Chapter 3

Gate Sizing and Threshold Voltage Assignment

Based on our motivation in Chapter 2, we propose a design flow to determine how to perform gate sizing and threshold voltage assignment. In Section 3.1, we first define the problem and show the design flow. In Section 3.2, the detailed algorithm will be presented.

3.1 Problem Definition and Design Flow

The problem can be defined as follows. Given a circuit, timing requirement, time profile of active and idle modes (i.e., the proportion of the total time that the circuit is in the active mode and in the idle mode) and cell library, minimize the power consumption (including leakage in idle mode and dynamic and leakage in active mode) by gate assignment including gate sizing and threshold voltage assignment.

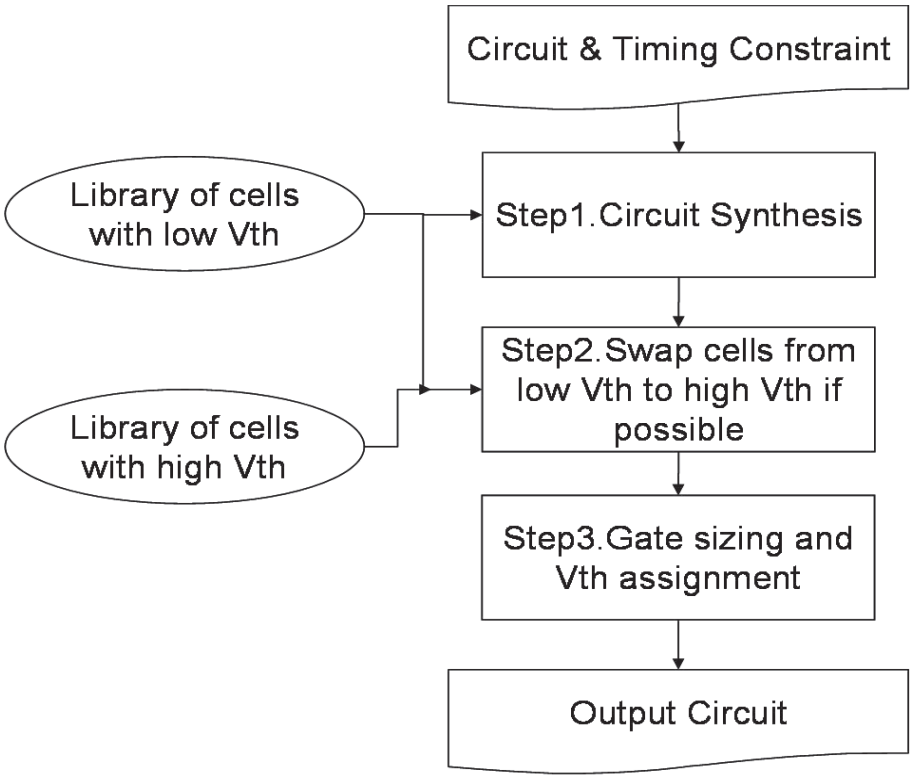


Figure 3.1: Design flow of the algorithm

To solve this problem, a design flow shown in Figure 3.1 is proposed. First, with the timing constraint, the circuit is synthesized using only cells with low Vth. The reason to use only cell with low Vth is that a circuit synthesized using cells with best timing (low threshold voltage) but minimal size allows gates to be sized up in the later optimization steps. Then, in the second step, all gates with low Vth are swapped to their corresponding high Vth cells. After this step, the timing constraint is no longer satisfied but the leakage is maximally reduced (the cell with low Vth has larger delay and less leakage). Finally, the last step is to perform gate sizing and threshold voltage re-assignment to restore the original timing performance. For nodes on critical path , our decision to choose gate for up-sizing or low threshold voltage assignment, will take the minimal increase in power consumption and area cost into consideration. For nodes on non-critical path, the algorithm utilize the slack of nodes to save more power consumption.

The first and the second steps are well understood. The detailed algorithm of the third step, **gate sizing and Vth assignment**, will be explained in the next subsection.

