Tabu 탐색 기법을 이용한 분산 컴퓨터 시스템 설계 - Design of Distributed Computer Systems Using Tabu Search Method -

홍 진 원*
Hong, Jin-Won
김 재 련**
Kim, Jae Yearn

Abstract

This paper determines the allocation of computers and data files to minimize the sum of processing and communication costs which occur in processing jobs at each node.

The problem of optimally configuring a distributed computer system belongs to the class of NP-Complete problems and the object function of this paper is nonlinear function and is hard to solve.

This paper seeks the solution of distributed processing system by Tabu Search. Firstly, it presents the method of generating the starting solution proper to the distributed processing system. Secondly, it develops the method of searching neighborhood solutions. Finally, it determines the Tabu restriction appropriate to the distributed processing system. According to the experimental results, this algorithm solves a real sized problems in reasonable time and is effective in the convergence of the solution. The algorithm developed in this paper is also applicable to the general allocation problems of the distributed processing system.

1. 서론

최근에는 다양한 기능과 규모의 컴퓨터와 데이타 파일이 지역적으로 여러 곳에 흩어져 있는 분산 처리 시스템이 사용자들의 관심을 끌고 있는데 그 이유는 분산 처리 시스템은 중앙 집중식 시스템에 비하여 신뢰도(reliability)와 가용도(availability)를 높여주며, 여러 컴퓨터의 병행처리를 통해 그 실행 능률(performance)를 향상시켜줄 수 있기 때문이다.

분산 처리 시스템은 통신 네트워크(communication network)로 연결된 computing 노드(site)들의 집합으로 구성되어 있다. 각 노드에는 처리, 저장 그리고 통신 장비를 갖추고 있다.

파일 할당 문제는 NP-Complete 문제임이 알려져 있으므로 최적해를 구하는 다항식 알고리듬은 존재하지 않는다[2]. 따라서 이것의 해결 방안으로 발견적 기법이 흔히 사용된다.

Ghosh와 Murthy[4]는 파일의 할당과 조회 경로(query route)를 결정하는 문제를 0-1 정

* 한양대학교 산업공학과 대학원
** 한양대학교 산업공학과 교수

수계획 모형으로 정식화 하였다. Ghosh는 발견적 기법으로 목적식의 상한(upper bound)을 구한 후 lagrangian relaxation을 사용하여 목적식의 하한(lower bound)을 구하여 이들을 비교한다. 발견적 기법으로 구한 해가 작으면 여기서 끝내고 크면 이것을 초기해로 하여 Branch & Bound로 계속 풀어 나간다. Gavish[2]는 컴퓨터와 데이터 파일을 할당하고 보고서의 경로를 결정하는 문제를 0-1 정수계획 모형으로 만들었다. 이 논문에서도 발견적 기법으로 목적식의 상한을 구하고 lagrangian relaxation으로 목적식의 하한을 구한 후 이들의 차이를 계속 줄여 나가는 방법을 사용하였다. Lee와 Liu Sheng[5]도 파일의 할당과 조회 경로(query route)를 결정하는 문제를 0-1 정수계획 모형으로 정식화 하였다. 그들이 제시한 모형은 목적식이 운영비용(operating cost), 가용도, 그리고 반응시간(response time)으로 구성된 모형이다. 각각의 목적식을 따로 풀어서 구한 해 중 의사결정자가 어느 요소를 더 중시하느냐에 따라서 해를 선택한다. 이 문제에서는 가용도에 관한 목적식이 비선형 형태로 되어 있는 것을 선형 형태로 변환하여 해를 구하였다. Jain[6]의 모형은 목적식이 비선형 형태이므로 경사법(gradient method)을 사용하여 문제를 풀었는데 이 방법은 국부 최적해(local optimum)에 빠지게 되면 벗어날 수 없다는 단점이 있다. 최근에 Kumar, Pathak 그리고 Gupta[1]는 신뢰도를 최대화하는 목적식을 가진 파일 할당 문제를 Genetic 알고리듬을 사용하여 풀었다. 그는 노드가 6개인 문제에 대하여 전체 해 공간(complete solution space)을 전부 검토(exhaustive enumeration)하여 찾아낸 최적해(optimum)와 Genetic 알고리듬으로 구한 해를비교하였다.

