A NEW EQUATION FOR THE LOAD BALANCE SCHEDULING BASED ON THE SMARANDACHE *F*-INFERIOR PART FUNCTION

Tatiana Tabirca[*]   Sabin Tabirca[**]

* University of Manchester, Department of Computer Science
** University College Cork, Department of Computer Science

Abstract. This article represents an extension of [Tabirca, 2000a]. A new equation for upper bounds is obtained based on the Smarandache f-inferior part function. An example involving upper diagonal matrices is given in order to illustrate that the new equation provide a better computation.

1.INTRODUCTION

Loop imbalance is the most important overhead in many parallel applications. Because loop structures represents the main source of parallelism, the scheduling of parallel loop iterations to processors can determine its decreasing. Among the many method for loop scheduling, the *load balance scheduling* is a recent one and was proposed by Bull [1998] and developed by Freeman *et.al.* [1999, 2000]. Tabirca [2000] studied this method and proposed an equation for the case when the work is distributed to all the processors.

Consider that there are p processors denoted in the following by $P_1, P_2, ..., P_p$ and a single parallel loop (see Figure 1.).

do parallel i=1,n
      call loop_body(i);
end do

Figure 1. Single Parallel Loop

We also assume that the work of the routine loop_body(i) can be evaluated and is given by the function $w: N \to R$, where $w(i) = w_i$ represents the number of routine's operations or its running time (assume that $w(0)=0$). The total amount of work for the parallel loop is $\sum_{i=1}^{n} w(i)$. The efficient loop-scheduling algorithm distributes equally this total amount of

work on processors such that a processor receives a quantity of work equal to $\frac{1}{p}\cdot\sum_{i=1}^{n}w(i)$.

Let l_j and h_j be the lower and upper bounds, $j=1,2,...,p$, such that processor j executes all the iterations between l_j and h_j. These bounds are found distributing equally the work on processors by using

$$\sum_{i=l_j}^{h_j}w(i)\approx\frac{1}{p}\cdot\sum_{i=1}^{n}w(i)\ \left(\forall j=1,2,...,p\right).\tag{1}$$

Moreover, they satisfy the following equations

$$l_1=1.\tag{2.a}$$

if we know l_j, then h_j is given by $\sum_{i=l_j}^{h_j}w(i)\approx\frac{1}{p}\cdot\sum_{i=1}^{n}w(i)=\overline{W}$.
$$\tag{2.b}$$

$$l_{j+1}=h_j+1.\tag{2.c}$$

Suppose that Equation (2.b) is computed by a less approximation. This means that if we have the value l_j, then we find h_j as follows:

$$h_j=h\ \Leftrightarrow\ \sum_{i=l_j}^{h}w(i)\le\overline{W}<\sum_{i=l_j}^{h+1}w(i)\ .\tag{3}$$

The Smarandache f-inferior part function represents a generalisation of the inferior part function $[,]:R\to Z$, $[x]=k\Leftrightarrow k\le x<k+1$. If $f:Z\to R$ is a strict increasing function that satisfies $\lim_{n\to-\infty}f(n)=-\infty$ and $\lim_{n\to\infty}f(n)=\infty$, then the Smarandache f-inferior part function denoted by $f_{[]}:R\to Z$ is defined by [see www.gallup.unm.edu/~smarandache]

$$f_{[]}(x)=k\Leftrightarrow f(k)\le x<f(k+1).\tag{4}$$

Tabirca [2000a] presented some Smarandache f-inferior part functions for which $f(k)=\sum_{i=1}^{k}i^a$. They are presented in the following:

$$f(k)=\sum_{i=1}^{k}i\ \Rightarrow\ f_{[]}(x)=\left[\frac{-1+\sqrt{1+8\cdot x}}{2}\right]\forall x\ge0.\tag{5}$$

$$f(k)=\sum_{i=1}^{k}i^2\ \Rightarrow\ f_{[]}(x)=\left[r(x)\right]\forall x\ge0,\tag{6}$$

where $r(x)=-\frac{1}{2}+\sqrt[3]{\frac{3\cdot x}{2}-\sqrt{\left(\frac{3\cdot x}{2}\right)^2+\frac{1}{1728}}}+\sqrt[3]{\frac{3\cdot x}{2}+\sqrt{\left(\frac{3\cdot x}{2}\right)^2+\frac{1}{1728}}}$.

