システムソフトウェアと
オペレーティング・システム 82－8
（１９９９．８．２）

ソフトリアルタイムにおける適合型CPUスケジューリングポリシ

滝沢　泰久

（株）ATR環境適応通信研究所
〒619-0288　京都府相楽群精華町光台2-2
TEL　0774-95-1535　　E-mail　takizawa@acr.atr.co.jp

あらまし　動画や音声に代表される連続メディアを扱うネットワークやアプリケーションにおいてQoS制御が多く研究されている。これらQoSを提供する機能をより正確に実行するために、それらの周期的な時間特性と相互に通信を行う依存関係に動的に適応するCPU管理方式が重要となる。本報告ではPDPモデルを用いた適応型CPUスケジューリングポリシについて提案し、そのシュミレーション結果を述べる。

キーワード　スケジューリングポリシ　適応

A Adaptive CPU Scheduling Policy for Soft RealTime

Yasuhisa TAKIZAWA

ATR Adaptive Communication Research Laboratories
2-2 Hikaridai, Seika-cho, Soraku-gun, Kyoto 619-0288, Japan
TEL 0774-95-1535　E-mail takizawa@acr.atr.co.jp

Abstract　This has prompted much research into QoS control for network and application that treat video and audio. It is important that CPU management adapts to temporal property and dependence of application for countinuous media , because QoS control needs to excute exactly. In this report, we propose Adaptive CPU Scheduling Policy with PDP model and describe simulation results for proposal scheduling policy.

Keywords Scheduling Policy　Adaptation

1．はじめに

動画や音声に代表されるマルチメディア情報を処理環境に適応した品質（Quality of Service:QoS）で送受信する研究が近年盛んに行われている。それらの多くはネットワーク環境へ適応したQoSの研究[1][2]、またはアプリケーション特性へ適応したQoSの研究[3][4]として実施されている。しかしエンドーエンドで環境に適応した品質を提供するためには、上記の２機能に加え、実行タスクのコンピューティング資源要求特性に適応したスケジューリング方式を提供する必要がある。

現在のオペレーティングシステム（以下OS）は

・ビジネスアプリケーションを実行目的とする商用 OS

・リアルタイムアプリケーションを実行目的とするリアルタイム OS

の２つに大別できる。前者のスケジューリング方式は動的な資源要求に対応しているが、要求の時間特性を考慮していない。一方、後者のスケジューリング方式は資源要求の時間特性を考慮しているが、既知の環境で可能である。また両者ともに資源要求に対して実行タスク間の協調／抑制は考慮されていない。このような実行環境では、

・任意のタスクが任意の時間に生成消滅する。

・タスク間に依存関係がある。

・タスクの資源要求は周期的時間特性をもつ。

ようなマルチメディアアプリケーションを正確に実行することが出来ない。

本報告では、上記問題点を解決するため実行タスクの時間特性およびタスク間の協調／抑制に適応するOSのCPU管理方式、適応型CPUスケジューリングポリシを提案し、そのシュミレーション結果を述べる。

2．適応型 CPU スケジューリングポリシ概要

2．1　特徴

提案スケジューリングポリシは以下の特徴を持つ。

・ソフトリアルタイムスケジューリング
マルチメディアタスクにおける周期的タスクをスケジューリング対象とし、タスクの時間特性として実行周期幅(最小周期／最大周期)を入力パラメータとする。

・タスク時間特性への適応
タスクの時間特性として実行周期はタスクの特性に依存するが、周期内のCPU使用時間は実行環境のハードウェアに依存する。このことから、周期内CPU使用時間はタスクから入力されるのではなく実行環境のハードウェアに則してスケジューラで割当てる。

・タスクの依存関係への適応
タスク間の依存関係をタスク間の通信リンクにより構成し、通信を互に行うタスクは通信待ち時間を小さくするため同時に実行する可能性を高くするように実行周期幅を制御する。

2．2　構成

提案スケジューリングポリシは以下の機能から構成される。(図１)

・アドミションコントロール
タスクからの入力パラメータである最小周期、最大周期とモニタからのフィードバックである周期内実CPU使用時間からスケジュール可能判定、CPU使用率予約および初期CPU使用時間割当てを行う。

・アダプテーション
タスクの依存関係およびモニタからのフィードバックである実周期、実CPU使用時間、通信待ち時間から実行周期をその最小周期、最大周期間で局所最適な周期を算出する。