3.2 Algorithm for Gate Sizing and Threshold Voltage Assignment

Our third step, **gate sizing and Vth assignment**, is conducted in two phases. In the first phase, up-sizing or Vth re-assigning to low is performed on nodes on critical path and in the second phase, on the nodes on non-critical path. First, timing analysis on the circuit is performed. The arrival time, the required time and the slack of each gate are computed. Then, based on this timing information, a path balanced graph $G = (V, E)$ for the circuit is constructed. Next, separator sets of the graph are computed. The nodes in the separator sets are candidates for sizing or threshold voltage re-assignment. This step continues until the timing constraint is satisfied. Once the timing constraint is met, we continue to minimize the power consumption of circuit in the second phase by utilizing the remaining timing slack on non-critical paths.

The algorithm is depicted in Figure 3.2. The details are described in the following.

3.2.1 Critical Path

A circuit can be viewed as a directed graph $G = (V, E)$, as shown in Figure 3.3, where x, y, z in (x, y, z) denote *slack*, *delay-reduction*, and *cost*, of the nodes, respectively (*delay reduction*, and *cost* will be defined later). After

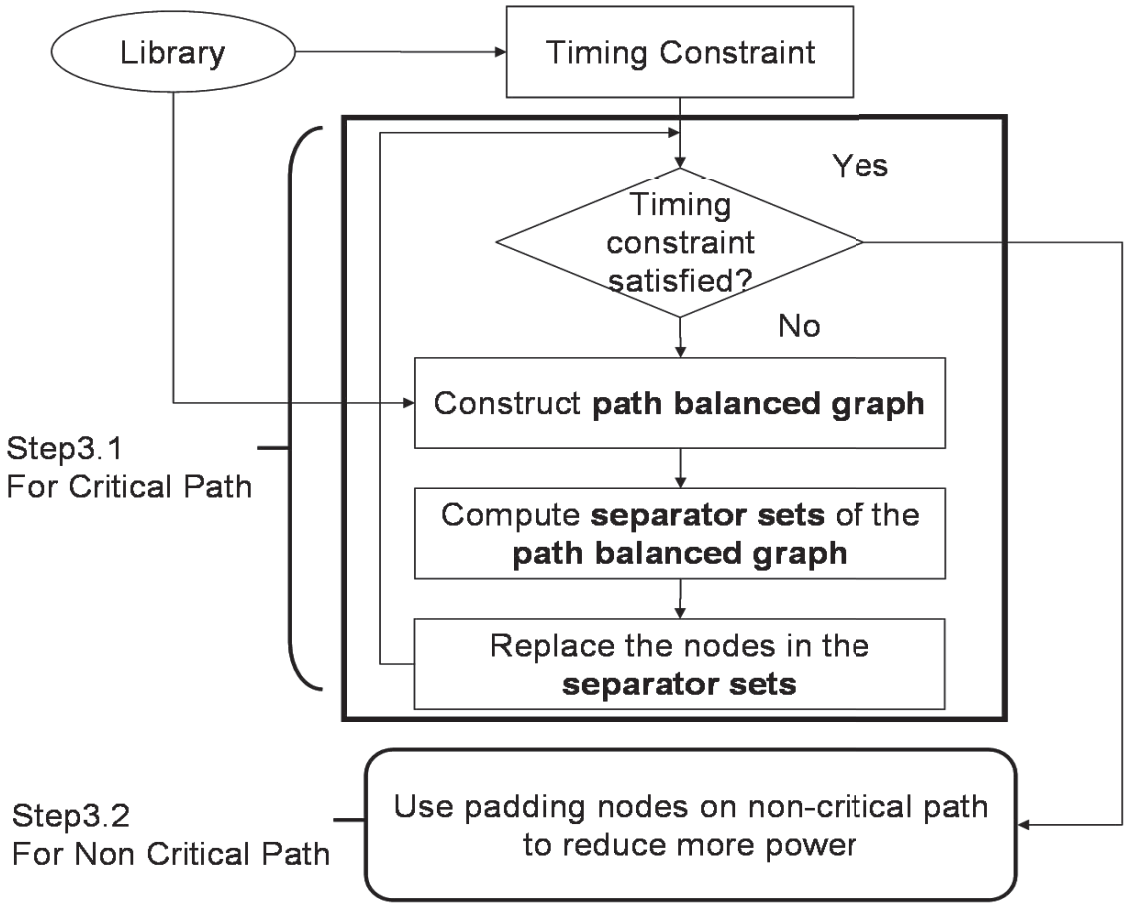


Figure 3.2: Step of **gate sizing and Vth assignment**

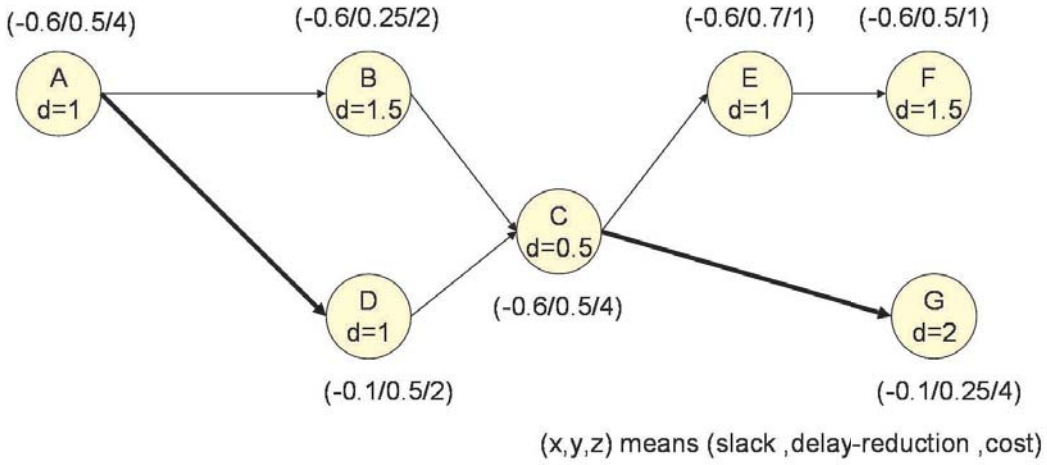


Figure 3.3: A circuit graph

timing analysis, the arrival, required times and slack of nodes are computed. Based on this timing information, to improve the timing performance, a set of nodes can be selected to speed up. Since there is usually more than one critical path, the selection step requires a lot of attention. The objective is usually to select a set of nodes with minimum cost.

One way to select nodes which can guarantee the circuit timing improvement is to select a separator set. However, simply selecting a separator set will not produce low cost result because slack on short path may not be fully utilized. Instead, we will select a separator set based on a *path balanced circuit graph* [7] in which slack of short paths can be fully exploited.

A *path balanced graph* is defined as follows [7]. First, for all edges e, $ds(e)$

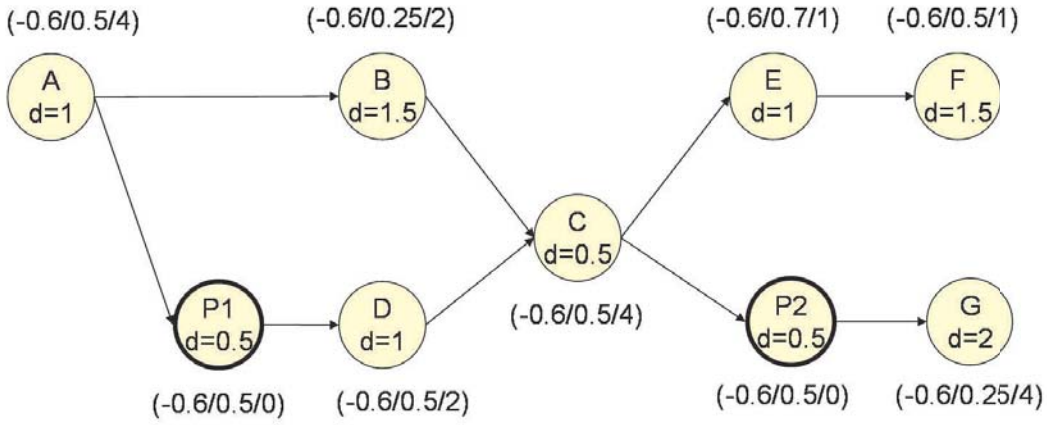


Figure 3.4: A *path balanced graph*

is computed. *ds(e)* is defined as the slack difference of nodes at the two ends of edge e. It is computed as