본 연구에서 사용하는 모형[6]의 목적식은 비선형 형태이므로 발견적 기법을 개발하는 것이 필요하다. 본 연구에서는 이 모형을 Tabu Search를 이용하여 효율적인 해를 제시하는 해법을 개발하였다.

본 연구는 발견적 기법의 하나인 Tabu Search Method를 분산 처리 시스템에 적용하는 방법을 연구해 보고 그 효과를 알아보는데 목적이 있다. 2장에서는 본 연구의 수리적 모형을 설명하고 3장에서는 본 연구에서 개발한 알고리듬을 제시하고 4장에서는 실험 결과를 요약하였다.

2. 수리 모형

본 연구의 모형은 각 노드에서 발생하는 작업(job)을 처리하기 위해 발생하는 처리 비용과 통신 비용을 최소화하도록 컴퓨터와 데이터 파일을 배정하는 것이다[6].

2.1 시스템 설명

1) 컴퓨터의 종류는 성능에 의해 micro-computer, minicomputer, small mainframe, large mainframe 그리고 supercomputer의 다섯 가지로 나뉜다.
2) 시스템에 도착하는 작업은 그것을 처리하는 컴퓨터의 능력에 따라 다섯 가지의 등급으로 나뉜다.
3) 어떤 노드에 도착하는 작업은 노드에 그것을 처리할 수 있는 능력의 컴퓨터 혹은 상위 등급의 컴퓨터가 존재하면 그 노드에서 처리되고 존재하지 않으면 다른 노드로 보낸다. 작업을 다른 노드로 보낼 때 그 작업을 처리할 수 있는 노드들끼리는 그 작업을 받아들일 확률이 모두 동일하다.
4) 모든 노드에는 컴퓨터가 하나씩만 존재한다.

2.2 기호 및 용어의 정의

다음은 본 연구에서 사용되는 기호와 용어에 대한 설명이다.

d_{ms} : $\begin{cases} 1, & \text{노드 } s\text{에 등급 } m\text{의 컴퓨터가 할당된 경우} \\ 0, & \text{그 밖의 경우} \end{cases}$

X_{sk} : $\begin{cases} 1, & \text{노드 } s\text{에 } k\text{번째 파일이 존재하는 경우} \\ 0, & \text{그 밖의 경우} \end{cases}$

λ_s : 단위시간당 노드 s에 도착하는 작업의 도착률.

b_{sr} : 노드 s에 도착하는 작업 중 등급이 r인 작업의 확률.

q_{rk} : 등급 r의 작업이 파일 k를 갱신(update)할 확률.

p_{rk} : 등급 r의 작업이 파일 k를 조회(query)할 확률.

O_{rk} : 등급 r의 작업으로 파일 k를 갱신하는데 필요한 평균 전송 데이터의 크기(kilobytes).

t_{rk} : 등급 r의 작업으로 파일 k의 복사본을 조회하는데 필요한 평균 전송 데이터의 크기(kilobytes).

g_r : 등급 r 작업의 평균 작업량(처리(transaction) 수로 측정된다).

CP_m : 등급 m의 컴퓨터 상에서 a million machine instructions을 처리하는데 드는 비용.

α_r : 등급 r의 작업을 수행하는데 필요한 평균 전송량(kilobytes).

β_m : 등급 m의 컴퓨터에서 처리되는 평균 machine instruction 수(thousands of instruction으로 측정된다).

C : 1 kilobyte 전송하는데 드는 비용.

b'_{sr} : 단위 시간당 노드 s에서 처리되는 등급 r인 작업들의 평균 처리 회수.