・タスク選択
EDF（Earlist Deadline First）[5]およびSRP（Stack Resource Policy）[6]の変形方式により実行タスクを選択する。

・モニタ
実CPU使用時間、実周期、通信待ち時間をアダプテーションおよびアドミッションコント

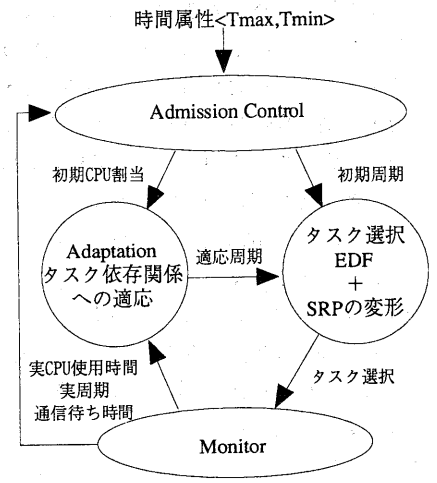


図1　適応型CPUスケジューラの構成

ロールへのフィードバックとして与える。

2．3　アドミッションコントロール

アドミッションコントロールではスケジュール可能であれば、CPU使用率を予約する。予約するCPU使用率はモニタからのフィードバックである実CPU使用時間を最小周期で割った値とする。つまり、タスクの最大CPU使用率を予約する。

$$R_i = \frac{C_i}{T_i^{min}} \qquad (1)$$

しかし、初回周期では実CPU使用時間がモニタから与えられてなく、不明である。そのため、初回周期に割当てるCPU使用時間はタスクの最大周期内で利用可能な（他のタスクに予約されていない）CPU時間とし、

$$C^{rest} = T_k^{max} - \sum_{i=1}^{n} \left\lceil \frac{T_k^{max}}{T_i^{min}} \right\rceil C_i \qquad (2)$$

T_n^{max} はタスクnの最大周期
T_n^{min} はタスクnの最小周期
C_n はタスクnの実CPU使用時間
T_k^{max} はタスクの中で最大の周期

$$C_{n+1}^{init} = \frac{C^{rest}}{\left\lceil \frac{T_k^{max}}{T_{n+1}^{min}} \right\rceil} \qquad (3)$$

により初期CPU使用時間を求める。

スケジュール可能判定は次のように行う。先ずは求められた初期CPU使用時間より実行を開始する。1周期実行するとモニタから実CPU使用時間が与えられるので、この時間が初期CPU使用時間内であればスケジュール可能判定し、実CPU使用時間で式(1)によりCPU使用率を予約する。初期CPU使用時間をこえる場合、スケジュール不能としタスクを中断する。

2．4　アダプテーション

アダプテーションではタスクの依存関係によりタスクの周期を最大周期と最小周期間の局所最適な周期（以下適応周期）に設定し、実行機会の制御を行う。

2．4．1　タスク相互結合ネットワーク

タスクの依存関係は各タスクのリンクによる相互結合型の並列分散処理（PDP）ネットワーク[7]として見立て

- ネットワークの各ノードの活性度はタスクの重要度（0以上1以下）とする。
- タスク間で通信する場合は協調関係とし、タスクは相互に重要度が高くなるようにタスク間のリンクに正の重み付けをする。
- タスク間で通信を行わない場合タスクは相互に抑制するとし、タスク間のリンクに負の重み付けをする。

により構成する。（図2）

2．4．2　適応周期更新則

PDPネットワークの活性化規則に確率的要因を取り入れ、タスクの適応周期を

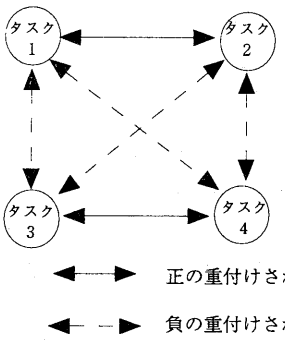


図２タスク１と２が通信、タスク３と４が通信する場合のタスク相互結合ネットワーク

$$a_i(t+1)=(1-\theta)a_i(t)+\begin{cases}(1-a_i(t))net_i(t) & prob(\Delta E_i(t))>K\\ a_i(t)net_i(t) & else\end{cases} \quad (4)$$