$$ds(e) = slack(tail\_node(e)) - slack(head\_node(e)) \tag{3.1}$$

If *ds(e)* $>$ *0*, it means that input is from a short path and there is timing slack on this path. A *padding node* is inserted at e whose delay is *ds(e)* and the cost of this node is *0*. By doing so, slack of all nodes become equal. When a separator set is to be selected, padding nodes are more likely to be selected. When the padding nodes are indeed selected in a separator for replacement in order to improve timing, the cost of replacing *padding nodes* is *0*. Figure 3.4 shows a *path balanced graph*, $G_{balanced} = (V, E)$ with padding nodes constructed from circuit graph, *G = (V, E)* of Figure 3.3. In this

figure, padding node *P1* is added between node A and node D because the slack difference of the edge $A \rightarrow D$ is *0.5* (*slack(node D) - slack(node A)*). Similarly padding node *P2* is added because the slack difference of the edge $C \rightarrow G$ is *0.5* (*slack(node G) - slack(node C)*).

Once the *path balanced graph* is constructed, we need to set the cost of a node. The cost is to be defined so that the less the cost of a node, the more probable the node is to be replaced for performance improvement. Before we present how to set the cost of a node, we need first to decide what the next candidate change is for a node to solve the timing violation problem. The objective of our algorithm is to select nodes with minimal power and area increase for replacement in order to meet the timing constraint. There are two ways to solve the timing violation: up-sizing gate or changing the gate with high Vth by that with low threshold voltage. However, either way will increase the total power (dynamic and leakage) in which up-sizing gate increases the load capacitance of fan-ins and hence dynamic power increases while assignment of low threshold voltage increases leakage power. To make a choice between these two options, based on the observation in Section 2, we should take the switching activity of gates into consideration. For gates with fan-ins of low switching activity, up-sizing should be selected because the increasing dynamic power of fan-ins may be very small even if gate size

is increased. On the other hand, for gates with fan-ins of high switching activity, assignment of low threshold voltage should be considered.

Based on this observation, we define a power penalty function, *penalty(g)* which is the penalty for the gate g if up-sizing or low Vth assignment is selected to replace the current gate g. It is calculated as

$$penalty(g) = \alpha \cdot p\_penalty(g) + \beta \cdot a\_penalty(g) \tag{3.2}$$

In this equation, *p_penalty* and *a_penalty* are the power and area increase overhead, respectively, and α and β are parameters to control the weights of power and area penalty. The $p\_penalty(g)$ is further defined as

$$p\_penalty(g) = a\_prop \cdot (\sum_{j \in fanin(g)} E(j) \cdot C_{inc}(g)V^2 + leak_{inc}(g)) + (1 - a\_prop) \cdot leak_{inc}(g) \tag{3.3}$$

where *a_prop* is the proportion of the total time that the circuit is in the active mode, *E (j)* is the switching activity of signal j, $C_{inc}(g)$ is the increased capacitance, V is the supply voltage, $leak_{inc}(g)$ is the increased leakage. The first term represents the power increase when the circuit runs in active mode and the second term the power increase when the circuit is in idle.

