〈論 文〉

階層構造 Computer Network에서 工程制御를 爲한 JOB Scheduling

JOB Scheduling for process Control in Hierarchical Computer Network

朴 逸*
(Park Yil)

(접수일자 80. 12. 10)

要 約

階層構造 COMPUTER Network을 通한 工程制御로 Processing을 分散하여 Fault tolerance를 極大化시키며 複雜하고 多樣한 變數의 相互關係를 週期的으로 監視制御하는 分散制御 Processor JOB은 그 週期와 實行時間으로 定義할 수 있다. 모든 JOB에 대하여 Tree structure인 關係를 가진 subset들로 구성하여 이에 JOB Scheduling Algorithm을 求하여본 결과 FCFS(First Come/First Service)인 Schedule보다 Processor의 利用에 있어 Loose Time을 減小시키고 處理 可能時間 確保에 有利하였다.

—Abstract—

The distributive processing job in a hierarchical computer network, which supervises and controls the complex relations between the variables periodically for raising the fault folerance, can be defined its periodicity and its execution time. All the job may be composed of the subsets in relation of Tree structure. For a processor job set this paper finds out a job scheduling algorithm that has the less loose time between period than that of FCFS.

Ⅰ. 序 論

Processor control system은 各 工程으로부터 發生되는 尨大한 量의 情報를 分析하고 最適指令을 下達하며 適節한 Point로 System을 安定시키기 爲하여 監視制御機能을 Section化 하며 危險負擔을 줄이고 經營과 現場의 距離를 좁히며 各 階層別로 Function을 分擔할 수 있는 multilevel Hierarchical Network가 적합하다. 이를 구성하는 各 processor는 System을 効果的으로 使用하기 위하여 Real Time Schedule이 필요하다.

FCFS(First come/First Service) 또는 Round Robin service 등의 JOB Scheduling에 있어서

① FCFS에서는 JOB order가 Random하여 idle Time으로 인한 Loose Time을 최소화하는데 정확한 해석이 곤란하다. JOB關係가 獨立的이면 Level 1 processor에서는 좋은 Service가 되지 못한다.[3]

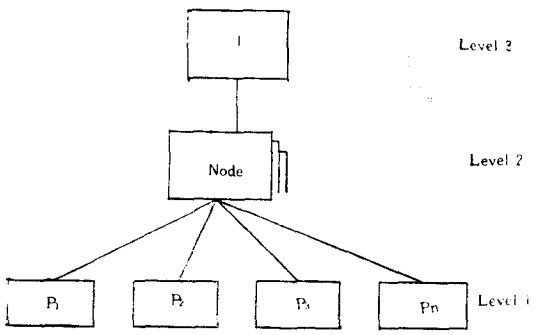
③ Round Robin Service인 경우 Overflow에 대하여는 좋은 解決方式이나 工程制御의 目的에는 적합하지 않다. Task의 實行時間이 同一한 경우는 〔8〕에서 이미 論하였다. Single processor에서 Job Scheduling의 최소 Settling time은 실행 순서가 어떻게 되느냐에 상관없이 전체 Task의 총 실행시간이 된다. Multi processor에서는 Settling time[3]의 최소화를 위한 JOB 상호관계와 processing에 있어 必要한 事項을 提示하고 있으나 具體的인 Algorithm은 아직 미정인 상태이다. 本 論文에서는 multiprocessor의 Element인 개별

* 東洋工專 正會員

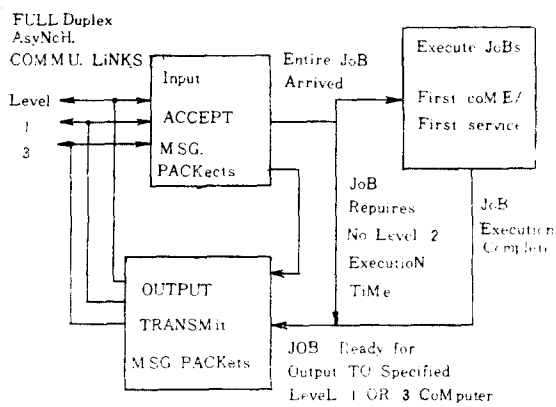
processor의 대상 JOB에 대하여 processor 전체의 Settling time에 문제를 두지 않고 각 processor의 이용가능 범위를 확대하기 위한 JOB 상호 관계를 설정하고 이의 屬性을 定義하기 爲한 Algorithm을 論하고자 한다.

