Fast diffusion load balancing algorithms on torus graphs

Gregory Karagiorgos, Nikolaos M. Missirlis and Filippos Tzaferis
Department of Informatics and Telecommunications
Panepistimiopolis 157 84, Athens, Greece,
{greg,nmis,ftzaf}@di.uoa.gr

National and Kapodistrian University of Athens

Abstract. In this paper, we consider the application of accelerated methods in order to increase the rate of convergence of the diffusive iterative load balancing algorithms. In particular, we compare the application of Semi-Iterative, Second Degree and Variable Extrapolation techniques on the basic Diffusion method and the Extrapolated Diffusion method for torus graphs. It is shown that our methods require approximately 30% less iterations to reach the balanced state compared to the existed ones.
Keywords: Iterative load balancing; Diffusion algorithms; Distributed processor network; accelerated techniques

1 Introduction

We consider the following abstract distributed load balancing problem. We are given an arbitrary, undirected, connected graph $G = (V, E)$ in which node $v_i \in V$ contains a number u_i of current workload. The goal is to determine a schedule to move an amount of workload across edges so that, the weight on each node is equal. Communication between non-adjacent nodes is not allowed. This problem describes load balancing in synchronous distributed processor networks and parallel machines when we associate a node with a processor, an edge with a communication link of unbounded capacity between two processors, and the weight as infinitly divisible independent tasks. Diffusion algorithms assume that a node of the graph is able to send and receive messages to/from all its neighbours simultaneously.

The performance of a balancing algorithm can be measured in terms of number of iterations to reach a balanced state and the amount of load moved over the edge of the graph. The original algorithm described by Cybenko [2] and, independently, by Boillat [1] lacks in performance because of its very slow convergence to the balanced state [14]. Recently, diffusive algorithms have been proposed [3, 6, 8, 10, 12] to speed up the iteration process by an order of magnitude. All commonly used diffusion schemes generate the unique l_2-minimal flow [3, 4, 8]. Most of the existing iterative dynamic load balancing algorithms [3, 8, 14] involve two steps:

– Flow calculation: Calculating the amount of workload to be migrated between neighbouring processors such that a uniform load distribution is achieved when the migration is carried out to satisfy the flow.
– Task selection: Deciding which particular tasks are to be migrated, and scheduling these tasks to the appropriate neighbouring processors.

In practice, the diffusion iteration is used for the flow calculation. The real movement of task is complex and the exact details depend on the applications and the data structures used. This paper is concerned with algorithms for the first step. We consider the application of accelerated methods in order to increase the rate of convergence of the diffusive iterative load balancing algorithms for torus graphs. In particular, we apply Semi-Iterative (SI), Second Degree (SD) and Variable Extrapolation (VE) techniques [13, 15] on the Extrapolated Diffusion (EDF) method [11] and compare their performances with the Diffusion (DF) method. It is shown that our methods require approximately 30% less iterations to reach the balanced state compared to the existed ones [4, 6, 8].

The paper is organized as follows. Section 2 presents the extrapolated diffusion method. Section 3 adapts the aforementioned accelerated techniques to the extrapolated diffusion method. These methods increase the rate of convergence of the basic iterative scheme by an order of magnitude. Section 4 presents our results and conclusions.

2 The extrapolated diffusion method

The Extrapolated Diffusion (EDF) method for the load balancing has the form [1, 2]

$$u_i^{(n+1)} = u_i^{(n)} - \tau \sum_{j \in A(i)} c_{ij} \left(u_i^{(n)} - u_j^{(n)} \right), \tag{1}$$

where c_{ij} are diffusion parameters, $A(i)$ is the set of the nearest neighbors of node i of the graph $G = (V, E)$, $u_i^{(n)}$, $i = 0, 1, 2, \ldots, |V|$ is the load after the n-th iteration on node i and $\tau \in \mathbb{R} \backslash \{0\}$ is a parameter that plays an important role in the convergence of the whole system to the equilibrium state. The overall workload distribution at step n, denoted by $u^{(n)}$, is the transpose of the vector $(u_1^{(n)}, u_2^{(n)}, \ldots, u_{|V|}^{(n)})$ and $u^{(0)}$ is the initial workload distribution. In matrix form (1) becomes