L_s : 단위 시간당 노드 s에서 처리되는 작업들의 평균 작업량.

$$L_s = \sum_{r \in R} b'_{sr} g_r$$

Z_s : 노드 s에 있는 컴퓨터의 속도(millions of instructions per second로 측정한다).

$$Z_s = \sum_{m \in M} d_{ms} \beta_m \sum_{r \in R} b'_{sr} g_r \quad \text{for } s \in S$$

2.3 수리 모형의 설명

본 연구의 모형은 처리 비용과 통신 비용의 합을 최소화하는 목적식과 컴퓨터 배정, 데이터 파일 할당, 및 작업 능력을 고려하는 제약식으로 구성되어 있다.

$$Min. \left[\sum_{s \in S} \sum_{m \in M} d_{ms} \beta_m L_s CP_m + C^* \left[\sum_{s \in S} \lambda_s \sum_{m \in M} d_{ms} \sum_{r=m+1}^{R} \alpha_r b_{sr} \right.\right.$$

$$\left.\left. + \sum_{s \in S} \sum_{k \in K} \left[\sum_{r \in R} b'_{sr} t_{rk} p_{rk} + X_{sk} \sum_{r \in R} b'_{sr} (O_{rk} q_{rk} - t_{rk} p_{rk}) \right] \right] \right] \quad 1)$$

제약식

$$\sum_{m \in M} d_{ms} = 1 \quad \text{for } s \in S \tag{2}$$

$$\sum_{s \in S} X_{sk} \geq 1 \quad \text{for } k \in K \tag{3}$$

$$\sum_{r=r_i}^{R} \sum_{s \in S} \lambda_s b_{sr} g_r \beta_r \leq \sum_{r=r_i}^{R} \sum_{s \in S} d_{sr} \cdot Z_s \quad \text{여기서, } r_i\text{는 노드 } i\text{에 할당된 컴퓨터의 등급이다.} \tag{4}$$

목적식은 운영비용을 최소화하는 것인데 운영비용은 작업들을 처리하는 처리 비용과 작업의 이동, 데이터 파일 갱신 및 조회하는데 필요한 통신비용으로 구성된다. b'_{sr}은 d_{ms}에 의해 결정되는 함수이므로 식 1)에서 $X_{sk} \sum_{r \in R} b'_{sr}(O_{rk} a_{rk} - t_{rk} b_{rk})$ 부분은 비선형 함수가 된다.

2)번 제약식은 각 노드에 컴퓨터는 하나씩 할당되는 것을 나타내고 3)번 제약식은 각 파일은 적어도 하나의 노드에 저장되어 있어야 하고 복사본도 허용한다는 뜻이다. 4)번 제약식은 각 노드에 도착하는 다양한 종류의 모든 작업들을 처리하기 위한 충분한 처리 능력이 각 노드에 필요하다는 것을 나타낸다.

3. 해법 개발

3 장에서는 Tabu Search Method를 이용하여 분산 처리 시스템의 해법을 구하는 방법을 제시한다.

3.1 Tabu Search Method의 기본 개념

Tabu Search는 해를 탐색한 과정을 Tabu list에 기록해 놓고 되돌아가는 것을 금지함으로써 cycle을 방지하고 국부 최적해를 벗어날 수 있는 기법이다. 또한 Tabu list에 있더라도 이 해를 선택하면 더 좋은 해가 나올 가능성이 있는 해에 열망 수준(aspiration level)을 주어 Tabu list를 해제하고 이것이 새로운 해가 될 수 있도록 한다. 이 과정을 주어진 회수 동안 반복하면서 가장 우수한 해를 출력하는 기법이다[3].

3.2 Tabu Search Method를 사용한 알고리듬

Tabu Search의 이론 및 응용에 관한 활발한 연구에도 불구하고, 이 기법을 분산 처리 시스템의 할당 문제를 푸는데 적용한 연구는 없었다. 따라서 이 절에서는 Tabu Search를 일반적인 분산 처리 시스템의 할당 문제에 적용하는 방법을 각 단계별로 살펴보기로 한다.

