopen("c:\\Results.txt", "a+");
/*fprintf(Result, "\nBest\tCost(
```

```
                for (j=0; j<=4; j++)
                {
                        for (k=0; k<=2; k++)
                        {
                                if (Netlist[Sol_B[i]-1][k+1] == Sol_A[j])
                                {
                                        if(NetOfBIsCut != 0)
                                                break;
                                        Cost += Netweight[Sol_B[i]-1];
                                        NetOfBIsCut = 1;
                                }
                        }
                }
        }
        return Cost;
}
```