$$u^{(n+1)} = M u^{(n)}, \tag{2}$$

where M is called the *diffusion matrix*. The elements of M, m_{ij}, are equal to τc_{ij}, if $j \in A(i)$, $1 - \tau \sum_{j \in A(i)} c_{ij}$, if $i = j$ and 0 otherwise. With this formulation, the features of diffusive load balancing are fully captured by the iterative process (2) governed by the diffusion matrix M. Also, (2) can be written as $u^{(n+1)} = (I - \tau L) u^{(n)}$, where $L = BWB^T$ is the *weighted Laplacian* matrix of the graph, W is a diagonal matrix of size $|E| \times |E|$ consisting of the coefficients c_{ij} and B is the vertex-edge incident matrix. At this point, we note that if $\tau = 1$, then we obtain the DF method proposed by Cybenko [2] and Boillat [1],

independently. If $W = I$, then we obtain the special case of the DF method with a single parameter τ *(unweighted Laplacian)*. In the unweighted case and for network topologies such as chain, 2D-mesh, nD-mesh, ring, 2D-torus, nD-torus and nD-hypercube, optimal values for the parameter τ that maximize the convergence rate have been derived by Xu and Lau [14]. Next, we consider the weighted case.
The diffusion matrix of EDF can be written as

$$M = I - \tau L, \qquad L = D - A, \tag{3}$$

where $D = diag(L)$ and A is the weighted adjacency matrix. Because of (3), (2) becomes $u^{(n+1)} = (I - \tau D)\, u^{(n)} + \tau A u^{(n)}$ or in component form

$$u_i^{(n+1)} = \left(1 - \tau \sum_{j \in A(i)} c_{ij}\right) u_i^{(n)} + \tau \sum_{j \in A(i)} c_{ij} u_j^{(n)}, i = 1, 2, \ldots, |V|. \tag{4}$$

The diffusion matrix M must have the following properties: nonnegative, symmetric and stochastic [2, 1]. The eigenvalues of L are $0 = \lambda_1 < \lambda_2 \leq \ldots \leq \lambda_n$. In case $c_{ij} =$ constant, the optimum value of τ is attained at [13, 15]

$$\tau_o = \frac{2}{\lambda_2 + \lambda_n} \tag{5}$$

and the corresponding minimum value of the convergence factor

$$\gamma(M) = \max\{|1 - \tau\lambda_n|, |1 - \tau\lambda_2|\} \tag{6}$$

is given by

$$\gamma_o(M) = \frac{P(L) - 1}{P(L) + 1}, \text{ where } P(L) = \frac{\lambda_n}{\lambda_2}, \tag{7}$$

which is the P-condition number of L. Note that if $P(L) \gg 1$, then the rate of convergence of the DF method is given by

$$R(M) = -\log \gamma_o(M) \simeq \frac{2}{P(L)}, \tag{8}$$

which implies that the rate of convergence of the DF method is a decreasing function of $P(L)$. In the sequel, we will express the optimum values of the parameters involved, in each considered iterative scheme, using the second minimum and maximum eigenvalues λ_2, λ_n, respectively of the Laplacian matrix. A first advantage of EDF is that it converges for any positive, real values of the parameters c_{ij} if $\tau \in (0, 1/||A||_\infty)$ [11], whereas in DF it is required that c_{ij} must satisfy the conditions $\sum_{j \in A(i)} c_{ij} < 1$ for at least one i. The problem of determining the diffusion parameters c_{ij} such that EDF attains its maximum rate of convergence is an active research area [3, 5, 11]. Introducing the set of parameters $\tau_i, i = 1, 2, \ldots, |V|$, instead of a fixed parameter τ in 4, the problem moves to the determination of the parameters τ_i in terms of c_{ij}. By considering local Fourier analysis [9, 11] we were able to determine good values (near