3.2.1 초기해 설정

Tabu Search Method에서 초기해를 선정하는 방법으로 임의의 초기해로 부터 시작하는 방법과 최적해에 되도록 근사한 초기해를 구하여 여기서 부터 출발하는 방법이 있다.

본 연구에서 초기해를 구성하는 컴퓨터와 데이터 파일의 할당은 다음과 같이 한다. 임의의 노드에 들어오는 모든 등급의 작업을 검토한 후 그 노드에서 가장 많이 수행하는 작업의 등급을 기록하여 그것에 해당하는 등급의 컴퓨터를 할당한다. 마찬가지로 그 작업이 가장 많이 사용하는 데이터 파일을 그 노드에 할당한다. 파일의 경우 하나의 파일이 여러 곳의 노드에 할당

될 수 있으므로 어떤 파일을 추가해 보아서 목적 함수를 감소시키면 그 파일을 추가한다. 즉, 최초로 할당된 파일에 새로운 파일을 첨가해 보는 것이다. 본 연구에서 제시한 초기해를 지능적(intelligent) 초기해라 부르기로 한다.

3.2.2 이웃해의 생성 과정

이웃해는 컴퓨터의 배정에 관한 것과 데이터 파일의 배정에 관한 것이 있다. 이웃해의 생성은 컴퓨터의 배정에 관한 이웃해를 먼저 생성한 다음에 파일의 배정에 관한 이웃해를 생성한다.

노드

컴퓨터	1	2	3	4	5	6
1	0	0	0	0	1	0
2	0	0	1	0	0	0
3	0	1	0	0	0	1
4	1	0	0	0	0	0
5	0	0	0	1	0	0

[Fig. 1] 컴퓨터가 할당된 예

노드

파일	1	2	3	4	5	6
1	1	1	0	1	0	0
2	1	1	0	0	0	0
3	1	0	0	1	0	0
4	0	0	0	0	1	0
5	0	1	0	0	0	1
6	0	0	1	0	0	0
7	0	0	1	0	0	0
8	0	0	1	0	0	0

[Fig. 2] 데이터 파일이 할당된 예

i) 컴퓨터 배정의 이웃해 탐색 방법

[Fig. 1]을 사용하여 설명한다. [Fig. 1]이 컴퓨터 배정의 초기해라 하면 노드 1에 대해서 컴퓨터 할당을 바꾸어 본다. 즉, 현재 노드 1에 할당되어 있는 컴퓨터 4의 할당을 제거하고 컴퓨터 1을 할당하여 본다. 이런 과정을 나머지 컴퓨터에 대해서도 해준다. 이때 가장 좋은 값을 기록한다. 노드 1의 할당 상태를 원래대로 해 놓고 노드 2 ~ 6까지에 대해서도 마찬가지 방법으로 한다. 이 여섯 가지 중 목적함수를 가장 좋게 하는 것을 현재해로 선택하고 이것을 Tabu list에 기록한다.

ii) 데이터 파일 배정의 이웃해 탐색 방법

[Fig. 2]가 파일 배정의 초기해라 하면 파일 1에 대해서 파일의 할당을 바꾸어 본다. 데이터 파일 1이 노드 1에 할당된 것을 파일이 없는 노드 3과 5와 6에 차례로 할당해 본다. 할당하는 과정에서 비용이 감소되면 할당 상태를 계속 바꾸어 준다. 파일 1의 할당 상태를 원래대로 하고 파일 2 ~ 8에 대해서 마찬가지 방법으로 한다. 이 여덟 가지 중 목적함수를 가장 좋게 하는 것을 현재해로 선택하고 이것을 Tabu list에 기록한다.

3.2.3 분산 처리 시스템에 맞는 Tabu 제한

분산 처리 시스템에서는 각 노드마다 작업이 들어오는데 현실적으로 노드마다 가장 많이 처리하는 등급의 작업과 가장 많이 사용하는 데이터 파일이 있다. 각 노드에서 가장 많이 다루는 작업을 처리할 수 있는 컴퓨터를 그 노드에 할당하는 것과 가장 많이 참조하는 데이터 파일을 그 노드에 할당해 놓는 것이 현실 상황에 맞다. 이러한 문제를 Tabu 제한을 이용하여 효과적으로 처리하는 방법을 만들어 본다.

