論 文
51P-2-1

전력 조류 계산의 병렬처리에 관한 연구

A Development of Parallel Processing for Power Flow analysis

李 春 模*
(Chun-Mo Lee)

Abstract - Parallel processing is able to be used effectively on computationally intense power system problems. But this technology is not still available is not only parallel computer but also parallel processing scheme. Testing these algorithms to ensure accuracy, and evaluation of their performance is also an issue. Although a significant amount of parallel algorithms of power system problem have been developed in last decade, actual testing on parallel computer architectures lies in the beginning stages because no clear cut paths. This paper presents Jacobian modeling method to supply the base being able to treat power flow by newton's method by the computer. This method is to assign and to compute teared blocks of sparse matrix at each parallel processors. The testing to insure accuracy of developed method have been done on serial computer by trying to simulate a parallel environment.

Key Words : Power flow, Parallel Processing method, Ordering scheme, Jacobian matrix

1. 서 론

병렬처리(parallel processing)는 데이터교환장치를 가지고 있는 두개 이상의 프로세서를 이용 어떤 문제의 해를 분담, 처리하는 것을 의미한다.[1] 최근에는 웹 기반으로 개발할 수 있는 자바 프로그램이 대중화 되면서 인터넷을 통한 병렬처리의 실현이 가능하게 되고 있어 비록 효율에서는 떨어지지만 값비싼 슈퍼 컴퓨터를 이용하지 않는 기법이 연구되고 있다.[2]-[4]

전력계통의 문제를 해결하기 위한 병렬처리기법은 크게 요소기법(elemental scheme)과 블록기법(block scheme)으로 구분할 수 있다. [5] 요소기법은 각 연산단계마다 많은 데이타 정보들의 교환이 이루어져야 하므로 프로세서간 데이타 교환을 위한 통신부담(communication overhead) 이 매우 커지게 되는 단점이 있다. 블록기법은 병렬성은 타스크가 아닌 블록단위로 존재하여 각 프로세서가 연산을 수행할 때 다른 프로세서와의 데이터 이동에 의한 통신부담을 최소가 되도록 하는 것이다. [6]-[9] 따라서 컴퓨터 구조상 요소기법은 데이터 교환이 용이한 공유 메모리 시스템에 적합하고, 블록기법은 데이터 교환을 최소화 하므로 분산 메모리 시스템에 적합하다. 그러나 병렬 컴퓨터의 가격은 분산 메모리 시스템에 비하여 공유 메모리 시스템이 구조상 매우 고가이기 때문에 비록 요소기법의 알고리즘이 정립되어 있기는 하지만 실용화 되고 있지 못한 실정에 있다.

따라서 본연구에서는 분산메모리 시스템을 이용하기 위한 블록 기법에 대한 연구를 수행하였다. 전력계통을 여러개의 블록으로 분할하여 병렬컴퓨터로 처리하기 위하여는 계통의 모선을 최적서열 및 계통의 분할 알고리즘 뿐만 아니라 각각의 프로세서에서 전력계통 해석의 연산처리를 병렬로 수행할 수 있도록 하는 병렬처리기법이 필요하다. 그러나 실제 전력계통 조류계산을 병렬처리하기 위하여는 초기의 서열단계에서부터 마지막 반복해까지의 전 과정을 병렬처리 할 수 있는 기법이 필요하게 된다. [10][11] 따라서 전력계통의 조류계산시 꼭 수반되는 Jacobian 행렬 연산의 모델링 및 LU분할에 의한 직접해법의 병렬처리는 필수적인 것이다. 따라서 본 논문에서는 실제 전력계통 해석 문제에서 자주 접하게 되는 전력 조류계산을 병렬처리 할 수 있도록 Jacobian 행렬의 연산과정 및 LU삼각화 직접해법을 블록기법에 의한 병렬 처리가 가능하도록 병렬처리기법을 제시하여 이미 개발을 완료한 모선의 재서열 알고리즘을 적용. 병렬처리가 가능하도록 하였으며, 개발된 기법이 정확히 시행될 수 있는지를 평가하기 위하여 모델전력계통에 대한 각 프로세서의 병렬처리기능을 프로그램으로 구성, 시뮬레이션을 실시하였다.

