RESOLVING THE RESOURCE ALLOCATION CONFLICT IN CIRCUIT SWITCHING USING MAXIMUM MATCHING

Manthan.D.Sanghavi[1], Shantan Sawa[2]

[1]UG student, Computer Science and Engineering, VIT University, Tamil Nadu, India
[2]UG student, Computer Science and Engineering, VIT University, Tamil Nadu, India

Abstract

In circuit switching during the set-up phase a case might arise where multiple systems (switch/source) might request for the same system which leads to conflict as the resources of a particular system can only be allotted to one system at a time. The whole purpose of our proposed algorithm is to resolve the conflict and achieve optimal solution such that almost all systems are being allotted to a suitable system and there is a dedicated path from the source to the desired destination. The proposed algorithm intends to use graph theory concepts like bipartite graphs and maximum matching to resolve the conflict. The algorithm also provides a solution to optimize the use of buffer and thus facilitating the establishment of a dedicated path from the source to the destination.

Keywords: *Circuit Switching, Bipartite Graph, Maximum Matching*

---***---

1. INTODUCTION

A circuit-switched network consists of a set of switches connected by physical links.

A connection between two stations is a dedicated path made of one or more links.

Three Phases
The actual communication in a circuit-switched network requires three phases: connection setup, data transfer, and connection teardown [4].

1.1 Circuit Switching

A circuit-switched network consists of a set of switches connected by physical links. A connection between two stations is a dedicated path made during the setup phase.
The actual communication in a circuit-switched network requires three phases: connection setup, data transfer, and connection teardown [3].

Before the two parties can communicate, a dedicated path needs to be established. The end systems are normally connected through dedicated path through the switches, so connection setup means creating dedicated channels between the switches. For example, in the given Fig. 1, when system A needs to send data to system M it forms the dedicated path from system A to system M by selecting the various switches that are required. This is called the 'Setup Phase'.

The conflict arises here when two systems require the same switch at the same instance to establish a dedicated path [2].
In the "Data Transfer Phase", once the establishment of the dedicated circuit (channels), the two parties can transfer data.

In the "Teardown Phase", when one of the parties needs to disconnect, the dedicated path is disconnected and the resources that were assigned to the systems are released [5].

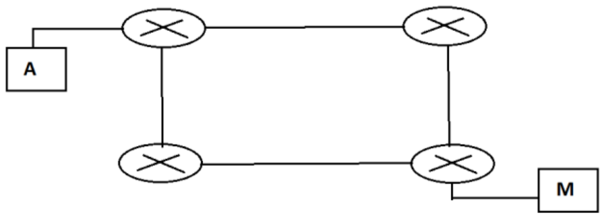

Fig 1. A circuit

1.2 Bipartite Graphs

A bipartite graph is a graph whose vertices can be divided into two disjoint sets U and V such that every edge connects a vertex in U to one in V. Vertex set U and V are usually called the components of the graph.

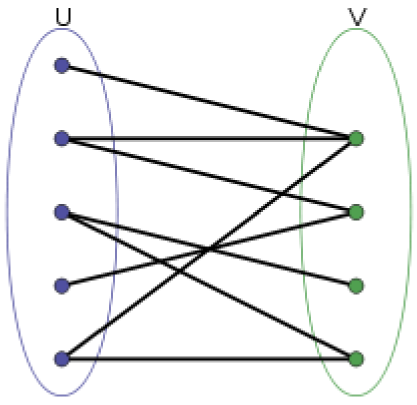

Fig 2. A Bipartite Graph

1.3 Maximum Matching

In bipartite graphs a matching is a set of pair wise disjoint edges. The Maximum matching algorithm aims at matching all elements of the set [1].

2. THE ALGORITHM

Details on matching theory and on the most popular algorithms can be found in [6].

The algorithm works on the basis of splitting the circuit into two disjoint sets, a bipartite graph G. The first disjoint set, Layer U comprises of systems requesting switches; and the second disjoint set, Layer V, comprises of systems which receive the immediate requests from the previous Layer U.

Every system is represented as a data structure "system", which stores three different values:

1. Name: Stores the name of the system in the field.
2. Hop: Stores the number of hops required to reach the destination.
3. Cumulative Hop: Store the cumulative hop values of all the systems requesting the particular system.
4. Requesting System: System that is currently active and requesting the current system.
5. Buffer Queue: Stores the requesting systems in a queue.

The graph G is a sub graph of the entire switching network. We deploy the maximum matching algorithm [1]. The matched graph is stored in M. M does not include the systems from Layer U that do not have an edge to the Layer V. These systems are assigned to the systems in Layer V based on the cumulative hop values of the systems that they request to.