the optimum) for τ_i. These values become optimum in case the diffusion parameters are constant in each dimension and satisfy the relation $c_j^{(2)} = \sigma_2 c_i^{(1)}$, $i = 1, 2, \ldots, N_1$, $j = 1, 2, \ldots, N_2$, where $\sigma_2 = \frac{1-\cos\frac{2\pi}{N_1}}{1-\cos\frac{2\pi}{N_2}}$ and $c_i^{(1)}$, $c_j^{(2)}$ are the row and column diffusion parameters, respectively, of the torus (see Fig. 1).

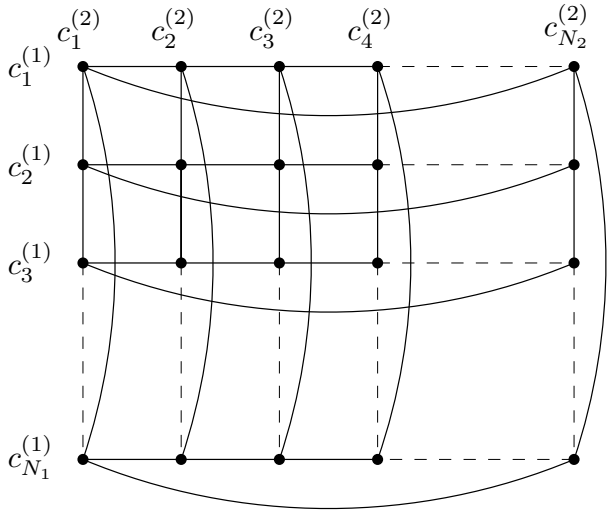

Fig. 1. The diffusion parameters in a 2-D torus

At the optimum stage EDF is twice as fast as DF for stretched torus, that is a torus with either $N_1 \gg N_2$ or $N_2 \gg N_1$ [11]. Apart from the fact that our approach produces a monoparametric set of optimum values for the diffusion parameters it also has the advantage of determining a closed form formula for the involved parameter τ and the convergence factor γ (see Table 1). These facts have two consequences. First, we avoid the computation of the second smallest and largest eigenvalue of the Laplacian matrix for the determination of the optimum value for τ. This is a time consuming process and was an open problem as far as we know. Second, we were able to study the convergence behaviour of the EDF method and predict its performance using the expression for γ. In order to further improve, by an order of magnitude, the rate of convergence of EDF we can apply accelerated techniques (Semi-Iterative, Second-Degree and Variable Extrapolation) following [15, 13, 7, 6, 12, 10].

N_1	N_2	Case	τ_o	$\gamma_o(M)$
Even	Even	1	$\left[3+2\sigma_2-\cos\frac{2\pi}{N_1}\right]^{-1}$	$\frac{1+2\sigma_2+\cos\frac{2\pi}{N_1}}{3+2\sigma_2-\cos\frac{2\pi}{N_1}}$
Odd	Odd	2	$\left[2+\sigma_2(1+\cos\frac{\pi}{N_2})+\cos\frac{\pi}{N_1}-\cos\frac{2\pi}{N_1}\right]^{-1}$	$\frac{\cos\frac{\pi}{N_1}+\cos\frac{2\pi}{N_1}+\sigma_2(1+\cos\frac{\pi}{N_2})}{2+\sigma_2(1+\cos\frac{\pi}{N_2})+\cos\frac{\pi}{N_1}-\cos\frac{2\pi}{N_1}}$
Even	Odd	3	$\left[3-\cos\frac{2\pi}{N_1}+\sigma_2(1+\cos\frac{\pi}{N_2})\right]^{-1}$	$\frac{1+\cos\frac{2\pi}{N_1}+\sigma_2(1+\cos\frac{\pi}{N_2})}{3-\cos\frac{2\pi}{N_1}+\sigma_2(1+\cos\frac{\pi}{N_2})}$
Odd	Even	4	$\left[2+2\sigma_2+\cos\frac{\pi}{N_1}-\cos\frac{2\pi}{N_1}\right]^{-1}$	$\frac{2\sigma_2+\cos\frac{\pi}{N_1}-\cos\frac{2\pi}{N_1}}{2+2\sigma_2+\cos\frac{\pi}{N_1}-\cos\frac{2\pi}{N_1}}$