2. 병렬처리 알고리즘

2.1 요소기법

요소기법은 프로세서의 수가 제약받지 않는다고 가정하면 프로그램의 작은 단위인 코드(code)들에 대한 연산을 각 프

* 正 會 員 : 忠淸大學 電氣情報科 敎授 · 工博
接受日字 : 2002年 3月 18日
最終完了 : 2002年 4月 12日

로세서에 할당하여 연산시키는 방법이다. 전진대입 과정중 요소기법은 비대각 요소들의 연산을 수행하기 위하여는 그 열번호의 대각요소가 계산될때 까지 각 프로세서들은 대기(idle)하고 있어야 하며 각 프로세서에서 수행된 비대각요소들의 값은 다음 대각요소의 계산을 위하여 대각요소를 연산하는 프로세서에 전송되어져야 한다. 이와 같이 요소기법은 각 연산단계마다 포함되는 대기시간(idle time)과 연산단계마다 각 프로세서들이 다른 프로세서에 의해서 구해진 연산 결과를 전송받아야 하기 때문에 많은 양의 데이타 교환으로 인한 통신부담(communication overhead)이 커다란 문제로 지적되고 있다.[12]

2.2 블록기법

블록기법(block scheme)은 행렬을 작은 블록으로 분할하여 각 블록들을 병렬처리시스템의 각 프로세서에 할당시켜 데이타 정보 교환의 수를 최소화하는 방법이다. 각 타스크에 대한 연산순서는 요소기법과 같으나 연산과정을 블록으로 분할하여 각 프로세서에서 처리토록 하므로써 데이타 교환에 의한 통신부담을 감소시킬 수 있게 한 것이다. 이러한 블록기법은 전력계통과 같이 계통을 작은 cluster들로 분할이 가능한 경우 각 분할된 cluster들의 독립성이 커져 병렬처리 과정중 발생하는 데이타 교환에 의한 통신부담을 최소화 할수 있게 된다. 그러나 계통을 완전히 분리하는 것은 사실상 불가능하여 계통을 분할할 때에는 각 분할된 블록들을 연결하는 상호 연결 부분 모선(interconnection bus)이 존재하게 된다. 그러므로 분할된 블록과 공통부분모선의 요소들에 대한 연산을 병렬처리하기 위하여는 각 프로세서에 분할된 블록을 할당하는 것 뿐만아니라 상호 연결 부분 모선들의 연산처리를 위한 병렬처리기법이 요구된다.

분산 메모리형 병렬컴퓨터에 프로세서의 수에 관계없이 계통을 규칙적인 방법에 의하여 분할할 수 있는 방법과 분할된 cluster들을 효율적으로 병렬처리 할 수 있는 블록 병렬처리기법을 개발하여 적용한다면 각 프로세서들이 필요한 공유 데이타의 양을 감소시키므로 데이타 교환을 위한 노력의 최소화로 병렬처리의 효율을 증대시킬 수 있을 뿐만 아니라, 분산 메모리형 컴퓨터의 커다란 문제로 지적되어 온 컴파일러의 표준화 문제를 보다 쉽게 해결할 수 있게 된다.[14]

3. 병렬처리를 위한 Jacobian 행렬 모델링

본 연구에서는 동시에 여러 프로세서들간의 데이터 교환이 가능한 하이퍼 연결망 구조를 갖는 분산 메모리형 병렬컴퓨터를 이용하여 전력조류계산을 병렬처리할 수 있도록 어드미턴스 행렬(Y_{BUS})의 분할된 블록들의 타스크 할당과 Jacobian 행렬의 모델링을 수행한다.

그림 3.1은 N 모선 계통을 m-1개의 크러스터로 분할한 후의 어드미턴스 행렬을 m개의 블록 형태로 나타낸 것이다. 여기서 m번째의 블록행렬은 계통을 분할하는 모선들에 해당하는 부분이다.