We compute the *penalty(g)* for the gate g for both up-sizing and low Vth assignment options. The option that has less penalty is selected as a candidate for replacement of the current gate g. Then, it is used to compute

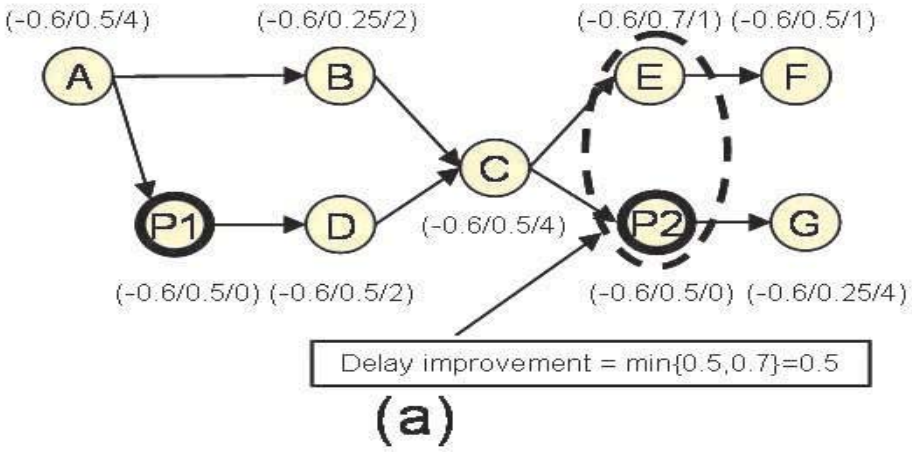
the *cost* of nodes in the *path-balanced graph*, $G_{balanced} = (V, E)$. Moreover, the delay reduction of the selected replacement is modelled as *delay-reduction* in the *path-balanced graph* .

Now, we show how to compute the *cost* of the *path-balanced graph* . The cost of a *padding node* is set to *0* and all other nodes g are computed as,

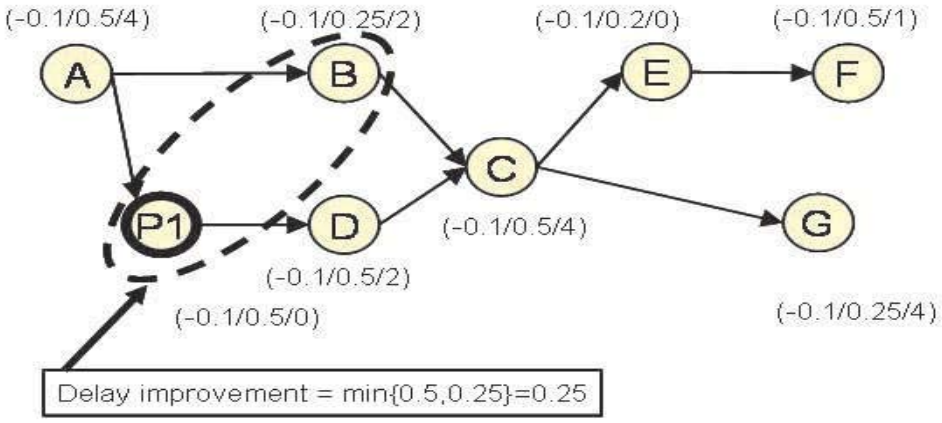
$$cost(g) = \gamma \cdot penalty(g) + \delta \cdot delay\_reduction(g) \tag{3.4}$$

where γ and δ are control parameters. Once the cost of the *path balanced graph* is computed, we will find a separator set of the graph. The nodes in the separator set are selected for replacement. Note that the *delay improvement* of this separator set, which is defined as the delay improvement of the circuit after the nodes of the separator set are replaced, is the minimum *delay-reduction* among the nodes in the separator set.