In a scenario of a conflict when two systems have been assigned to the same switch, we call on the buffer() function which discerns which system shall transmit data first. This is decided on the hop value of the requesting system. The system with the least hop value is given priority and transmitted.

Once the two layers have been resolved, we move on the next immediate two layers of requesting and receiving systems.

2.1 Nomenclature

G	The bipartite Graph consisting of layers U and V
U	Layer U, consisting of requesting systems.
u_i	Object of set U
V	Layer V, consisting of systems receiving the requests.
v_j	Object of Set V
M	Graph comprising of maximum matched objects
A	Set of all objects from U not belonging to M
a_i	Object of set A
B	Set of objects from V that an object from A makes requests to.
b_j	Object of set B
X	Set of all objects from U requesting the object v_j
x_i	Object of set X
Y	Subset of set X of all the objects not in the buffer of the switch
y_i	Object of set Y
buf	Number of buffers a switch can hold
ctr	Counter
maximum_matching(G)	Function to find maximum matching in graph G
sort(A)	Function to sort objects in A in ascending order on the basis of HOP value
cumulative_sort(B)	Function to sort objects in ascending order on the basis of CUMULATIVE HOP value.
enqueue(y_i)	Putting the object y_i in a queue
dequeue(y_0)	Removing the object y_0 from the queue
Wait()	Function to wait for the data to be transmitted.

2.2 Pseudo Code

```
Procedure Match(G, U, V){
BEGINM=maximum_matching(G);
SET A={};
for each $v_j$ "belongs to" V
{
        for each $u_i$ belongs to M
        {
                if($u_i$ = $v_{j)}$
                {
                hop($v_j$) = hop($u_i$) – 1;
        cumulative_hop($v_j$) = hop($u_i$);
                }
        }
}

for each $u_i$ "belongs to" U
{               if   ($u_i$ does not belong to M)
         {
                A=A + $u_i$;
         }
}
sort(A);

 For each $a_i$ belongs to A
{
         SET B={};
         for each $v_j$ belongs to SET V
         {
                 if ($a_i$=v)$_j$
                 {
                         B=B + $v_j$;
                 }
         }
         cumulative_sort(B)
         $a_i$ = $b_0$;
    cumulative_hop($B_0$) = cumulative_hop($B_0$) + hop($A_i$);
}
For each $v_j$ belonging to SET V
{
        SET X={};
        For each $u_i$ belonging to SET U
        {
                if ($u_i$ equals to $v_j$)
                {
                        X = X union $u_i$
                }
        }
        Buffer(X,$v_j$)
}
END
}
```

```
Procedure Buffer(X, $v_j$)
{
BEGIN
buf = capacity of the buffer
ctr = 0;
sort(X)
Label 1:
$x_0 = v_j$
hop($v_j$) = hop($x_0$);
X = X – $x_0$
SET Y = set of all the objects in X not in the buffer
For each $Y_i$ belonging to Y
{
        if (ctr < buf)
        {
                enqueue($Y_i$);
                ctr = ctr + 1
        }
}

wait();
Cumulative_hop($v_j$) = Cumulative_Hop($v_j$) – Hop($x_0$);
$x_0$ = dequeue($y_0$);
ctr= ctr – 1
if (X is empty)
{
        wait();
        END
}
else
{
        Go to Label 1
}
}
```

---

3. WORKING EXAMPLE

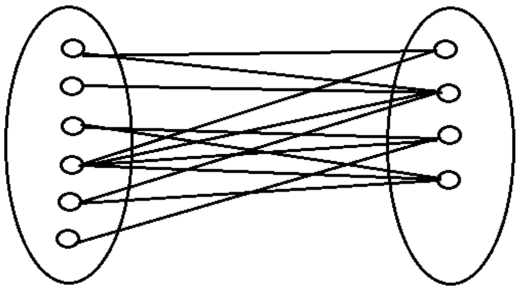

Fig. 3: An Example Circuit

The above Figure 3 gives us a circuit with Layer U comprising of six systems and requesting from Layer V, consisting of four systems.

The hop value of each system from Layer U is given in the Table 1.

Table -1: Systems of Layer U

System Name	Hop Value	Cumulative Hop Value	Requesting System
U_1	5	0	-
U_2	7	0	-
U_3	3	0	-
U_4	6	0	-
U_5	9	0	-
U_6	4	0	-

We call the function, maximum_matching(), to find the maximum matched graph M, as shown in Figure 4.