Jacobian행렬의 계수는 어드미턴스 행렬(Y_{BUS})과 전압을 유효분과 무효분으로 나누어 다음과 같이 구해진다.

i) 비대각 요소(J_{pq})

$$H_{pq} = L_{pq} = |V_p||V_q|(G_{pq}SIN\theta_{pq} - B_{pq}COS\theta_{pq})$$

$$N_{pq} = -M_{pq} = |V_p||V_q|(G_{pq}COS\theta_{pq} + B_{pq}SIN\theta_{pq}) \quad p \neq q \quad (3\text{-}1)$$

ii) 대각 요소(J_{pp})

$$H_{pp} = -Q_p - B_{pp}|V_p|^2$$
$$L_{pp} = Q_p - B_{pp}|V_p|^2$$
$$N_{pp} = P_p + G_{pp}|V_p|^2$$
$$M_{pp} = P_p - G_{pp}|V_p|^2 \quad p = q \quad (3\text{-}2)$$

여기서 $V_p = |V_p| \angle\theta_p$; $\theta_{pq} = \theta_p - \theta_q$

$Y_{pq} = G_{pq} + j\,B_{pq}$

Jacobian 행렬은 Y_{BUS}행렬과 같은 구조로 구해지게 되는데, 그림 3.1과 같이 블록으로 분할된 경우 Jacobian 행렬의 비대각 블록(non-diagonal block)은 비대각 요소 "J_{pq}"로 부터 구해진다. 식 (3-1)에서 J_{pq}는 동일한 모선번호 p,q의 전압 V_p,V_q와 어드미턴스 요소 Y_{pq}에 의하여 구해지므로 Jacobian 행렬의 비대각 요소의 연산은 해당 블록 모선의 전압과 어드미턴스만 필요하다.

그러나 대각블록(diagonal block)의 요소 "J_{pp}"는 식 (3-2)에서처럼 모선 p의 행에 포함되어 있는 어드미턴스 행렬의 영이 아닌 요소와 이 요소들의 열 번호에 해당하는 전압을 필요로 하게 된다. 따라서

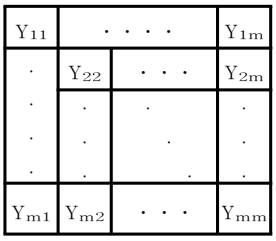
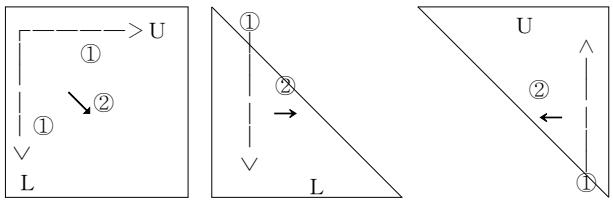


그림 3.1 m개의 대각 블록으로 구성된 Y_{BUS}
Fig 3.1 Y_{BUS} composed with m diagonal block

Jacobian 행렬 연산을 병렬처리하기 위하여는 그림 3.1과 같이 행으로 구분된 Y_{BUS}행렬의 대각블록, 비대각블록 및 해당 전압을 같은 프로세서의 기억장치에 할당하여야 한다. 이때 마지막 부분에 위치한 상호 연결부분 모선 (interconnection bus)의 전압 벡터는 각 프로세서의 기억장치에 공통으로 기억시켜 연산을 수행하게 된다. 그림 3.2는 LU분할 및 전진/역진대입의 연산순서를 나타낸 것이다. 이 그림에서 보는 바와 같이 LU 분할의 연산 및 전진/역진 대입은 먼저 ①과 같이



(a)LU 분할 (b)전진대입 (c)역진대입

그림 3.2 LU분할 및 전진/역진대입의 연산순서
Fig 3.2 The calculation sequence of LU factorization and forward/backward substitution

행렬의 각 행과 열에 대하여 연산을 수행한 후 ②의 방향으로 순차적으로 수행된다. 따라서 Jacobian행렬이 그림 3.3과 같이 할당될 경우 LU분할에 의한 직접해법 적용시 각 연산단계마다 프로세서들은 "processor #M"의 국부 기억장치(local memory)에 저장된 각각의 비대각 블록행렬 $[J_{m1}]$,$[J_{m2}]$, ···· ,$[J_{m-1,m}]$의 데이터들을 전송받아 연산을 수행해야만 한다.