3.2.2절에서는 이웃해 생성 단계에서 구해진 현재해를 Tabu list에 기록하여 다음 일정 기간에는 다시 찾아 볼 수 없도록 해 놓았다. 이 현재해에 Tabu 제한을 하나 더 준다. 즉, 현재

해에서 할당된 컴퓨터가 그 노드에서 처리할 수 있는 작업의 처리량을 계산한다. 이것들 중 해당 노드의 작업을 가장 많이 처리하는 컴퓨터를 구해 낸 후 그 노드를 Tabu list에 기록한다. 데이터 파일의 경우에는 현재해에서 각 노드마다 할당된 데이터 파일이 있다. 각 노드에 대해서 가장 많이 참조되는 데이터 파일을 찾아내어 이 데이터 파일을 Tabu list에 기록한다.

3.2.4 알고리듬의 단계

본 알고리듬을 단계별로 기술하면 다음과 같다.

단계 1. [초기해의 구성]

1) 작업에 따라서 컴퓨터와 데이터 파일을 할당한다. 이것을 현재해와 최선해로 둔다.

2) Tabu list를 초기화시킨다.

단계 2. [해의 개선]

1) Tabu list에 없는 해와 Tabu list에 있지만 일방 수준을 만족하는 이웃해 중 가장 좋은 해를 현재해로 선택한다.

2) 현재해를 Tabu list에 기록한다.

단계 3. [반복]

주어진 회수 동안 단계 2를 수행하여 해를 개선시켜 나간다.

단계 4. [해의 출력]

구한 해 중 가장 좋은 해를 출력하고 종료한다.

4. 수치 예제

노드 3개, 컴퓨터 2 종류, 데이터 파일이 2개인 경우를 문제 A라 하고 노드 10개, 컴퓨터 5 종류, 데이터 파일 12개인 경우를 문제 B라 하고 노드 20개, 컴퓨터 5 종류, 데이터 파일 20개인 경우를 문제 C라고 한다.

본 연구의 알고리듬을 평가하기 위해서 문제 A의 경우에는 Tabu해를 전체 해 공간을 전부 검토하여 찾아낸 최적해와 비교하였다. 문제 B와 C에서는 최적해를 구하기 어려우므로 초기해를 임의로 했을 때와 초기해를 지능적으로 한 경우를 비교하였다. 또한 문제 C와 같은 크기의 문제를 10개 만들어서 Tabu 제한을 강화할 때의 효과를 알아본다.

본 연구의 알고리듬은 C 언어로 작성되어 IBM-PC 호환 기종 486-DX2 상에서 수행되었다.

문제 A의 최적해

Tabu Tenure를 1로 하고 최대 반복 회수를 50번으로 하여 수행한 결과 23,27,47번째에서 최적해와 같은 결과가 나왔다. 이 문제의 최적 목적식 값은 [Table 6]에 나와 있고 수행 시간은 0.77초가 걸렸다. 또한 최적해에서 노드 1에는 2등급의 컴퓨터가 할당되고 노드 2에는 1등급의 컴퓨터와 데이터 파일 1이 할당되며 노드 3에는 2등급의 컴퓨터와 데이터 파일 2가 할당된다. 문제 A에 관한 자료는 [Table 1] ~ [Table 5]에 나와있다.

[Table 1] 컴퓨터 관련 자료

	컴퓨터 등급	
	Micro	Main Frame
CP_m(\$)	0.01	0.03
β_m(thousands of instruction)	1.0	70.0

위의 표는 컴퓨터가 작업을 처리할 때 관련되는 비용이다.

[Table 2] 작업 관련 자료

	작업 등급	
	1	2
g_r	50.0	150.0
α_r(kio byte)	7.5	70.0

위의 표는 등급 r 작업의 평균 작업량과 평균 전송량에 관한 자료이다.