$$net_i(t)=\sum_{j\neq i}^{n}\omega_{ij}a_j(t)+bias_i(t) \quad (5)$$

$$prob(x)=\frac{1}{1+\exp\left(\frac{-x}{Temp}\right)} \quad (6)$$

$$T_i(t)=T_i^{max}+\left(T_i^{min}-T_i^{max}\right)a_i(t) \quad (7)$$

$a_i(t)$は時間tのタスクiの重要度（$0\leq a_i(t)\leq 1$）

θは活性度の緩和率（$0\leq\theta\leq 1$）

ω_{ij}はタスクiとタスクjの結合の重み

$bias_i(t)$は時間tのタスクiにかかる外部入力

Kは[0,1]間の一様乱数

$\Delta E_i(t)$は$(1-a_i(t))net_i(t)$

$T_i(t)$は時間tのタスクiの適応周期

として、タスクの到着時にタスク毎に非同期に更新規則を適用する。

式(4)(5)(6)(7)から適応周期更新則は

・次のタスクの重要度は緩和減少した現在の重要度に、ある確率で重要度が高くなる方向（式(4)の右辺第２項の上式）へ力が加わるか、または重要度が低くなる方向（式(4)の右辺第２項の下式）へ力が加わり求められる。

・タスクに加わる力は他のタスクの重要度とそのタスクとの結合の重みの積の和（式(5)の右辺第１項、協調関係のタスクが重要度が高い場合は大きな正の値となり、抑制関係のタスクの重要度が高い場合は大きな負の値となる）、および外部から力（式(5)の右辺第２項）により構成される。（図３）

・確率はボルツマン関数を用いる。

・タスクの重要度を適応周期に１次変換する。（重要度が１の場合は最小周期、重要度が０の場合は最大周期）

を数式化している。

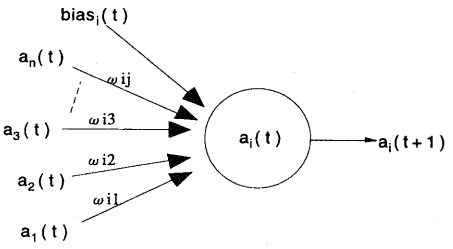


図３　他のタスクから力と外部入力

本提案方式では外部入力biasはモニタにより与えられる実周期とアダプテーションにより算出された適応周期の差および通信待ち時間とし、

・適応周期が実周期より長い場合は、タスクが必要以上に抑制されている状況と判断し重要度を高くなるように正の値をとる。

・適応周期が実周期より短い場合は、タスクが必要以上に活性されていると状況と判断しを重要度を低くするように負の値をとる。

・通信待ち時間がある場合は、CPU使用時間が増加したと判断し、CPU時間を稼ぐように周期を最大周期に近付ける。このため重要度を低くするよう負の値となる。

２．４．３　評価関数

タスクの正または負の重みを持つリンクにより

構成される相互結合ネットワークのエネルギーは

$$E(t) = -\sum_i \sum_j \omega_{ij} a_j(t) a_i(t) - \sum_i bias_i(t) a_i(t) \quad (8)$$

となり、これを評価関数とし、最小とする状態を見つける。

式(8)はタスクの結合が正の重みを持つ場合は互いに重要度が1（最高重要度）になると小さな値となり、またタスクの結合が負の重みを持つ場合少なくとも1つが重要度が0（最低重要度）になると小さな値となる。つまり、協調関係のタスクは互いに重要度が1（適応周期が最小周期）となる状態、抑制関係の場合は重要度が0（適応周期が最大周期）となる状態を探し出すことになる。

式(4)を繰り返し操作し、活性度を更新すると式(8)は局所最小状態へ至る。

2．4．4　ネットワークの安定／不安定制御

ネットワークは式(4)を繰り返えし操作すると局所安定になる。しかし、安定状態に至ると外部環境の変化に鈍感になり、大きな外部入力を伴わないと状態を移動しようとしなくなる。

本提案方式では、短い周期でタスクの実行状況が変わる環境に適応するため、ネットワークの状態を極小な安定状態にはせず、安定状態の近傍を浮動している準安定状態と外部環境の変化に従い新しい準安定状態を探している検索状態（不安定状態）の2つの状態を移動する。

上記の状態を制御するため、次のようにネットワークの状態に関するパラメータを制御する。

・安定状態からの距離
　協調関係のタスクがすべて重要度1となり、抑制関係のタスクはすべて重要度が0となると式(8)は最小となる。このような状態をネットワークの安定状態とし、現在の状態との距離を計算する。