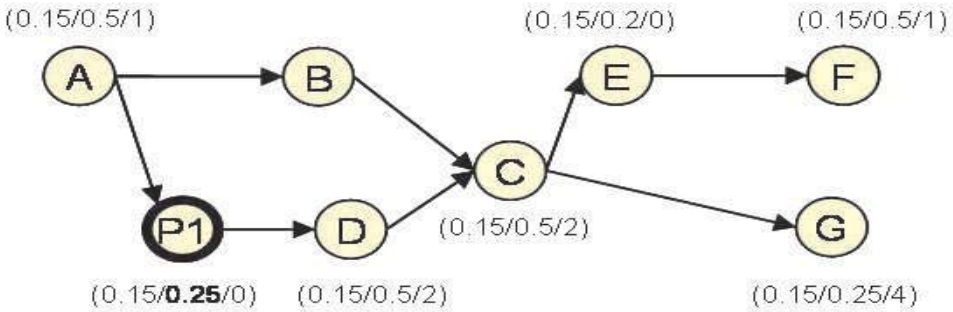
The next iteration will start with the timing analysis. If the timing constraint is not satisfied, the procedure continues. Also note during the reconstruction of *path balanced graph*, the *path balanced graph* is only partially modified. It proceeds as follows. First of all, *delay-reduction* of nodes in separator set are decreased by the *delay improvement*. For *padding nodes* with *0 delay-reduction*, they are removed from the graph. For a replaced cell, if the *delay reduction* is larger than the *delay improvement*, the *delay-reduction* of this *replaced cell* is decreased by the *delay improvement* of the circuit.



(a)



(b)



(c)

Figure 3.5: Process of the algorithm in Section 3.2.1
(a)A separator set of *path balanced graph* Figure 3.4 (b)A separator set of *path balanced graph*(a) (c)A final *path balanced graph* after replacement

Take the example shows in Figure 3.5 to demonstrate out selection algorithm. Figure 3.5(a) shows a separator set of *path balanced graph* of the original graph shown in Figure 3.4. $\{E,P2\}$ is selected to be the separator set with the minimal cost. The *delay improvement* of this set is *0.5*. The *delay-reduction* of node E and $P2$ are decreased by *0.5* as shown in Figure 3.5(b). Since the *delay-reduction* of node $P2$ equals to *0*, it is removed from the graph in Figure 3.5(b). In the next iteration, the separator $\{B,P1\}$ is selected as shown in Figure 3.5(b) and it results in *0.25* delay reduction. We can continue finding separator sets and update *path balanced graph* until timing constraint is met. The final *path balanced graph* is shown in Figure 3.5(c).

3.2.2 Non-Critical Path

After sizing and Vth re-assignment are performed on critical path, the timing constraint of circuit is met now. The objective of the next step is to utilize the remaining timing slack on *padding nodes* to reduce more power consumption. There are two ways to save the power consumption of a gate: down-sizing gate or changing the gate with high threshold voltage.

Recall that adding a *padding node* between node A and node B means that there is slack on the edge $A \rightarrow B$. Node A can be delayed ϵ time if every path going out from A has ϵ slack. In other words, if and only if all

```
1 Algorithm Step3.2()
2 {
3   /* Compute available slack */
4   For every node Ni
5     If all fanout nodes Nj are padding nodes then
6       Avai-slack(Ni) = min{ delay reduction of Nj}
7
8   /* Compute delay penalty caused by down-sizing or re-assigning Vth */
9   For every node Ni
10    Delay-penalty(Ni) = New-Delay(Ni) - Delay(Ni)
11
12  /* Compute power saving caused by down-sizing or re-assigning Vth */
13  For every node Ni
14    P_saving(Ni) =Power - New-Power(Ni)
15
16  /* Re-assign size or Re-assign Vth*/
17  For every node Ni
18    If Delay-penalty(Ni) > Avai-slack(Ni) then
19      Down-size Ni or Re-assign Vth of Ni to high based on P_saving
20}
```

Figure 3.6: Algorithm Step3.2. Utilize the padding nodes on non-critical path to save more power

the *fanout nodes* of node A are *padding nodes* with slack ϵ, node A can be delayed ϵ without affecting the timing of circuit. This maximum ϵ time of node A is denoted as the minimum number of *delay reduction* of *padding nodes*. Therefore, for every node *Ni* in the graph, we compute available slack of it. If all the outgoing edges of *Ni* end up with padding nodes, the available slack of *Ni* is computed as the minimal *delay-reduction* of *Nj*, *Nj* is the fanout node of *Ni*. Otherwise, the available slack of *Ni* is *0*.