Tabirca [2000] also proposed an equation for the upper bounds of the load balance scheduling method based on the Smarandache f-inferior part function. If the work w satisfies certain conditions [Tabirca, 2000], then the upper bounds are given by

$$h_j^{(1)} = f_{\mathbb{I}}\left(j \cdot \overline{W}\right), j = 1,2,...,p.$$ (7)

Moreover, Tabirca [2000a] applied this method to the product between an upper diagonal matrix and a vector. It was proved that the load balance scheduling method offers the lowest running time in comparison with other static scheduling methods [Tabirca, 2000b].

2. A NEW EQUATION FOR THE UPPER BOUNDS

In this section, a new equation for the upper bounds is introduced. Some theoretical considerations about the new equation and Equation (7) are also made. Consider that

$f : N \rightarrow R$ is defined by $f(k) = \sum_{i=1}^{k} w_i$, $f(0) = 0$. For the work w, we assume the following [Tabirca, 2000]:

A1: $w_j \leq \dfrac{1}{p} \cdot \sum_{i=1}^{n} w_i, j = 1,2,...,n.$

A2: There are equations for the functions $f, f_{\mathbb{I}}$.

Theorem 1. *The upper bounds of the load balance scheduling method are given by*

$$h_j^{(2)} = f_{\mathbb{I}}\left(f(h_{j-1}^{(2)}) + \overline{W}\right), j = 1,2,...,p.$$ (8)

Proof. For easiness we denote in the following $h_j = h_j^{(2)}$. Equation (3) gives the upper bounds of the load balance scheduling method. We start from the equation

$\sum_{i=l_j}^{h_j} w(i) \leq \overline{W} < \sum_{i=l_j}^{h_j+1} w(i)$ and add $f(h_{j-1}) = \sum_{i=1}^{k_{j-1}} w_i$ to all the sides

$$\sum_{i=1}^{h_j} w(i) \leq f(h_{j-1}) + \overline{W} < \sum_{i=1}^{h_j+1} w(i).$$

Based on the definition of $f_{\mathbb{I}}$, we find that $h_j = f_{\mathbb{I}}\left(f(h_{j-1}) + \overline{W}\right).$ ♦

The following theorem illustrates how these bounds are.

Theorem 2. $h_j^{(2)} \leq h_j^{(1)}, j = 1,2,...,p.$

Proof. Recall that these two upper bounds satisfy

$$\sum_{i=1}^{h_j^{(1)}} w_i \leq j \cdot \overline{W} < \sum_{i=1}^{h_j^{(1)}+1} w_i$$ (9.a)

$$\sum_{i=l_j^{(2)}}^{h_j^{(2)}} w_i \leq \overline{W} < \sum_{i=l_j^{(2)}}^{h_j^{(2)}+1} w_i .$$ (9.b)

All the sums from Equation (9.b.) are added finding

$$\sum_{i=1}^{j} \sum_{k=l_i^{(2)}}^{h_i^{(2)}} w_k \leq j \cdot \overline{W} \Leftrightarrow \sum_{i=1}^{h_j^{(2)}} w_i \leq j \cdot \overline{W} .$$

Because $h_j^{(1)}$ is the last index satisfying Equation (9.a) we find that $h_j^{(2)} \leq h_j^{(1)}$ holds.   ◆

Consequence: $f(h_j^{(2)}) \leq f(h_j^{(1)}) \leq j \cdot \overline{W}, j = 1,2,...,p$.

This consequence obviously comes from the monotony of f and the definition of the bounds.

Now, we have two equations for the upper bounds of the load balance scheduling method. Equation (8) was obtained naturally by starting from the definition of the load balance. It reflects that case when several load balances are performed consecutively. Equation (7) was found by considering the last partial sum that is under $j \cdot \overline{W}$. This option does not consider any load balance such that we expect it to be not quit efficient. Moreover, it is difficult to predict which equation is the best or is better to use it of a given computation. The best practical advice is to apply both of them and to choose the one, which gives the lowest times.

3. COMPUTATIONAL RESULTS

In this section we present an example for the load balance scheduling method. This example deals with the product between an upper diagonal matrix and a vector [Jaja, 1992]. All the computations have been performed on SGI Power Challenge 2000 parallel machine with 16 processors. The dimension of the matrix was $n=300$.

DO PARALLEL i=1,n
$\qquad y_i = a_{i,1} \cdot x_1$
$\qquad$ **DO** j=2,i
$\qquad\qquad y_i = y_i + a_{i,j} \cdot x_j$
$\qquad$ **END DO**
END DO

Figure 2. Parallel Computation for the Upper Matrix – Vector Product.