Table 1. Formulae for the optimum τ_o and $\gamma_o(M)$

3 Accelerated methods

3.1 The Semi-Iterative method

We now consider iterative schemes for further accelerating the convergence of EDF. It is known [13, 15] that the convergence of (2) can be greatly accelerated if one uses the Semi-Iterative scheme

$$u^{(n+1)} = \rho_{n+1}\left[\bar{\rho}Mu^{(n)} + (1-\bar{\rho})u^{(n)}\right] + (1-\rho_{n+1})u^{(n-1)} \tag{9}$$

with

$$\bar{\rho} = \frac{2}{2-(\beta+\alpha)}, \rho_1 = 1, \rho_2 = \left(1-\frac{\sigma^2}{2}\right)^{-1}, \rho_{n+1} = \left(1-\frac{\sigma^2}{4}\rho_n\right)^{-1},\ n = 2,3,\ldots, \tag{10}$$

where

$$\sigma = \frac{\beta-\alpha}{2-(\beta+\alpha)}, \tag{11}$$

with

$$\alpha \le \mu_i \le \beta, \tag{12}$$

where μ_i are the eigenvalues of M. Because of (3), the eigenvalues of M and the Laplacian matrix L are related via the following relationship

$$\mu_i = 1-\tau\lambda_i, \quad i = 1,2,\ldots,n, \tag{13}$$

hence

$$\alpha = 1-\tau\lambda_n \quad \text{and} \quad \beta = 1-\tau\lambda_1, \tag{14}$$

since $\tau > 0$. Expressing $\bar{\rho}$ and σ (see (10) and (11)) in terms of λ_1 and λ_n, with the use of (14), we find that

$$\bar{\rho} = \frac{2}{\tau(\lambda_1+\lambda_n)} \quad \text{and} \quad \sigma = \frac{P(L)-1}{P(L)+1}, \tag{15}$$

with

$$P(L) = \frac{\lambda_n}{\lambda_1}. \tag{16}$$

But the optimum value of τ, τ_o is given by (5) which on substitution in the expression of $\bar{\rho}$ in (15) yields

$$\bar{\rho} = 1. \tag{17}$$

Moreover, for $\tau = \tau_o$, (6) becomes

$$\gamma_o(M) = \frac{P(L) - 1}{P(L) + 1}. \tag{18}$$

By (18) and (15) we have that

$$\sigma = \gamma_o(M). \tag{19}$$

Thus expressing (9) in terms of the Laplacian matrix L, using (3) and (17), we obtain

$$u^{(n+1)} = \rho_{n+1}(I - \tau_o L)u^{(n)} + (1 - \rho_{n+1})u^{(n-1)}, \tag{20}$$

where $\rho_{n+1}, \sigma, \tau_o$ are given by (10), (19) and (5), respectively. It is worth noting that σ is equal to $\gamma_o(M)$ (see (19)), which is the minimum value of the convergence factor of EDF. In addition, $\gamma_o(M)$ and τ_o, for EDF, are given by the expressions of Table 1 for the corresponding values of N_1 and N_2. It can be shown [7, 13, 15] that

$$\gamma(P_n(M)) = \frac{2r^{n/2}}{1 + r^n}, \tag{21}$$

where $P_n(M)$ is a certain polynomial in M (which is related to Chebyshev polynomials) and

$$r^{1/2} = \frac{\sigma}{1 + \sqrt{1 - \sigma^2}} = \frac{\sqrt{P(L)} - 1}{\sqrt{P(L)} + 1}.$$