Ⅱ. 階層構造網의 特性

階層의 頂上은 根低 Level에서 얻은 資料를 토대로 意思를 決定하고 各 階層은 Function을 Module 化하여 工程의 變化 擴張時 該當 Module을 變動시킨다. 低 Level processor는 Real Time으로 動作하기 때문에 엄격한 時間制約을 받지 않을 수 없는 고로 遲延없는 Response를 爲해서 Level 1 processor는 過負荷가 되어서는 안된다. 계층구조 Computer Network은 그림〔1〕과 같다.



〔**Fig 1**〕 Hierarchical Computer Network



〔**Fig 2**〕 Behavior of Level 2 processor

低 Level로부터 2nd Level까지는 直接的인 CONTROL을 하며 Level 1에서의 入力 Data는 Level 1 이상으로 傳送한 후 蓄積시킨다. Level 2 Computer는 (그림 2) Level 1 보다 强力하여야 하고 보다 많은 情報處理機能으로 資料의 仲介, 蓄積, 命令의 下達等을 한다. 工程制御 Computer Network에서 Module間 Communication 方式은 大型 Computer의 Selector channel이나 Multiplexor channel대신 Communication을 爲한 單一週邊機器로 構成하고, Data 傳送 速度는 9600 Baud 이상이면 工程制御用으로는 充分한 能力이다. Level 2에서의 情報交換은 보다 制限的이 되지 않으면 안된다. 이것은 Level 1의 processor 들로부터 同時에 發生한 경우가 있을 수 있기 때문이다. 9600 Baud 이상이면 1ms의 handling time이 되어 Interrupt Mode로 處理될 때 Operating system에서는 最小 200 μs가 소요된다. 同時에 Line이 겹쳐서 發生하면 情報를 忘失할 우려가 있으며 이런 경우 한쪽 Line은 무시하고 次後 再傳送을 기다리거나 Microprocessor를 利用한 Line controller로 Sequential 處理를 하여야 한다. Message의 送受는 CPU의 負荷에 아무 영향을 끼치지 않으며 packet當 Interrupt는 送受의 最終段階에서만 발생한다.[4] Engelberg는[4] 계층구조 Network에 대해서 다음과 같이

1. Processor 간의 Communication
2. Network JOB으로 구분하여 Function을 定義했다.

2—1. Network 內 Communication

① Processor 간 Communication은 interrupt를 발생시키므로 Asynchronous 方式으로 行한다.

② Communication Message는 packet로 구성한다.

③ 다음 packet가 發生되기 前에 認知 Operation이 完了되어야 한다.

④ Full duplex 方式으로 上下 어느 Level로도 可能하여야 한다.

⑤ 送受를 同時에 하는 processor는 各 packet를 취급하는 시간이 반드시 重復되도록 한다.

2—2. Network JOB

① JOB은 Level 1, Level 3 processor 어느 곳에서도 起動이 可能하고

② JOB은 그 決果가 보내져야 하는 processor(target computer)를 確定하며 target computer와 關連 priori를 가진다.

③ 有限個의 Message Packet는 target computer로 보내는데 1 Level 이상 經由시키려면 中間 Level에서 收容한 후 再傳送한다.

④ Target computer는 一定時間內에 JOB을 處理하고 이 決果를 最初 processor로 Return시킨다.

⑤ JOB은 Queue에서 待期한 後 遂行된다. 물론 實行時間이 必要치 않으면 Queue에 들어가지 않는다. 入力 Data의 到着時間과 Service 時間이 서로 獨立的이면 M/M/1 Queuing 體制로 보아 Level 1 processor의 稼動確率 P_n는

$$P_n=(1-\rho)\rho^n \quad n=0,1,2,\cdots$$

이다. 이때의 平衡상태는 $\rho<1$ 이 되어야 한다. 즉 到着率이 Service 率보다 적어야 統計的인 平衡을 이루고 다음 週期가 開始되기 前에 Queue에 들어온 JOB이 終了될 때 平衡이다. (ρ : 活用率 또는 稼動率).[9] 附加하면 Scheduling과 Queuing theory는 다르다. 兩者의 差異는 決定變數의 性質에 있다고 하겠으며 後者인 Queuing theory는 到着率 또는 Service 率에 대한 決定이 主가 된다.[11]

⑥ Network의 I/O communication JOB은 遂行上 priority가 있다.

