

データ鮮度に着目した新しい応答時間の定義とその応用†

福 岡 和 彦††

リアルタイムシステムの応答時間を評価する尺度として、処理要求から出力までの経過時間という従来の尺度（反応時間）に加えて、出力されるデータの新鮮さという新概念の尺度を導入し、この尺度をデータ鮮度と名付けることにした。本論文では、データ鮮度を、各プロセスの処理時間、プロセス相互の結合関係（同期結合または非同期結合）、プロセスの起動間隔、および外界の状態変化間隔の関数として求め、さらにデータ鮮度と反応時間の関係を明らかにする。また、電話での天気予報案内と、列車の進路制御を例にして、データ鮮度と反応時間の計算例を示す。性能評価の世界にデータ鮮度の概念を導入したことにより、鮮度を犠牲にすれば反応時間を短縮できることとか、制御時刻以前に出された計画変更指示が制御に反映されない可能性があることなど、並行処理の本質にかかわる事象を無理なく説明できるようになった。

1. ま え が き

午後7時のテレビの天気予報を見ていると、リアルタイムシステム¹⁾の興味深い特徴に気付く。

まず、雲の流れを示すために、数枚の航空写真が連続して写し出される。これらはいずれも、雲の写真と呼ばれる気象データである。雲の写真という同じ名前のデータが、古いものから新しいものまで多数存在しており、午前9時とか午後3時など、気象衛星から気象データを受けた時刻によって識別されている。このことから、リアルタイムシステムのデータは、名前と時刻を対にして初めて、ユニークに識別できることが分かる²⁾。

次に、気圧配置を示すために、午後3時の雲の写真と天気図を重ね合わせた図が写し出される。この天気図は、午後3時の気象データから作られたものであろう。実は、雲の写真は午後6時のものまで示されるのに、午後6時の天気図は示されない。つまり、視聴者に示される天気図は、最新の気象データから作られたものではない。何時間か過去の気象データから作られた天気図ではあるが、何とか視聴者の役に立っているのであろう。ただ、それより古くなれば、抗議が殺到しそうである。このことから、リアルタイムシステムの出力データには、その時間よりも古くてはならないような許容時間が存在することが分かる。

電話を使って天気予報を知ろうとすると、ダイヤル177を回せば、何秒かのちに何時間か過去の予報データを得ることができる。このとき、電話を掛けた

人は二種類の応答時間を意識する。一つは、要求を出してから返答を得るまでの時間（反応時間）であり^{3),4)}、もう一つは、得られるデータの新鮮さ（データ鮮度）である。

応答時間は、従来より、リアルタイムシステムの性能を評価するための尺度として用いられてきた。ところが、従来より議論されてきた応答時間は、すべて反応時間についてであり、データ鮮度を応答時間評価尺度として捕えた例は見当たらない。電話による天気予報案内の例からも明らかなように、出力データの鮮度を無視して、反応時間だけでリアルタイムシステムの応答時間を議論することはできない。

本報告では、出力データの鮮度に着目した応答時間の新定義を与え、鮮度の計算式を求め、鮮度の応用例を示す。

2. 二種類の応答時間

2.1 平均応答時間と許容応答時間

リアルタイムシステムは、実世界を外界とするシステムである。図1に示すように、リアルタイムシステムは、外界より外界状態を入力し、そのデータを処理して、処理結果を外界に出力する。

リアルタイムシステムの外界は、主に機械である。機械には判断能力がないため、システムからの応答が遅れると、動作しなかったり誤動作したりする。外界ごとに定まる許容応答時間内に応答できるか否かで、システムの性能が評価されねばならない。このような性質は、ハードリアルタイムと呼ばれている⁵⁾。

それに対し、通常のオンラインシステムの外界は、人間である。人間には判断能力があるため、システムからの応答が多少遅れても許容される。平均応答時間の大小で、システムの性能が評価される。このような

† Data Freshness: A New Concept for Response Time Definition by KAZUHIKO FUKUOKA (Systems Development Laboratory, Hitachi, Ltd.).

†† (株)日立製作所システム開発研究所

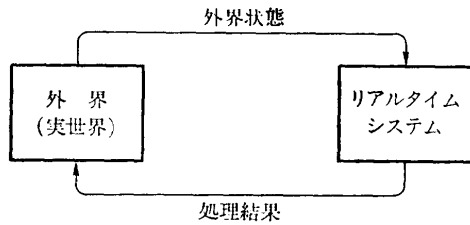


図1 リアルタイムシステムと外界との関係

Fig. 1 Relations between a real time system and its external world.