[Table 3] 작업의 도착률과 등급 확률

		노드		
		1	2	3
λ_s(per hour)		1500	1200	800
작업 등급	1	0.4	0.8	0.3
	2	0.6	0.2	0.7

위의 표는 작업의 도착률과 각 노드에 따라 발생하는 등급의 작업에 관한 자료이다.

[Table 4] 파일 관련 자료

		파 일	
		1	2
q_{rk}	1	0.2	0.8
	2	0.1	0.9
p_{rk}	1	0.5	0.5
	2	0.4	0.6
O_{rk}	1	3	10
	2	60	7
t_{rk}	1	1	5
	2	3	5

위의 표는 작업의 갱신과 조회에 관한 자료이다.

[Table 5] 통신 비용

	통신 비용
C(\$)	0.01

위의 표는 1 kilobyte 전송하는데 드는 통신 비용이다.

[Table 6] 문제 A의 결과

	Tabu 해	최적해
운영비용	\$ 810.18	\$ 810.18

4.1절에서는 문제 B와 C에 대해서 초기해를 변화하였을 때의 결과를 설명하고 4.2절에서는 문제 C와 같은 크기의 문제를 10개 만들어서 Tabu 제한을 강화할 때의 효과를 알아본다.

4.1 초기해의 변화

1) 종료 규칙(stopping rule)의 결정

문제 B와 C는 반복수 200 이후로는 더 이상 해가 개선되지 않았다. 따라서, 종료 규칙은 최대 반복수 200으로 한다.

2) Tabu Tenure의 결정

최대 반복 회수를 200으로 고정시켜 놓고 문제 B와 C에 대해 Tabu Tenure를 다르게 해 주었을 경우 가장 좋은 해는 다음과 같다.

[Table 7] Tabu Tenure에 따른 목적식값의 변화(문제 B)

Tabu Tenure	문제 B의 Tabu해	반복수
2	14698	3
4	14749	5
6	14686	167
8	14902	133

위의 표에서 반복수란 알고리듬에서 단계 2를 반복하는 회수를 말한다.

[Table 8] Tabu Tenure에 따른 목적식값의 변화(문제 C)

Tabu Tenure	문제 C의 Tabu해	반복수
5	23738	59
7	23269	74
10	23345	61
12	23559	63
15	23927	83
17	24096	122

3) 초기해를 다르게 했을 경우 수행 결과 비교

각각은 최대 반복 회수를 200으로 하고 100번 수행한 것 중 가장 좋은 결과를 뽑은 것이다.

[Table 9] Tabu Tenure에 따른 목적식값의 변화(문제 B)

	랜덤한 초기해			지능적인 초기해		
	초기해	최선해	반복수	초기해	최선해	반복수
운영비용	$ 22064	$ 14726	157	$ 15541	$ 14686	167

문제 B를 최대 반복수 200으로 한번 수행하는데는 12초가 걸렸다.

[Table 10] Tabu Tenure에 따른 목적식값의 변화(문제 C)

	랜덤한 초기해			지능적인 초기해		
	초기해	최선해	반복수	초기해	최선해	반복수
운영비용	$ 32680	$ 23460	171	$ 30849	$ 23269	74

문제 C를 최대 반복수 200으로 한번 수행하는데는 8분 20초가 걸렸다.

위의 결과를 보면 두 가지 문제에 대해서 지능적 초기해의 목적식 값이 더 작은 것을 알

수 있다. 그러나 최선해가 수렴하고 있으므로 본 연구의 알고리듬이 효율적이라는 것을 알 수 있다.

4.2 Tabu 제한 강화

3.2.4절에 있는 알고리듬의 단계 중 단계 2에서 구해진 현재해에 3.2.3절에 있는 방법으로 Tabu 제한을 준다. 문제 C와 같은 크기의 문제를 10개 만들어서 이것에 대해 Tabu 제한을 하나만 줄 때와 강화할 경우에 따라 실험을 해 보았다. [Table 11]에 있는 결과는 Tenure를 7로 고정하고 최대 반복 수를 200으로하여 20번 푼 것 중 가장 좋은 결과를 뽑아 놓은 것이다.