⑦ Level 2, 3는 FCFS Scheduling Algorithm으로 Service하는 것이 Round Robin이나 SJF(Shortest job First) 보다 Network의 Workload 特性이 우수하다.[3]

Ⅲ. Processor의 制限事項

各 Level processor의 JOB 처리는 起動시킨 Level의 processor로부터 target processor 사이에 發生한 Event이다. 工程制御에서의 target processor는 最初 processor에 依하여 定해지므로 이에 따른 JOB Schedule은 한쪽의 Schedule에 依해 一意的으로 定해진다. 따라서 Level 1 processor의 Schedule이 문제가 된다. Level 1의 임의의 processor P_i에서 JOB J_i는 1회 처리 최대소요시간 E_i, 주기 T_i, 일때 $J_i:(T_i, E_i)$, $1\leq i<N$이다. 최대한 processor의 이용범위를 규정하기 위해 다음 사항을 가정한다.

① processor 내에서 JOB은 서로 獨立的이며 終了와 起動은 關係가 없으므로 終了와 起動을 一致시켜 Loose time을 最小化하도록 한다.

② JOB은 相互관계가 없고 Nonpreemptive하므로 priority는 부여하지 않는다.

③ Level 1 processor의 安定된 JOB service를 하기 위해서 동일 processor 內에서 $Ei>T_i$인 JOB은 processor JOB Set의 Element를 構成하지 않는다.

④ $E_i\leq T_i$인 JOB들의 週期 T_i를

$$T_2=2T_1$$
$$T_3=2T_2=4T_1$$
$$\vdots$$
$$T_i=2T_{i-1}=2^{i-1}T_1 \quad (i>1)$$

인 JOB Set로 구성한다. 그 一例로 표 1에 JOB을 提示했다.

[3] Example of Jobs

	실행시간	주 기	Freq.
J_1	1	4	1/4
J_2	2	8	1/8
J_3	1.5	16	1/16
J_4	3	32	1/32
J_5	4	64	1/64

모든 JOB은 위와같은 關係를 가진 Subset들로 구성된 집합으로 表示할 수 있으므로 이를 대상으로 Scheduling은 하려한다. Nonperiodic Task로 구성된 JOB Set인 경우 minimum processor 數를 구하고 이의 最短處理時間을 求하는 方法은 [5]에서 提示하고 있다. 또한 주어진 2개의 processor에 대하여 最短時間內에 完了할 수 있는 方法은 Johnson's Rule을 適用하여 最適解를 求할 수 있다.[11]

Ⅳ. Schedulig Approach

4—1. JOB TiME

例로 든 J_1과 J_2에서

$T_1-E_1=\phi_1$ 이라하면 $\phi_1\geq E_2$이므로 JOB J_1과 J_2는 同一 processor로 遂行이 可能하다. 그러나 E_2가 增加하여 ϕ_1보다 커졌을 때는 J_1의 다음 週期로 overflow가 된다. J_1과 J_2가 同一 processor JOB이 되기위하여

$$E_i\leq T_1-E_1=\phi_1$$

을 만족하여야 한다. 그림 [4], [5]에서 그 결과를 보인다. J_i를 어떤 時點에서 P_i에 割當된 JOB Set라 하면 J_1과 J_2의 例와같이 이 JOB은 periodic하므로 이의 idle Time도 週期를 가진다. JOB은 처리 週期 Ti가 길수록 짧은 JOB보다 處理頻度가 낮아져서 JOB Set 구성시에 priority를 割當한 것과 같은 意味를 내

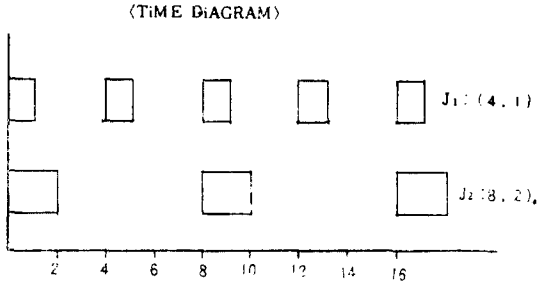
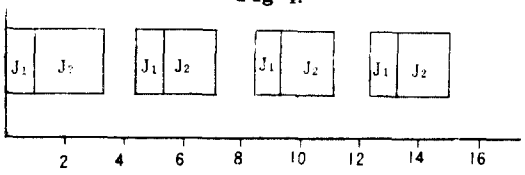


Fig 4.