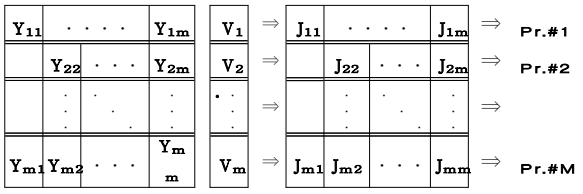


그림 3.3 Y_{BUS} 행렬 및 Jacobian행렬의 타스크 배분
Fig 3.3 Task distribution for Y_{BUS} and Jacobian matrix

그림 3.4는 LU 분할 및 전진/역진대입의 연산순서를 감안한 데이터 할당 방법을 나타낸 것으로 Y_{BUS} 행렬의 대각 블록행렬과 해당 비대각 블록행렬을 동일한 프로세서의 기억장치에 할당시키도록 하였다. 이와 같은 데이터 할당은 Jacobian 행렬의 연산시에는 데이터 교환량이 증가하나 조류계산의 연산량의 대부분을 점유하는 LU 삼각화 직접해법의 병렬처리시에는 각 프로세서가 연산 수행중 필요한 대부분의 데이터를 자신의 국부 기억장치에 저장하고 있어 데이타 교환을 최소화 할 수 있고 동일한 데이터의 공유를 억제하므로 전체의 기억용량을 감소시킬 수 있게 된다. 이때 각 프로세서는 각각의 저장장치에 기억된 비대각 블록행렬 $[Y_{m,1}][Y_{m,2}]\cdots[Y_{m-1,m}]$의 데이터와 식 (3-2)에 의하여 공통부분 모선에 대한 H_{pp}, N_{pp}, M_{pp}, L_{pp}의 요소들의 부분 해가 구해진다. 여기서 공통부분의 Jacobian 행렬인 $[J_{mm}]$은 각 프로세서에서 구해진 공통부분의 $[J^1{}_{mm}]$,$[J^2{}_{mm}]$ ···· $[J^M{}_{mm}]$로 부터 식 (3-3)과 같은 방법에 의하여 구해 질 수 있다.

$$[J_{mm}]=[J^1{}_{mm}]+[J^2{}_{mm}]+\cdots+[J^M{}_{mm}] \qquad (3\text{-}3)$$

이때 병렬처리 연산 단계에서 주로 상호 연결 부분에 대한 연산을 위한 데이터 전송 및 각 프로세서의 연산 결과에 대한 덧셈이 필요하게 되는데 이 과정이 블록 병렬처리기법으로 처리할 때 추가로 소요되는 시간이다.

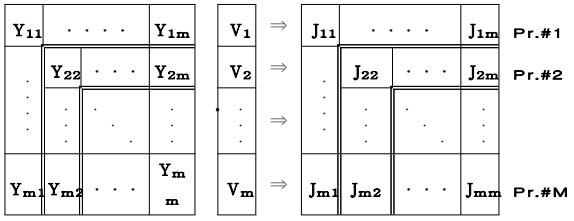


그림 3.4 제안된 Jacobian 행렬 연산의 타스크 배분
Fig 3.4 Proposed task distribution for Jacobian matrix calculation

4. 적용예 및 결과 고찰

4.1 전력계통의 구성과 타스크 할당

본 논문에서 제시한 Jacobian 행렬의 모델링을 적용하기 위하여 그림 4.1의 모델 계통에 적용하였는데 이 6모선 모델 계통에 대한 그림은 병렬처리를 위한 분할기법 11)에 의하여 분할한 결과를 나타낸 것이며 모선 "5"와 "6"에 의하여 두개의 블록으로 분할되었다. 모선번호는 분할의 효율을 향상시키기 이한 MPRLD 서열산법12)을 적용하여 재부여 된 것이며 ( ) 안의 번호가 원래의 모선 번호이다.

이 모델 계통에 대한 데이터는 표 4.1과 표4.2에 50MVA를 기준으로 한 단위법으로 표시하였으며 기준모선은(slack bus)은 1번 모선으로 가정하였다.

본 실험은 각 프로세서의 연산을 프로그램으로

No. Type
1 Slack
2 P-V
3 P-Q
4 P-Q
5 P-Q
6 P-Q

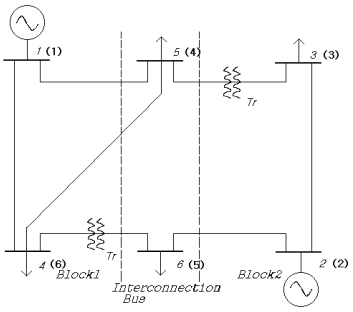


그림 4.1 분할된 6모선 모델 전력 계통
Fig. 4.1 The tearing of 6 bus model power system

표 4.1 모델 전력계통의 선로 데이터
Table 4.1 Line Data of Model Electric Power System

Between Buses	Line impedance	Half line charging admittance	Off-nominal turns ratio(a)
1 - 5	0.08 +j0.37	0.007	-
1 - 4	0.123+j0.518	0.010	-
2 - 3	0.723+j1.05	0.	-
2 - 6	0.282+j0.64	0.	-
5 - 3	0. +j0.133	0.	0.909
5 - 4	0.097+j0.407	0.0076	-
4 - 6	0. +j0.3	0.	0.976