In addition, for $P(L) \gg 1$, we have

$$r \simeq 1 - \frac{4}{\sqrt{P(L)}}, \tag{22}$$

thus the asymptotic average rate of convergence for the Semi-Iterative EDF (SI-EDF) method is given by

$$R_\infty(P_n(M)) = -\frac{1}{2}\log r \simeq \frac{2}{\sqrt{P(L)}} \tag{23}$$

as $n \to \infty$. From (8) and (23) the following relationship holds between the reciprocal rates of convergence[1] of SI-EDF and EDF

$$RR_\infty(P_n(M)) \simeq \frac{\sqrt{RR(M)}}{2}. \tag{24}$$

Therefore, the use of Semi-Iterative techniques results in an order of magnitude improvement in the reciprocal rate of convergence of EDF and in turn in the number of iterations.

[1] $RR(.) = \frac{1}{R(.)}$.

3.2 The Second Degree method

An accelerated scheme similar to (9) can be produced by considering constant iteration parameters throughout the process. It is known as the Second Degree (SD) method and is given by [15]

$$u^{(n+1)} = u^{(n)} + (\hat{\omega}_o - 1)(u^{(n)} - u^{(n-1)}) + \hat{\omega}_o(Mu^{(n)} - u^{(n)}), \tag{25}$$

where $\hat{\omega}_o = \frac{2}{1+\sqrt{1-\sigma^2}}$ with σ given by (19). Expressing (25) in terms of the Laplacian matrix L, we obtain

$$u^{(n+1)} = \hat{\omega}_o(I - \tau_o L)u^{(n)} + (1 - \hat{\omega}_o)u^{(n-1)}. \tag{26}$$

If $\hat{M}$ is the iteration matrix of the SD method, then [15]

$$\gamma(\hat{M}) = (\hat{\omega}_o - 1)^{1/2} = r^{1/2}, \tag{27}$$

thus the rate of convergence of the Second Degree EDF (SD-EDF) method is $R(\hat{M}) = -\frac{1}{2}\log r$, which is comparable with the one obtained by semi-iterative techniques. Also, by (23) and (27) we conclude that the rate of convergence of semi-iterative and second degree methods depends on the same quantity r. This implies that (23) and (24) hold also for the SD method.

3.3 The Variable Extrapolation method

In the previous sections, it was shown how we can find effective iterative processes. Note that in the new procedures each vector $u^{(n+1)}$ requires the computation of the two previous vectors $u^{(n)}$ and $u^{(n-1)}$. In case we face memory limitation problems we can consider another iterative scheme (sometimes called the Richardson method [13, 15]) of accelerating the EDF method, where $u^{(n+1)}$ is computed using $u^{(n)}$ only. This can be achieved by applying the following iterative scheme

$$u^{(n+1)} = \theta_{n+1}Mu^{(n)} + (1 - \theta_{n+1})u^{(n)}, \tag{28}$$

where $\theta_{n+1} = \frac{2}{2-(\beta-\alpha)\cos\frac{(2n-1)\pi}{2m} - (\beta+\alpha)}$. The iteration parameters θ_{n+1} are selected in the cyclic order $\theta_1, \theta_2, \ldots, \theta_m, \theta_1, \theta_2, \ldots, \theta_m$, where m is an integer. Expressing (28) in terms of L we have

$$u^{(n+1)} = (I - \hat{\theta}_{n+1}L)u^{(n)}, \tag{29}$$

where $\hat{\theta}_{n+1} = \theta_{n+1}\tau_o = \frac{\tau_o}{1-\sigma\cos\frac{(2n-1)\pi}{2m}}$ with σ given by (19). The spectral radius of the Variable Extrapolation EDF (VE-EDF) method is given by [15]

$$\gamma(P_{\ell m}(M)) = \left(\frac{2r^{m/2}}{1+r^m}\right)^{\ell}, \tag{30}$$

where ℓ is an integer determining the number of cycles. It can be seen from (21) and (30) that as m increases, then the rapidity of convergence tends to the one given by the semi-iterative method. However, numerical experiments [15] show that, for large m, numerical instability may occur. Also, it is undesirable to select m very large because convergence is expected after ℓm iterations.