1. Processor JOB(J_1, J_2)

Fig 5.

포하고 있다.

즉 $J=\{J_i\}$ 이고 $E_i \leq \phi_1$

$\forall i \in [2, N]$ 이다.

4—2. JOB Set의 periodicity

최초 J_1이 P_i에 割當된다면

$T_2=2T_1$ T_2 Sec 마다 $2\times\phi_1$

$T_3=2T_2$ T_3 Sec . . $2\times\phi_2$ $(=4\phi_1)$

$\vdots$

$T_n=2T_{n-1}$ T^n Sec .. $2\times\phi_{n-1}$ $(=^{n-1}\phi_1)$

이 使用 可能 idle Time이 된다. 그러므로 $J_1 \cap J_i$의 Set는 overflow되지 않고 T_1 내에서 滿足할 수 있는 Schedule을 가진다. J_1과 J_2로된 J_1의 Execution Time을 ϵ_1라 하면

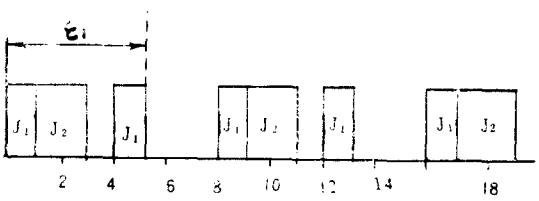


Fig 6. $\{J_1, J_2\}$의 TiME DiAGRAM

$\epsilon_1=T_1+E_1$

JOB의 조합에 따른 週期에서의 idle Time $T^i{}_j$는 JOB T_j가 P_i에 할당된 J_i의 idle Time을 뜻한다.

$T^i{}_j=\phi_i-E_j$ $j\in[2, N]$

가 된다.

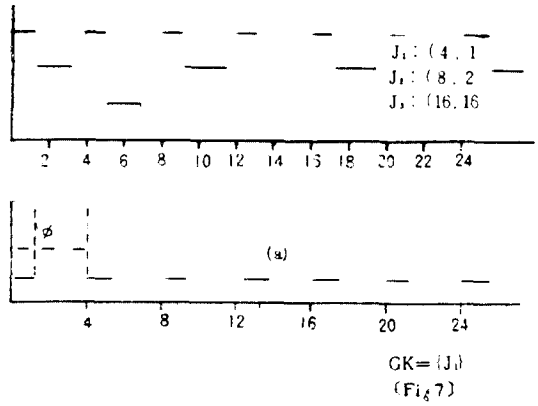


Fig 7.

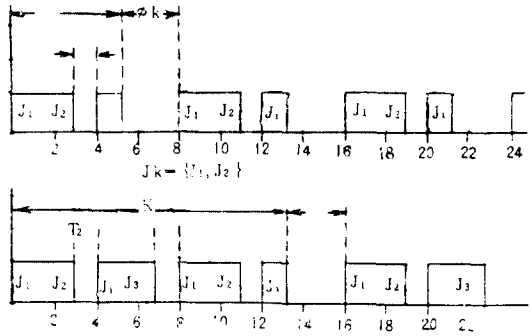


Fig 8. $J_k=\{J_1, J_2, J_3\}$
JOB Set J_k. TIME DIAGRAM

$T^k{}_2$: J_1에 J_2가 組合된 idle Time

$T^k{}_3$: J_1에 J_3가 〃

$T^k{}_1=0$, $T^k{}_2$, $T^k{}_3$. periodic 하다.

J_1과 J_3가 J를 구성하면

$$\epsilon_k=4E_1+2E_2+E_3+2T^k{}_2+T^k{}_3$$
$$=4E_1+2E_2+E_3+2(\phi_1-E_2)+(\phi_1-E_3)$$
$$=4E_1+2E_2+E_3+2(T_1-E_1-E_2)+(T_1-E_1-E_3)$$
$$=3T_1+E_1$$
$$=13$$
$$\phi_k=T^k{}_3=3$$
$$\therefore T=\epsilon_k+\phi_k$$
$$=13+3=16$$

J_1과 J_3에서는 (그림 8에 도시) $E_3 \leq \phi_1$을 만족해야 한다.