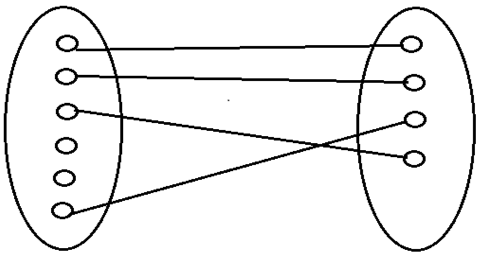

Fig. 4: Maximum matched graph M of network G

Once M is found, we update the Hop and Cumulative Hop values of systems in Layer V, as given in Table 2.

Table -2: Systems of Layer V

System Name	Hop Value	Cumulative Hop Value	Requesting System
V_1	4	4	U_1
V_2	6	6	U_2
V_3	3	3	U_6
V_4	2	2	U_3

Notice that system u_4 and u_5 are not included in the network. We must assign the two to either of their connections by comparing the Cumulative Hop value of the requested systems.

U_4 makes requests to all the systems in Layer V. The least Cumulative Hop Value belongs to v_3. Hence u_4 requests v_3. The updated Cumulative Hop Value of v_3 is now 8 (3+6-1).

Similarly, u_5 requests v_2 and v_4. The least Cumulative Hop Value of the two is of v_4. Hence, u_5 requests v_4 and the updated Cumulative Hop Value is 10 (2+9-1).

Figure 5 gives the simplified graph G'.
Table 3 gives the updated Layer V details.

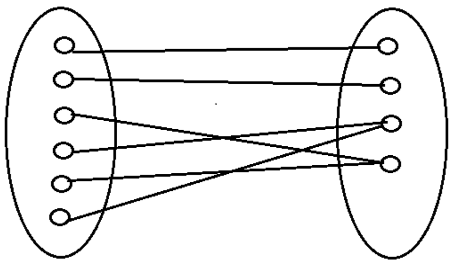

Fig 5. G' with the least connections between the layers

Table -3: Systems of Layer V

System Name	Hop Value	Cumulative Hop Value	Requesting System
V_1	4	4	U_1
V_2	6	6	U_2
V_3	3	8	U_6, U_4
V_4	2	10	U_3, U_5

There is a clear conflict in V_3 and V_4 as to which source should transmit data first. This is resolved by the function Buffer().

Buffer() first arranges the requesting objects with increasing Hop Values. Assigns the first requesting system to the v_j and the rest of the systems fill the buffer queue of vj.

V_3 has two requesting systems, u_6 and u_4. U_6 has the lowest hop value; hence data from this system gets transmitted first and u_4 gets stored in the queue.

Similarly, for V_4, U_3 gets transmitted first and u_5 is stored in the queue. Table 4 gives us the updated values.

Table -4: Systems of Layer V

System Name	Hop Value	Cumulative Hop Value	Requesting System	Buffer Queue
V_1	4	4	U_1	-
V_2	6	6	U_2	-
V_3	3	8	U_6	U_4
V_4	2	10	U_3	U_5

CONCLUSIONS

Resolving the resource allocation problem in circuit switching is really very important and as shown this can be resolved using graph theory concepts of bipartite graphs and maximum matching algorithms. The use of buffers as shown will facilitate the better allocation of resources.

REFERENCES

[1]. J. E. Hopcroft and R. M. Karp. An n5/2 algorithm for maximum matching in bipartite graphs. SIAM J. Comput., 2(4):225–231, 1973.

[2]. V. Benes, Mathematical Theory of Connecting Networks and Telephone Traffic, Academic Press, New York, 1965.

[3]. Circuit Switching: Unique Architecture and Applications Amos E. Joel, Jr., Computer, Volume12 Number6 (ISSNO018.9162), June 1979.

[4]. A Sampler of Circuit Switching Networks Gerald M Masson, George C Ginger, and Shinji Nakamura , Computer, Volume12 Number6 (ISSNO018.9162),June 1979.

[5]. Guest Editor's Introduction: Circuit Switching KennethJ. Thurber, Computer, Volume12 Number6 (ISSNO018.9162), June 1979.

[6]. B. Korte and J. Vygen, "Combinatorial optimization: theory and algorithms," 2005.

BIOGRAPHIES

Manthan D. Sanghavi, currently pursuing B. Tech. degree in the field of Computer Science and Engineering at VIT University, Vellore.

Shantan Sawa, currently pursuing B. Tech. degree in the field of Computer Science and Engineering at VIT University, Vellore.