A static shunt capacitor at bus 5 is present with admittance=j0.005

표 4.2 모델 전력계통의 모선 데이터
Table 4.2 Bus Data of Model Electric Power System

Bus No.	Type	Bus Loading		Specified Voltage[V]	Reactive power Limit	
		P(MW)	Q(MVAR)		Q_{min}	Q_{max}
1	Slack	-	-	1.05	-	-
2	P-Q	0.5	-	1.10	0.13	0.2
3	P-Q	0.55	0.13	-	-	-
4	P-Q	0.5	0.05	-	-	-
5	P-Q	-		-	-	-
6	P-Q	0.3	0.18	-	-	-

구성하여 한대의 컴퓨터로 시뮬레이션 시키는 방법에 의하여 수행하였다. 프로세서간의 연산 데이터의 교환은 각 프로세서의 연산 결과를 필요로 하는 프로세서, 즉 시뮬레이션된 프로그램에 데이터를 연산전에 미리 입력시키는 방법에 의하여 처리하였으며, 병렬처리기법의 연산 수행 결과를 검증하기 위하여 분할하지 않은 계통을 순차적인 방법에 구한 Jacobian 행렬을 얻은 결과와 2개의 크러스터로 분할된 계통에 대해서 Jacobian 행렬의 모델링 기법을 시뮬레이션하여 얻은 결과를 비교, 검토하였다.

4.2 Jacobian 행렬의 모델링에 의한 전력조류계산의 병렬처리

모델계통의 선로정수로 부터 MPRLD 서열산법으로 재서열된 어드미턴스 행렬(Y_{bus})을 구하여 각 모선들을 분할된 계통의 그룹으로 표현하면 표 4.3과 같이 구해진다. 이 어드미턴스 행렬을 제시한 데이터 할당방법에 의하여 3개의 프로세서 "#a", "#b", "#c"의 각 기억장치에 각각의 블록을 할당시도록 하엿다. 표의 제일 좌측과 상측에 부여된 번호는 모선번호를 나타낸 것이다.

표 4.3 모델계통의 어드미턴스 행렬 (Y_{bus})

Table 4.3 Admittance Matrix of Model System (Y_{bus})

	1	2	3	4	5	6
1	0.9922 -j4.4025			-0.4339 j1.8275	-0.5583 j2.5820	
2		1.0214 -j1.9545	-0.4449 j0.6401			-0.5765 j1.3085
3		-0.4449 j0.6401	0.4449 -j8.1649		0.0 j8.2715	0.0 j0.0
4	-0.4339 j1.8275			0.9880 -j7.6517	-0.5541 j2.3249	0.0 j3.4153
5	-0.5583 j2.5820		0.0 j8.2715	-0.5541 j2.3249	1.1124 -j13.9995	0.0 j0.0
6		-0.5765 j1.3085	0.0 j0.0	0.0 j3.4153	0.0 j0.0	0.5765 -j4.6418
	Block 1	Block 2		Block 1	Interconnection Bus	

각 프로세서의 기억장치에 할당된 어드미턴스 행렬과 모선 데이터로 부터 제시된 방법에 의하여 각 프로세서에 의하여 Jacobian 행렬의 모델링을 수행하여

표 4.4 병렬처리를 위한 Jacobian행렬의 모델링에 의하여 구해진 진 값

Table 4.4 The value by Jacobian matrix modeling for parallel processing

2.15006	-.71071	-.48939			-1.43935	-.63415		
-.71071	8.98221	.40041	-8.27150	.00000	.00000	.00000		
-.48939	.40041	7.34759	.00000	-8.27150	.00000	.00000		
	-8.27150	.00000	13.30750	1.08448	.00000	.00000	-2.32490	.55410
	.00000	-8.27150	1.08448	14.69150	.00000	.00000	-.55410	-2.32490
-1.43935	.00000	.00000	.00000	.00000	4.85465	.51885	-3.41530	.00000
-.63415	.00000	.00000	.00000	.00000	.51885	4.42895	-2.32490	.55410
			-2.32490	-.55410	-3.41530	.00000	7.65907	.96631
			.55410	-2.32490	.00000	-3.41530	.96631	7.64433

병렬처리를 위한 Jacobian행렬의 모델링에 의하여 구해진 값은 표4.4과 같다.