4 Numerical results and conclusions

The purpose of the paper is to compare the application of accelerated techniques to the DF and EDF methods. As expected, both methods produce the same results in case of square torus [11], whereas for rectangular torus EDF tends to achieve twice as fast convergence rate compared to DF. In particular,

$$RR_\infty(EDF) \simeq \frac{1}{2} RR_\infty(DF) \tag{31}$$

when $N_1 \gg N_2$ or $N_1 \ll N_2$ (stretched torus) [11]. This is shown in columns DF and EDF of Tables 2 and 3, respectively, where we present the number of iterations for both schemes to converge using the same criterion. The convergence criterion for both schemes was $\sum_{i=1}^{|V|} \left(u_i^{(u)} - \bar{u}\right)^2 < \epsilon$, where $\bar{u} = \left(\sum_{i=1}^{|V|} u_i\right)/|V|$ and $\epsilon = 10^{-6}$, while the initial load $u_i^{(0)} = u_i$ was randomly distributed on the nodes of the graph. For all cases we used the optimum values for the parameters involved. These values are τ_o and $\sigma = \gamma_o(M)$, which were obtained by the formulae of Table 1.

N1 × N2	τ_o	$\gamma_o(M)$	DF	SI-DF	SD-DF	VE-DF	m	ℓ
5 × 5	0.232	0.679	40	16	18	18	9	2
5 × 11	0.254	0.919	174	38	41	42	14	3
5 × 21	0.260	0.977	605	74	79	81	27	3
5 × 51	0.262	0.996	3375	182	194	178	30	6
5 × 101	0.262	0.999	13102	366	366	597	30	20
6 × 6	0.222	0.778	60	21	23	22	10	2
6 × 10	0.200	0.908	184	55	38	38	19	2
6 × 20	0.246	0.976	572	73	76	81	27	3
6 × 50	0.249	0.996	3375	182	192	232	29	8
6 × 100	0.249	0.999	12799	366	361	575	34	20

Table 2. Number of iterations for DF, SI-DF, SD-DF and VE-DF methods

By (24) we have

$$RR_\infty(SI-EDF) \simeq \frac{1}{2}(RR_\infty(EDF))^{1/2}, \tag{32}$$

or using (31),

$$RR_\infty(SI-EDF) \simeq \frac{1}{2\sqrt{2}}(RR_\infty(DF))^{1/2}. \tag{33}$$

But (24) holds also for the DF method, this means that

$$RR_\infty(SI-DF) \simeq \frac{1}{2}(RR_\infty(DF))^{1/2}. \tag{34}$$

Therefore, (33), because of (34), yields

$$RR_\infty(SI-EDF) \simeq \frac{1}{\sqrt{2}} RR_\infty(SI-DF) \tag{35}$$

which indicates that the number of iterations of SI-EDF will be approximately 30% less than the number of iterations of SI-DF in case of stretched torus. Clearly, this result holds for the other two accelerated methods (SD-EDF and VE-EDF) as both these methods tend to obtain the same rate of convergence as SI-EDF (see (27) and (30)). In Tables 2 and 3 we also present the number of iterations for the accelerated versions of DF and EDF methods, respectively. For the VE version of both methods the value of m was determined experimentaly such that the number of iterations is minimum. The results of Tables 2 and 3 clearly show that fixing one dimension of a torus and increasing the other, the number of iterations of the accelerated versions of EDF (SI-EDF, SD-EDF, VE-EDF) is 30% less than the number of iterations of the corresponding versions of DF. Similar results were also obtained in the odd/even cases. Therefore, our theoretical expectation, which is expressed by (35), is verified. Finally, comparing the accelerated versions of EDF we note that SI and SD have similar behaviour, which is better than the VE version (Table 3).

