# S-38.220
# Postgraduate Course on Signal Processing in Communications,
# FALL - 99

## Topic: Iteration Bound


## Harri Mäntylä

harri.mantyla@hut.fi


Date: 11.10.1999

# 1. INTRODUCTION

Digital signal processing algorithms are often recursive in nature. The loops or cycles that are formed in recursive and adaptive algorithms represent a fundamental limit to the computation speed. Once this limit, also called iteration period bound or iteration bound, is reached the addition of more computing power has no effect on the resulting speed. The iteration bound is fundamental to the used algorithm and is independent of the implementation architecture. The approximation of the iteration bound is often the goal of an implementation.

DSP algorithm has to be mapped onto a data-flow graph (DFG) presentation in order to be able to calculate the iteration bound. DFG models the way computational operations depend on data produced by other operations. DFGs are formed of directed edges, modeling the precedence constraints among the nodes, and nodes, which accept input data from their predecessors, execute some tasks and output data to the next node.

In this presentation the iteration bound is defined and two techniques are presented for computing the iteration bound. Also the iteration bound of multirate DFGs is discussed.

## 2. DATA-FLOW GRAPH (DFG) REPRESENTATIONS

DSP algorithm can be presented as a DFG as shown in Figure 1. In the figure node A represents addition with execution time of 2 u.t. and node B represents multiplication with execution time of 4 u.t. The edge A→ B has zero delays and the edge B→ A has one delay. Execution of all nodes of the DFG exactly once completes an iteration of the DFG. The precedence constraints between two nodes are defined by edges of the DFG. These constraints specify the order of the node execution in the DFG. The critical path is defined as the path with the longest computation time among all paths containing zero delays. The critical path represents the minimum computation time for one iteration of the DFG.

A DFG may be recursive or nonrecursive. Nonrecursive DFG contains no loops and can be executed arbitrarily fast, if the required computing power is available and increased parallelism is applied. Recursive DFG presents a fundamental limit on computation speed. This limit is called the iteration bound, $T_\infty$.

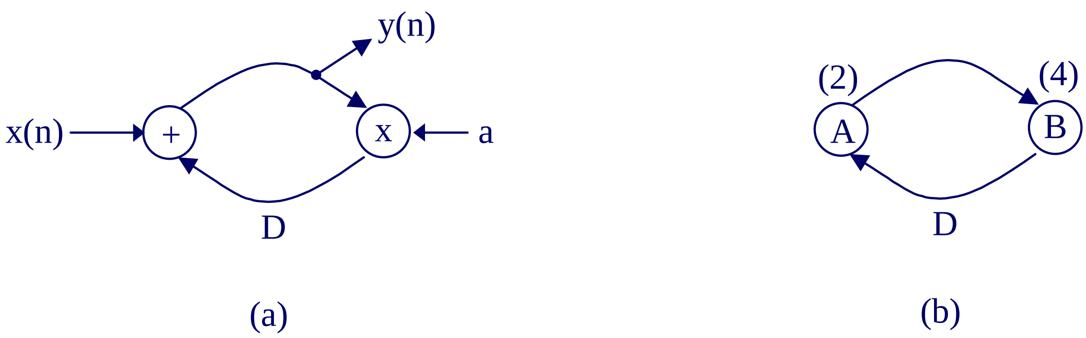


*Figure 1: y(n)=ay(n-1)+x(n) presented (a) graphically and (b) as a DFG.*

## 3. LOOP BOUND AND ITERATION BOUND

A loop or a cycle is a directed path in a DFG beginning and ending at the same node. Loop bound represents the lower bound on the loop computation time. Loop bound of the *l*-th loop is defined as $t_l / w_l$, where $t_l$ is the loop computation time and $w_l$ is the number of delays in the loop.

The loop with the maximum loop bound is called the critical loop. The loop bound of the critical loop is called the iteration bound of the DSP algorithm.

The iteration bound is defined in Equation ( 1).

$$T_{\infty} = \max_{l \in L} \left\{ \frac{t_l}{w_l} \right\} \qquad (1)$$

where $L$ is the set of loops in the DFG.

## 4. ALGORITHMS FOR COMPUTING ITERATION BOUND

The iteration bound of a DFG can be found by calculating the loop bound for all loops of the DFG as was shown in Equation 1. This straightforward technique may require long execution times as the number of loops in a DFG can be exponential with respect to the number of nodes. In the following two techniques providing polynomial execution times for finding the iteration bound are presented. These techniques are the longest path matrix algorithm (LPM) and the minimum cycle mean algorithm (MCM).

### 4.1 Longest Path Matrix (LPM) Algorithm

In the LPM algorithm a series a matrices is constructed and the iteration bound is found by examining the diagonal elements of the matrices.

Constructing of the matrix:

$d$ = number of delays in the DFG
$\mathbf{L}^{(m)}$ = matrix $m$ , where $m$ = 1, 2, …, $d$
$l_{i,j}^{(m)}$ = element of matrix $\mathbf{L}^{(m)}$
= longest computation time of all paths from delay element $d_i$ to delay element $d_j$ that pass through exactly $m$ – 1 delays, if no such path exists then value –1 is used.

Matrix $\mathbf{L}^{(1)}$ must be constructed in the above described way. In computing the longest path a path algorithm needs to be implemented. For example the Bellman-Ford algorithm or the Floyd-Warshall algorithm could be used.

The higher order matrices, from $\mathbf{L}^{(2)}$ to $\mathbf{L}^{(d)}$, can be recursively calculated from lower order matrices according to the formula in Equation ( 2 ).

$$l_{i,j}^{(m+1)} = \max_{k \in K}\left(-1, l_{i,k}^{(1)} + l_{k,j}^{(m)}\right) \qquad (2)$$

where $K$ is the set of integers $k$ in the interval [1,$d$] such that neither $l_{i,k}^{(1)} = -1$ nor $l_{k,j}^{(m)} = -1$ holds.