이와 같이 본 논문에서 제안한 병렬처리를 위한 Jacobian 행렬의 모델링의 결과와 순차적인 방법에 의하여 얻은 결과에서 나타나는 작은 오차는 시뮬레이션 과정에서 데이터 교환시 소수점 자리의 간략화 과정에서 발생한 것으로 병렬처리를 위한 Jacobian 행렬의 모델링이 정확히 수행되었음을 알 수 있다.

제안된 병렬처리기법의 연산시간은 각 블록의 크기중 가장 큰 블록에 포함된 모선 수와 이들 모선과 관련된 삼각행렬 [L],[U]에 있는 영이 아닌 요소의 수에 의하여 좌우되는데, 이 두 요소들은 어드미턴스 행렬의 재서열에 따라 변할 수 있다. 또한, 경계모선의 숫자는 블록 병렬처리기법의 연산 효율에 매우 중요한 영향을 미치게 된다. 그러나 블록기법에 의한 병렬처리는 요소기법에 비하여 데이터 교환에 소요되는 시간은 무시할 수 있을 정도로 작고, 계통의 크기가 커질 수록 공통부분의 모선 수는 분할그룹의 모선 수에 비하여 매우 작게 되므로, 상호연결망 (interconnection network)을 이용한 정보교환의 부담(communication over head)이 매우 작아지고, 블록기법의 공통부분을 처리하기 위한 데이터 교환은 전체 연산에 비하여 아주 작게 된다.본 실험은 한대의 컴퓨터로 각 프로세서의 연산을 프로그램으로 구성하여 시뮬레이션 시키는 방법에 의하여 수행하였으며, 프로세서간의 연산 데이터의 교환은 각 프로세서의 연산 결과를 필요로 하는 프로세서, 즉, 시뮬레이션된 프로그램에 데이터를 연산 전에 미리 입력시키는 방법에 의하여 처리하였다. 그림

표 4.5 각 프로세서에서 병렬처리 연산 결과

Table 4.5 Parallel solution obtained by each processors

"Processor #3"

열번호	x(i) = $[U]^{-1} \cdot y(i)$
6	-.22392
7	.12751
8	-.21971
9	.15450

"Processor #1,#2"

열번호	Processor #1	Processor #2
1		.04475
2		-.27008
3		.25497
4	-.23791	
5	.14854	

표 4.6 Serial 로 수행된 첫번째 반복계산 결과

Table 4.6 Serial computation result of 1st iteration

x(i) = 〔 Δθ, | ΔV | / | V | 〕	
Δθ2	.04755
Δθ3	-.27008
| ΔV3 | / | V3 |	.25497
Δθ6	-.23791
| ΔV4 | / | V4 |	.14854
Δθ4	-.22392
| ΔV5 | / | V5 |	.12751
Δθ5	-.21971
| ΔV6 | / | V6 |	.15450

4.2는 IEEE 14 모선, 30,모선, 57모선 계통에 대한 시뮬레이션한 결과를 나탄낸 것이다. 이들 모선 계통에 대한 병렬처리의 평가 결과는 프로세서 수를 4개, 9개 및 5개가 되었을 경우 최적임을 보이고 있다.