N1 × N2	τ_{opt}	γ_{opt}	EDF	SI-EDF	SD-EDF	VE-EDF	m	ℓ
5 × 5	0.232	0.679	40	16	18	18	9	2
5 × 11	0.091	0.874	113	30	32	30	30	1
5 × 21	0.029	0.958	348	54	59	58	29	2
5 × 51	0.005	0.992	1966	133	142	161	27	6
5 × 101	0.001	0.998	7176	264	269	387	30	13
6 × 6	0.222	0.777	60	21	23	22	10	2
6 × 10	0.129	0.870	109	29	32	30	30	1
6 × 20	0.043	0.956	328	53	56	58	29	2
6 × 50	0.007	0.992	1770	130	137	150	30	5
6 × 100	0.001	0.998	6824	261	260	385	26	13

Table 3. Number of iterations for EDF, SI-EDF, SD-EDF and VE-EDF methods

Acknowledgements

We would like to thank the anonymous referees for their positive comments which resulted in the present form of the paper.
The project is co-funded by the European Social Fund and National Resources (EPEAK II) Pythagoras.

References

1. J. E. Boillat, Load balancing and poisson equation in a graph, *Concurrency: Practice and Experience* **2**, 1990, 289-313.
2. G. Cybenko, Dynamic load balancing for distributed memory multi-processors, *Journal of Parallel and Distributed Computing* **7**, 1989, 279-301.
3. R. Diekmann, S. Muthukrishnan, M. V. Nayakkankuppam, Engineering diffusive load balancing algorithms using experiments, *IRREGULAR'97*, **LNCS 1253**, 1997, 111-122.
4. R. Diekmann, A. Frommer, B. Monien, Efficient schemes for nearest neighbour load balancing, *Parallel Computing* **25**, 1999, 789-812.
5. R. Elsässer, B. Monien, S. Schamberger, G. Rote, Toward optimal diffusion matrices, *International Parallel and Distributed Processing Symposium*, IEEE Computer Society Press, 2002.
6. B. Ghosh, S. Muthukrishnan and M. H. Schultz, First and second order diffusive methods for rapid, coarse, distributed load balancing, *8thACM Symposium on Parallesim in Algorithms and Architectures*, 1996, 72-81.
7. G. H. Golub and R. S. Varga, Chebyshev semi-iterative methods, successive overrelaxation iterative methods, and second-order Richardson iterative methods, *Numer. Math. Parts I and II* **3**, 1961, 147-168.
8. Y. F. Hu and R. J. Blake, An improved diffusion algorithm for dynamic load balancing, *Parallel Computing* **25**, 1999, 417-444.
9. G. Karagiorgos and N. M. Missirlis, Fourier analysis for solving the load balancing problem, *Foundations of Computing and Decision Sciences* **27**, No 3, 2002.
10. G. Karagiorgos and N. M. Missirlis, Accelerated diffusion algorithms for dynamic load balancing, *Information Processing Letters* **84**, 2002, 61-67.
11. G. Karagiorgos and N. M. Missirlis, Local convergence analysis for the diffusion load balancing methdod in torus *(submitted)*.
12. S. Muthukrishnan, B. Ghosh and M. H. Schultz, First and second order Diffusive methods for rapid, coarse, distributed load balancing, *Theory of Computing Systems* **31**, 1998, 331-354.
13. R. Varga, Matrix iterative analysis, *Prentice-Hall*, Englewood Cliffs, NJ, 1962.
14. C. Z. Xu and F. C. M. Lau, Load balancing in parallel computers: *Theory and Practice*, Kluwer Academic Publishers, Dordrecht, 1997.
15. D. M. Young, Iterative solution of large linear systems, *Academic Press*, New York, 1971.