・確率関数の温度 Temp
　式(4)はある確率で安定状態に向かうか、不安定な状態に向かうかが決まる。確率は式(6)から分かるようにネットワークの変化量とその時の温度 Temp によりきめられる。この温度を安定状態からの距離に従い、近傍にある場合は減少させ、遠く離れている場合は増加させる。つまり、安定状態の近傍では安定方向にし、離れている場合は新しい状態を探すため不安定（動き回る）状態にする。

・緩和率 θ
　式(4)の緩和率を大きな値にするとネットワークは不安定な状態となる。このことにより、外部入力が大きく正の方向または負の方向へ向けるように働く場合、緩和率を増加させる。つまり、外部入力量が大きいことは環境が変動したかまたは適応が不十分の場合であるので、緩和率を大きくし新しい状態を検索させる。

2．4．5　タスクへの適応動作

前述の適応操作を行うと次のような適応動作をしめす。

・重要度の高いタスクと協調関係にあるタスクは重要度が高められ、つまり適応周期が最小周期に近づき、タスク選択のEDFにより実行機会が協調関係タスクと同時に高くなる。

・重要度の高いタスクと抑制関係にあるタスクは重要度が低くなり、つまり適応周期が最大周期に近づき、タスク選択のEDFにより実行機会が低くなる。

・外部環境の変化生じた場合は、外部入力量が大きくなるため、緩和率が高くなり不安定状態となる。さらに安定状態からの距離も大きくなることから温度が増加し、より強く新しい状態を検索するようにを動き回る。

・色々な状態を動き回るうちに、安定状態の近傍に行くと温度が減少し安定状態に引き込まれる。

・ネットワークは安定状態に停止することはなく、それにより環境の変化に対応する。

2．5　タスク選択

実行タスクの選択は、EDFとSRPとの組み合わせで行う。ただし、SRPはタスク間通信モデルに適用するため、次のように変更する。

- ・タスクのプリエントレベルはタスクのデッドラインとし,小さな値程レベルが高いとする。
- ・実行権を得たタスクは通信相手タスクが通信待ちでかつ相手タスクのデッドラインが手前である場合、実行タスクは相手のプリエントレベルで実行する。
- ・通信相手タスクが実行可能状態のなった時点で、実行タスクのプリエントレベルを本来のレベルに戻す。

これにより通信相手が通信待ちの場合、プリエントされる回数が少なくなり、実行タスクが優先的に処理される。

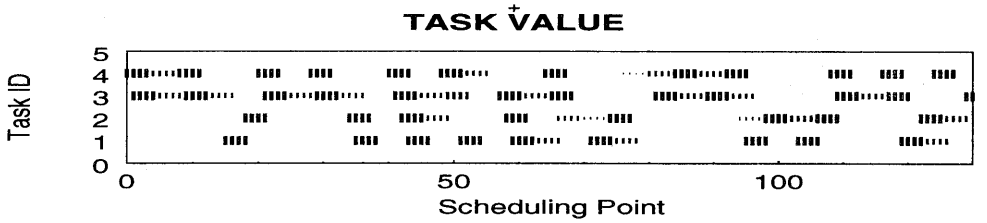
3　シュミレーションとその評価

2つのタスクモデルにより本提案方式のシュミレーションを行った。

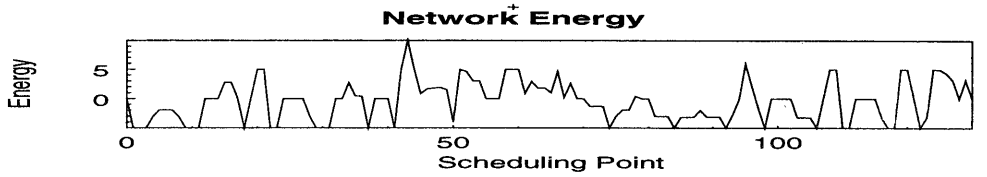
3．1　適応周期の時間的推移

図2に示すタスクネットワークモデルで各タスクの時間特性は最大周期110msec、最小周期90msec、実周期100msec、実CPU使用時間10msecとして適応周期を計算した。結果は図4に示す。