$T^k{}_j$는 Duration이 ϕ_1이고 最終 부가된 JOB의 週期가 이 JOB Set의 週期가 됐다는 것을 確認했다. 한편 $E_i \leq \phi_1$, $i\in[2, N]$인 關係에서도 分明하다.

V. Algorithm

① 最初에는 J_i의 割當이 없는 상태이다.

$J_i=\{\ \}\ |\phi_k|=\infty$

JOB J_i가 J_l에 割當되면 $\phi_l=\phi_j$

② $E_i\leq\phi_l$인 JOB J_i를 J_l에 追加

③ $E_i>\phi_l$인 JOB J_n은 J_n. 割當

全 processor JOB $J=\{J_i\}$

$J_1=\{J_i\}$

$J_2=\{J_j\}$

$\vdots$

$J_m=\{J_k\}$

가 되고 다음 條件들을 만족한다.

① P_i에서 Overflow는 없다. $V_i\in[1,N]$

② P_i 內 JOB은 서로 排他的이다.

③ $V_i\neq j$ $J_i\cap J_j=\phi$ (Disjoint Relation)

④ $J=\{J_i\}$, $\overset{m}{\underset{i=1}{\cup}} J_i=J$

먼저 例示한 J_1, J_2, J_3, J_4, J_5에 대하여 適用한 結果 아래와 같은 Gantt chart로 나타났다.

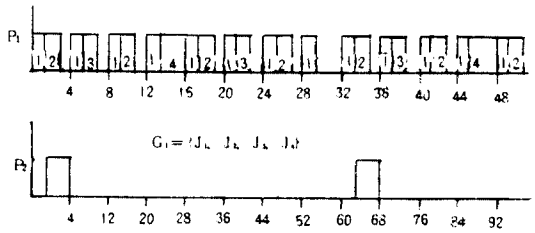


Fig 9. $J_2=\{J_5\}$
$J=\{J_1, J_2\}$ schedule chart

VI. 結 論

Level 1 processor의 JOB을 Real Time으로 處理하기 위하여 이를 shift하지 않으면 同時에 발생될 수 있는 最大 JOB수와 同一하게 precessor를 확보하여야하나 위의 例에서와 같이 2개의 processor로가능하였으며 P_1의 Loose Time 감소, P_2의 實效 availabe Time 증가, 다른 processor 이상시 처리 불가인 JOB 수를 최소화했다. 그러나 JOB scheduling시 본 논문에서와 같이 개별 processor의 Loose Time을 最小化하느냐 또는 전 processor의 settling Time을 최소화하느냐 하는 것은 사례별로 연구하여야 할 대상이다.

參 考 文 獻

1. Ashenhurst, R.L VANderohe, R.H
"A Hierarchical Network" Datamation Febr 1975p.40~44
2. C.V. Ramamoorthy K,M. chandy and Mario. Gonzalez, Jr "Optimal Scheduling Strategy in a multiproessor System" IEEE Trans. Comp. Vol C-21 No 6 1972 p.137~146.
3. George p. Engelberg, James, A. Howard and Duncana, Melichamp "JOB Scheduling in a Single-Node Hierarchical Network for process Control" IEEE Trans. Comp. Vol C-29 No 8AUGUST 1980
4. George. p. Engelberg, James. A. Howard and Duncama. Mellichamp "A real time Simmulation of a hierarchical computer Network for process Control" Simmulation p.37~49. 1978.
5. Edurdo B. Fernandez and Bertram Bussell "Bounds on the number of processors and Time for multiprocessor optimal schedules" IEEE. TRANS. COMP. Vol. C-n No 8 August p.745~p.751. 1973.
6. D.D. Chamberlin, S.H. Fuller and L.Y. Liu "An Analysis of page allocation Strategies for multiprogramming systems with virtual memory" IBM. J. RES. Develop p.404~412. 1973.
7. JAN welgarz, "Multiprocessor scheduling with Memory Allocation-Aderministic Approach" IEEE. TRANS. COMP. Vol.C-29 No 8 August 1980.
8. T.C. Hu. "Parallel Seguencing and assembly line problems" Oper. Res. Vol. 9 p.841~848. Nov. 1961.
9. 金吉昌 : 運營體制
10. Donovan: Operating System
11. 金海天, 高延燮, 池靑 '經營意思決定論'