Recall that $a = (a_{i,j})_{i,j=\overline{1,n}} \in M_n(R)$ is upper diagonal if $a_{i,j} = 0, i < j$. The product $y = a \cdot x$ between an upper diagonal matrix $a = (a_{i,j})_{i,j=\overline{1,n}} \in M_n(R)$ and a vector $x \in R^n$ is given by

$$y_i = \sum_{j=1}^{i} a_{i,j} \cdot x_j \quad \forall i = 1,2,...n .$$ (10)

The parallel computation of Equation (10) is shown in Figure 2.

The work of iteration i is given by $w(i) = i, i = 1,2,...,n$. We have that the total work is

$$f(n) = \sum_{i=1}^{n} i = \frac{n \cdot (n+1)}{2} \quad \text{and} \quad \overline{W} = \frac{n \cdot (n+1)}{2 \cdot p}.$$ The Smarandache f-inferior function is

$$f_{\square}(x) = \left[\frac{-1 + \sqrt{1+8 \cdot x}}{2} \right] \forall x \geq 0.$$ Therefore, the upper bounds of the load balance scheduling method are given by

$$h_j^{(1)} = \left[\frac{-1 + \sqrt{1 + 4 \cdot j \cdot \dfrac{n \cdot (n+1)}{p}}}{2} \right], j = 1,2,...,p \qquad \text{or} \qquad (11)$$

$$h_j^{(2)} = \left[\frac{-1 + \sqrt{1 + 4 \cdot h_{j-1}^{(2)} \cdot (h_{j-1}^{(2)} + 1) + 4 \cdot \dfrac{n \cdot (n+1)}{p}}}{2} \right], j = 1,2,...,p. \qquad (12)$$

The running times for these two types of upper bounds are presented in Table 1. Figure 3 proves that these two types of bounds for the load balance scheduling are comparable the same.

	P=1	P=2	P=3	P=6	P=8
$h_j^{(1)}$	1.847	1.347	0.987	0.750	0.482
$h_j^{(2)}$	1.842	1.258	0.832	0.639	0.412

Table 1. Times of the computation.

4. FINAL CONCLUSSION

An important remark that can be outlined is the Smarandache inferior part function was applied successfully to solve an important scheduling problem. Based on it, two equations for the upper bounds of the load balance scheduling methods have been found. These equations have been used to solve the product between an upper diagonal matrix and vector and the computational times were quite similar. The upper bounds given by the new equation have provided a better computation for this problem.
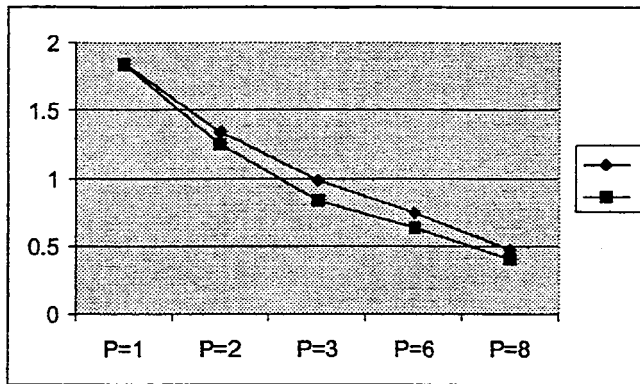
28

Figure 3. Graphics of the Running Times.

REFERENCES

Bull M.J. (1998) Feedback Guided Loop Scheduling: Algorithm and Experiments, Proceedings of Euro-Par'98, Lecture Notes in Computer Science, Springer Verlang.

Bull M.J., R.W.Ford, T.L.Freeman and D.J.Hancock (1999) A Theoretical Investigation of Feedback Guided Loop Scheduling, Proceedings of Ninth SIAM Conference on Parallel Processing for Scientific Computing, SIAM Press.

J.M.Bull and T.L.Freeman (2000) Convergence of Feedback Guided Loop Scheduling, Technical Report, Department of Computer Science, University of Manchester.

Jaja, J. (1992) *Introduction to Parallel Algorithms*, Adison-Wesley.

Tabirca, T. and S. Tabirca, (2000) Balanced Work Loop-Scheduling, *Technical Report*, Department of Computer Science, Manchester University. [*In Preparation*]

Tabirca, T. and S. Tabirca, (2000a) A Parallel Loop Scheduling Algorithm Based on The Smarandache *f*-Inferior Part Function, *Smarandache Notions Journal* [to appear].