Based on our motivation mentioned in Chapter 2, we know that if the switching activity is high, smaller gate with lower Vth should be selected while if the switching activity is low, high Vth with larger size gate is more power efficient. Therefore, we compute power saving *p_saving(g)* for the gate g for both down-sizing and re-assigning Vth. The $p\_saving(g)$ (*Power - New-Power(Ni)*) is further defined as

$$p\_saving(g) = a\_prop \cdot (\sum_{j \in fanin(g)} E(j) \cdot C_{dec}(g)V^2 + leak_{dec}(g)) + (1 - a\_prop) \cdot leak_{dec}(g) \quad (3.5)$$

where *a_prop* is the proportion of the total time that the circuit is in the active mode, *E (j)* is the switching activity of signal j, $C_{dec}(g)$ is the decreased capacitance, V is the supply voltage, $leak_{dec}(g)$ is the decreased leakage. The first term represent the power decrease when the circuit runs in active mode and the second term the power decrease when the circuit is in idle.

After calculating the available slack of all nodes, we compute the delay penalty which is caused by down-sizing nodes or re-assigning Vth to high. The *delay-penalty* of *Ni* is computed as *New-Delay(Ni) - Delay(Ni)*. *Delay(Ni)* is the node delay of *Ni* with current size and Vth, and *New-Delay(Ni)* is the node delay of *Ni* after down-sizing or re-assigning Vth to high. As to selecting sizing or high Vth assigning of a gate, the amount of power saving is used as a selecting criterion. When the delay penalty of both options is less than the available slack, whether to select down-sizing or re-assigning Vth to high is based on the power saving of these two options. The detailed algorithm is described in Figure 3.6.

Take the circuit in Figure 3.7 as an example. Let's take a look at node A. There are two paths starting from node A. Both of them end up with *padding nodes*, *P1* and *P2*. Therefore, available slack of node A is the minimal number of *delay reduction* of *P1* and *P2*, which is *0.4*. The delay penalty of node A *0.3*, is greater than the available slack of it. As a result, the size of node A can be reduced. On the contrary, there are two paths starting from node E, but only one ends in *padding node*. Hence, the available slack of node E is *0*, and node E can't be delayed without affecting the timing of circuit.

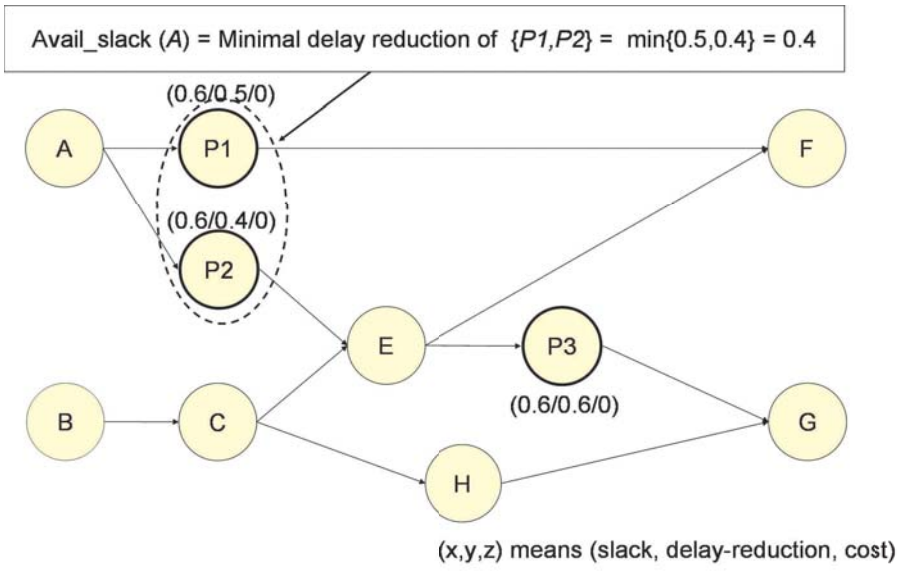


Figure 3.7: A modified *path balanced graph* after Section3.2.1 . Delay-penalty(A) = *0.3*, Delay-penalty(E) = *0.2*