After computing matrices $\mathbf{L}^{(m)}$ the iteration bound is:

$$T_\infty = \max_{i,m \in \{1,2,\ldots,d\}} \left\{ \frac{l_{i,i}^{(m)}}{m} \right\} \qquad (3)$$

In the LPM algorithm the value $l_{i,i}^{(m)}$ represents the longest computation time of all loops having $m$ delays and containing the delay element $d_i$. The maximum value of all possible values presented in Equation ( 3 ), represents the maximum loop bound that is the iteration bound of the algorithm in question.

## 4.2 The Minimum Cycle Mean (MCM) Algorithm

The MCM algorithm reduces the calculation of the iteration bound to finding the MCM of a graph. The algorithm for computing the MCM that is described here uses the concepts of a cycle mean, the maximum cycle mean and the MCM.

The cycle mean *m(c)* of a cycle *c* is the average length of the edges in *c*. This can be calculated by dividing the sum of edge lengths by the number of edges in the cycle. The MCM $\lambda_{min}$ is the minimum value of all cycle means and the maximum cycle mean $\lambda_{max}$ is the maximum value of all cycle means.

A new graph $G_d$ needs to be formed of the DFG for which we are computing the iteration bound. Graph $G_d$ has *d* nodes corresponding to each of the delays in the DFG. The weight *w(i,j)* of the edge $i \rightarrow j$ in $G_d$ is the longest path length in the original DFG from the delay $d_i$ to the delay $d_j$ containing no delays. If no such path exist then that edge does not exist in $G_d$.

The iteration bound of the DFG equals the maximum cycle mean of the $G_d$:

$$T_\infty = \lambda_{\max, G_d} = \max_c \{m(c)\} \quad (4)$$

Efficient algorithms have been developed for finding the MCM ($\lambda_{min}$)of a graph. To be able to utilize one of these MCM algorithms $G_d$ must be converted to $\overline{G}_d$ by multiplying all edges of $G_d$ by –1:

$$\overline{G}_d = -1 \times G_d \quad (5)$$

From $\overline{G}_d$ the iteration bound can be found by using MCM algorithm:

$$\lambda_{\min, \overline{G}_d} = -1 \times \lambda_{\max, G_d} = -1 \times T_\infty \quad (6)$$

Thus the iteration bound is:

$$T_\infty = -1 \times \lambda_{\min, \overline{G}_d} \quad (7)$$

## 5. EXAMPLE

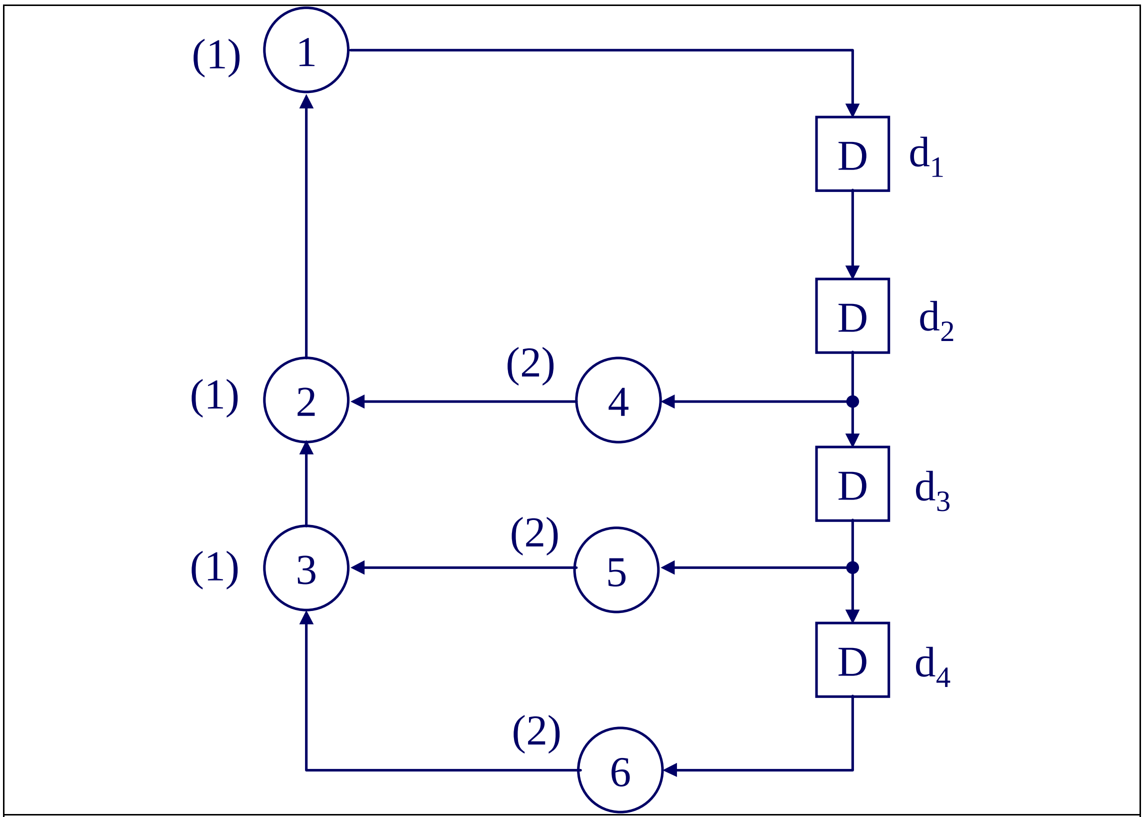


*Figure 2:DFG with three loops.*