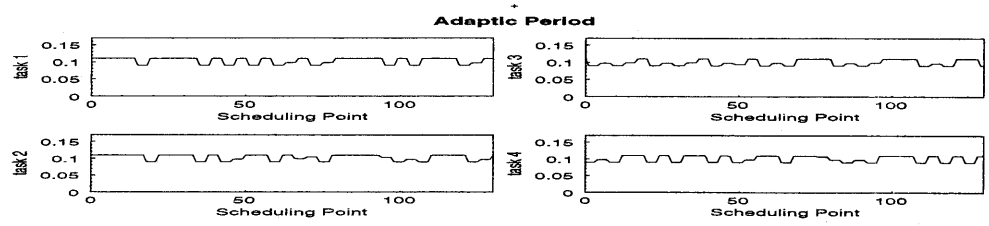
図4(a)は各タスクの重要度を示しているが、より太く長い線は重要度が高いこと表している。タスク1と2は協調関係にあるため、この2つのタスクが同時に重要度が高められる。一方、タスク3と4の重要度は低くなる傾向が分かる。また逆に、タスク3と4の重要度が高められると、タスク1と2の重要度が低下していく。この2つ状態は図4(b)から分かるようにエネルギーが低い安定状態であり、この安定状態と検索状態がそれぞれ現れる。



(a)タスクの重要度



(b)ネットワークエネルギー



(c)タスク適応周期の変動

図4　タスクの活性度の時間的推移

また、図4(c)のタスク適応周期の変動のグラフはタスク1と2およびタスク3と4が似た形をしていることから、協調関係にあるタスクの重要度はほぼ同期して高められることが分かる。

3．2　CPU割当とコンテキストスイッチ

図5に示すクライアントサーバ型のタスクネットワークモデルで各タスクの正の重みを持つ結合関係に通信を発生させて、タスクへのCPU割当てとコンテキストスイッチの状況をEDFと共にシュミレーションし、比較評価を行った。

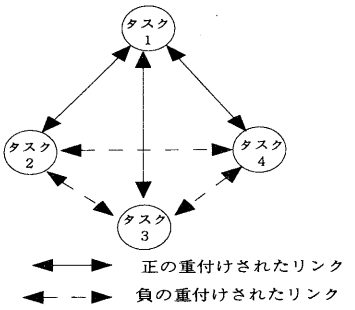


図5　クライアント・サーバ型のタスクネットワーク

図6、7はその結果であるが、各タスクの時間特性は次の通りである。

(a)EDFおよびアダプテーションなし提案方式(EDF+変形SRP)

	実周期	実CPU	送信開始	受信待ち
Task1	40msec	12msec	11msec	1msec
Task2	150msec	30msec	15msec	15msec
Task3	120msec	24msec	12msec	12msec
Task4	100msec	20msec	10msec	10msec

(b)アダプテーションあり提案方式

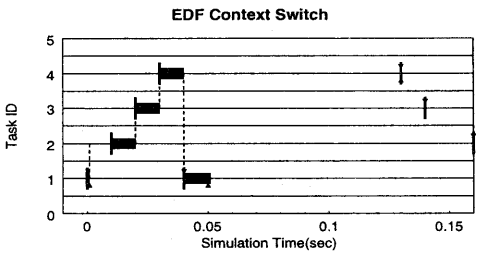
実周期、実CPU使用時間、送信開始時間、受信待ち開始時間は(a)と同様である。

	最大周期	最小周期
Task1	46mes	36msec
Task2	170msec	135msec
Task3	140msec	108msec
Task4	120msec	90msec

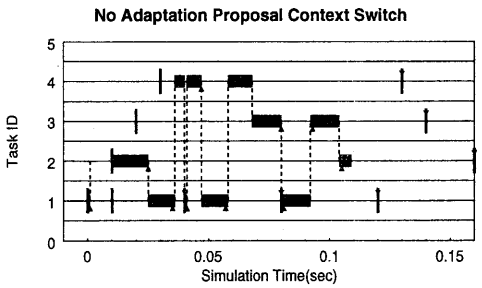
(c)実行開始初期オフセット

Test(1)、Test(2)の各タスク初期起動時間を次ように設定して実施した。

	Test(1)	Test(2)
Task1	0msec	34msec
Task2	10msec	10msec
Task3	20msec	5msec
Task4	30msec	0msec