[Table 11] Tabu 제한에 따른 결과 비교

data		Tabu 제한이 하나일 때		Tabu 제한을 강화한 경우	
		최소 비용	반복수	최소 비용	반복수
skewed input data	C-1	$ 4397448	137	*$ 4357036	128
	C-2	*$ 256037	62	$ 256128	192
	C-3	$ 73436	168	*$ 72789	128
	C-4	$ 67118	93	*$ 65985	41
	C-5	$ 235812	179	*$ 234902	155
random input data	C-6	$ 23269	74	*$ 23129	57
	C-7	*$ 28328	75	$ 29247	41
	C-8	$ 28691	106	*$ 28337	109
	C-9	*$ 31092	181	$ 31323	17
	C-10	$ 22213	200	*$ 22033	49

이 실험의 결과로 볼 때 본 연구의 데이터의 경우에 있어서는 Tabu 제한을 강화하는 것이 좋고 데이터가 특정하게 몰려있을 경우에는 특히 더 Tabu 제한을 강화할 필요가 있음을 알 수 있다.

5. 결론

본 연구에서는 Tabu Search Method를 분산 처리 시스템의 할당 문제에 적용하는 방법에 대해서 연구하였다. 본 연구에는 초기해 및 이웃해 생성 방법을 개발하였고 Tabu 제한을 변화시키면서 실험 결과를 분석하였다.

본 연구의 결과 지능적 초기해의 경우 목적식 값이 적지만 지능적 초기해 및 임의의 초기해 모두의 최선해가 수렴하고 있음을 보여주므로 본 연구의 알고리듬이 효율적이라는 것을 알 수 있다. 또한 실험 결과 Tabu 제한을 강화하는 것이 좋고 데이터가 특정하게 몰려있을 경우에는 특히 더 Tabu 제한을 강화할 필요가 있음을 알 수 있었다. 본 연구에서 사용한 Tabu Search Method는 작은 크기의 문제(문제 A)에 대해서는 최적해를 구하였고 현실적인 크기의 문제(문제 B와 문제 C)에서도 짧은 시간안에 해를 구하였고 해의 수렴성면에서도 효율적이라고 볼 수 있다.

그러나 이웃해를 생성할 때 삽입에 의해서만 하였으므로 교환, 삭제 그리고 첨가의 방법으로 이웃해를 구해보는 것이 필요하다. 연구 결과로 Tabu Search는 분산 처리 시스템의 할당 문제를 결정하는데 효과적인 것으로 밝혀졌다.

참 고 문 헌

1. Anup Kumar, Rakesh M. Pathak and Yash P. Gupta, Genetic Algorithm Based Approach for File Allocation on Distributed Systems, *Computers Ops. Res.*, Vol. 22, No. 1, pp 41-54, 1995.
2. Bezalel Gavish, Optimization Models for Configuring Distributed Computer Systems, *IEEE Transactions on Computers*, Vol. C-36, No. 7, July 1987.
3. Colin R. Reeves, *Modern Heuristic Techniques for Combinatorial Problems*, John & Sons, INC. New York Toronto, 1993.
4. Deb Ghosh and Ishwar Murthy, A Solution Procedure for the File Allocation Problem with File Availability and Response Time, *Computers Ops. Res.*, Vol. 18, No. 6, pp 557-568, 1991.
5. Heeseok Lee and Olivia R. Liu Sheng, A Multiple Criteria Model for the Allocation of Data Files in a Distributed Information System, *Computers Ops. Res.*, Vol. 19, No. 1., pp 21-33, 1992.
6. Hemant K. Jain, A Comprehensive Model for the Design of Distributed Computer Systems, *IEEE Transactions on Software Engineering*, Vol. SE-13, No. 10. pp 1092-1104, Octbor 1987.