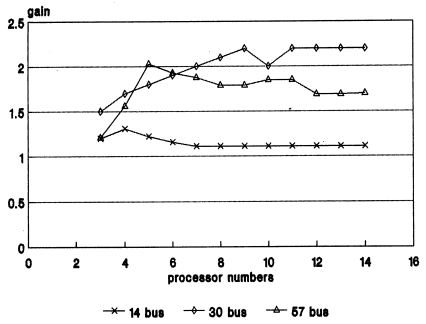


그림 4.2 전력조류 계산에 대한 병렬처리의 연산 이득
Fig. 4.2 Parallel computation gain for power flow

5. 결 론

전력계통의 조류계산시 꼭 수반되는 Jacobian 행렬 연산의 병렬처리를 위한 모델링은 비록 전체의 연산시간에 차지하는 비중이 적기는 하지만은 실제 전력계통 조류계산을 병렬처리과정은 필수적인 과정이다. 따라서 본 논문에서는 실제 전력계통 해석 문제에서 자주 접하게 되는 Newton 법에 의한 전력 조류계산을 병렬처리할 수 있도록 Jacobian 행렬의 연산과정을 처리하기 위한 모델링 기법을 제시하였으며, 실제 전력조류 계산에 적용하기 위하여 모델 전력 계통 해석에 적용하여 개발된 기법이 정확히 시행될 수 있음을 보였다.

본 논문에서 수행된 중요한 내용은 다음과 같다.

1) 이 본 저자가 수행한 병렬처리를 위하여 개발된 MPRLD 서열산법과 전력계통의 분할기법을 적용하여 연구의 연속을 기하였다.

2) 병렬컴퓨터의 각 프로세서에 대한 어드미턴스 Y_{bus}행렬의 타스크 할당 방법을 제시하였다

3) 전력조류계산의 병렬처리를 위한 Jacobian행렬의 연산과정에 대한 모델링 기법을 제시하였으며, 모델전력계통에 대한 각 프로세서의 병렬처리기능을 프로그램으로 구성,시뮬레이션을 실시하여 신뢰성을 확보하였다.

이러한 결과는 이미 개발된 MPRLD 서열산법과 전력계통 분할기법과 함께 향후 전력조류계산의 완전한 병렬처리 프로그램의 개발이 가능하도록 할 것이다.

본 논문은 2000년도 충청대학 자체 연구비 지원에 의하여 이루어 졌음

참 고 문 헌

[1] M. Feilmeier, G. R. Joubert, U. Schendel, "Parallel Computing", North-Holland-Amsterdam, 1986.

[2] Qusay H.Manmoud, "Distributed Programing With Java", 인포복 출판사 , 2001

[3] Merin hughes, "Java Network Progrming", 인포.북 출판사, 2000

[4] M.Pedro Silva, Alexandra V.Sousa, "Web based DMS-distribution management system", IEEE Trans. on power system p2338-2343. 2000.

[5] 이 춘모, 서 희석 "스파스 선형방정식의 병렬처리에 관한 연구", 충청전문대학 논문집 P225-246, 1997. 12.

[6] C. P. Arnold, M. I. Parr,and M. B. Dewe,"An efficient parallel algorithm for the solution of large sparse linear matrix equations", IEEE Trans. on Computers, Vol. C-32, No.3, pp.265-272, March 1983

[7] O. Wing, J. w. Huang,"A computation model of parallel solution of linear equations",IEEE Trans. Comput., Vol. C-29, pp.632-638, July, 1980

[8] Betancourt and F.L.Alvarado, "Parallel Inversion of Sparse matrix", IEEE Trans. on PWRS-1, pp.74-81, 1986

[9] David C.Yu and H. Wang, "A new approach the forward and backward substitution of parallel solution of sparse linear equation based on data flow architecture", IEEE Trans. ON PAS Vol. 5, No 2, May, 1990

[10] 이 춘모, 조 인숙, 신 명철, " 최적서열산법을 이용한 전력계통 분할 해법의 개발", 성대논문집(과학기술편) 제41집 No. 1, pp. 65-79, 1990

[11] 이 춘모, 신 명철, "전력계통해석의 병렬처리를 위한 분할기법에 관한 연구" 대한전기학회지, 제45권 ,제12호 pp.1672-1678, 1996. 12.

[12] Danial J. Tylavsky, Anjan Bose " Parallel Processing in Power Systems Computation," An IEEE committe report by task force of the computer and analytical methods subcommittee of the power systems engineering committee,IEEE Trans.PAS Vol. 7, No.2, pp 629-638, May

저 자 소 개

이 춘 모(李 春 模)

1957년 3월27일생. 1983년 성균관대 전기공학과 졸업. 1985년 동 대학원 전기공학과 졸업(석사). 1993년 동 대학원 전기공학과 졸업(박사). 1992년~현재 충청대학 전기정보과 교수

Tel : 043-230-2352
Fax : 043-230-2279
E-mail : lhju2@ok.ac.kr