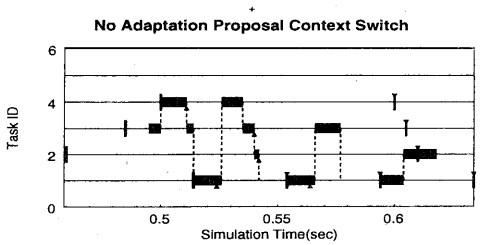
(a)EDF



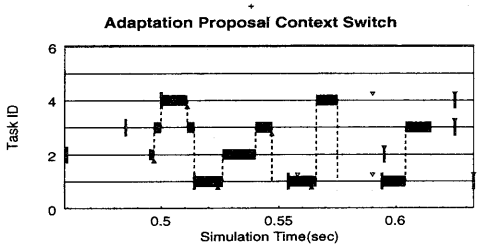
(b)アダプテーションなし提案方式

図6　Test(1)におけるコンテキストスイッチ

Test(1)の場合、EDFは0.04秒でタスク1がデッドラインをオーバしている(図6(a))。これはタスク3、4がタスク2をプリエンプトしてタスク1への通信が最も遅いタイミング（タスク4）で行われていることによる。しかし、提案方式はアダプテーションが有無限らす、デッドラインをオーバーしない(図6(b))。これはタスク1の通信相手であるタスク2がもっと高いプリエントレベルで実行し、他のタスクにプリエントされないためである。しかし、Test(2)の場合、アダプテーションなし提案方式でも0.61秒でタスク2がデッドラインをオーバしている(図7(a))。これはデッドラインが最も遅いタスク2がタスク1、3、4にプリエントされ、タスク2の処理が大きく遅れること



(a)アダプテーションなし提案方式



(b)アダプテーションあり提案方式

図7 Test(2)におけるコンテキストスイッチ

による。

一方、アダプテーションあり提案方式はTest(2)の場合でもデッドラインオーバを発生させない(図7(b))。これはタスクの協調抑制関係（タスク2、3、4は相互に抑制する。負の重みを持つ）による適応周期更新則から、この期間においてタスク2の適応周期が最小周期（重要度が高）に、タスク3、4がの適応周期が最大周期（重要度が低）に変動し、タスク2がタスク3、4により先に処理が行われる。これによりデッドラインオーバを防止している。

4 むすび

本報告では実行タスクの周期的時間特性とタスク間の協調／抑制関係にオンラインで対応する適応型CPUスケジューリングポリシを提案し、そのシュミレーション結果を述べた。

本提案方式は計算機シュミレーションの結果、EDFより明らかに予想しがたい通信待ち時間発生やタスク到着の乱れに有効に機能することを確認した。

今後は、周期幅と適応限界の関連およびより短い時間で環境に適応するため、現在、スケジュールポイントで適応周期更新則を適用しているがこれをコンテキストスイッチ毎に適用、さらに固定とている結合の重みもコンテキストスイッチのタイミングでより短期的な適応のため変動させることを検討する。その上で、実システムに実装し、実環境での評価を行う予定でいる。また、可能であれば他のコンピューティング資源の管理方式として検討を進めていきたい。

謝辞

日頃御指導戴く立命館大学理工学部情報学科、大久保教授並びに（株）ATR環境適応通信研究所、松田室長に深謝します。

参考文献

[1]J. Bae and T.Suda. Survey of Traffic Control Schemes and Protocol in ATM Networks. Proceedings of the IEEE,79(2):170-189,Feburary 1991 (p16)

[2]H.Tokuda, Y.Tobe, S.Chou, and J.Moura ContinuContinuous Media Communication with Dynamic QOS Control Using ARTS with an FDDI Network. Computer Communication Review, 22(4): 88-98, October 1992(P24)

[3]中沢、他「エンドユーザネットワーキングーユーザアダプティブエージェントによるQoS制御」、信学技法、CQ97-6,PP.36-46, 1997

[4]河内谷　清久仁、徳田　英幸「MKngプロジェクトにおけるマルチメディア技術：QoS制御のための資源交渉手法の提案」、情報勝利学会第55回全国大会講演論文集、2Z-4, Sept, 1997

[5]W. Horn, “Some simple scheduling algorithms” ,Naval Research LogisticsQuarterly 21, 1974

[6]T.P.Baker, “Stack-based scheduling for real-time processes”, Journal of Real-Time Systems, 3, 1991

[7]David E.Rumelhart, Jmes L.McClelland, PDP Reserach Group PARALLEL DISTRIBUTED PROCESSING, The MIT Press, 1986