性質は、ソフトリアルタイムと呼ばれている。

本報告で扱う対象は、ハードリアルタイムな許容応答時間の存在するシステムである⁶⁾。

2.2 反応時間とデータ鮮度

Martin は、応答時間を「計算機システムに対する処理要求が発行されてから、システムが応答を返すまでの経過時間」と定義した⁹⁾。これが、広く一般に認められた応答時間の評価尺度である。ところが、リアルタイムシステムには、この尺度だけでは表現できない応答時間がある。それを、簡単なモニタリングシステムの例を用いて説明する。

図2において、トラッキングプロセス x は、外界 e_1 からのトラッキング要求によって起動され、その時点の外界 e_1 の状態を取り込み、計算機内に保持する外界 e_1 状態データを更新するプロセスであり、モニタリングプロセス y は、外界 e_2 からのモニタリング要求によって起動され、その時点で最新の外界 e_1 状態データを、外界 e_2 に出力するプロセスである。

この状況において、プロセス y の出力動作に対する

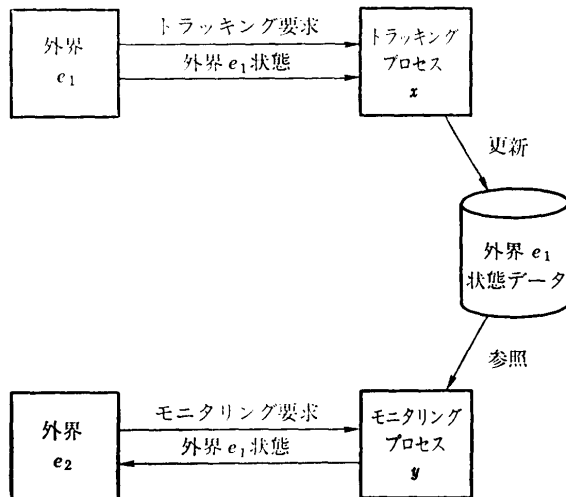


図2 モニタリングシステムの構成例

Fig. 2 An example of a monitoring system structure.

二種類の応答時間が考えられる。一つは、プロセス y の起動から出力までの経過時間である。本報告では、これを「反応時間 (Reaction time)」と呼ぶことにする。もう一つは、プロセス y の出力データ (外界 e_1 状態データ) を作るもととなった外界の状態 (外界 e_1 の状態) が別の状態に変化する直前の時刻から出力までの経過時間である。この時間は、出力データの内容の新鮮さを表す尺度であり、本報告では、これを「鮮度 (Freshness)」と呼ぶことにする。なお、鮮度は正の値で表すものとし、その値が小さいほど新鮮なデータであることを意味するものとする。

3. 鮮度を求めるための前提条件

3.1 外界の特性

リアルタイムシステムの外界は、互いに非同期に動作する複数個の要素から構成される (以下、各要素をそれぞれ別個の外界と考えることにする)。

各外界の状態は、時々刻々変化する。炉内温度のように、その状態が連続して変化する外界もあれば、炉内鋼塊数のように、その状態が不連続に変化する外界もある。これらの状態変化 (または状態) がリアルタイムシステムに取り込まれる。しかし、すべての状態変化が取り込まれるわけではなく、リアルタイムシステムの処理に必要な状態変化だけが取り込まれる。以下、外界の状態変化という言葉は、リアルタイムシステムの処理に必要な変化という意味で用いることにする。

いま、外界 e の状態変化の時間間隔の最小値 (最小変化間隔) を S_e 、最大値 (最大変化間隔) を L_e とし、どの外界においても、 $S_e > 0$ と仮定する。

また、外界 e の状態が時刻 t で変化したとき、変化前のデータとは異なるデータがその時点で生成されたものと考え、そのデータを $D_{e,t}$ と表現し、名称 e と時刻 t で識別することにする。

3.2 リアルタイムシステムの特性

(a) プロセスの起動

リアルタイムシステムは、多数のプロセスから構成される。それらのプロセスは、起動条件の違いによって二種類に分けられる。

一つは、外界の状態変化時に、外界から発せられる割込信号によって起動されるプロセスである。タイマも外界と考えて、タイマからの割込信号によって、周期的あるいは定刻に起動されるプロセスも、この分類に入れる。このようなプロセスを、「先頭プロセス」

と呼ぶことにする。外界 e によって起動される先頭プロセス p の最小起動間隔 S_p , 最大起動間隔 L_p は, それぞれ, 外界 e の最小変化間隔 S_e , 最大変化間隔 L_e に等しい。

もう一つは, 他プロセスによって起動されるプロセスである。このようなプロセスを「途中プロセス」と呼ぶことにする。プロセス q によって起動される途中プロセス p の起動間隔は, プロセス q の起動間隔に等しい。

プロセス q がプロセス p を起動するタイミングは, プロセス q の動作終了時点であると仮定する。また, プロセスの動作終了時点で, そのプロセスのデータ生成が完了するものとする。

(b) データ受渡し形態

各プロセスは, 他プロセス (または外界) によって生成されたデータを参照して, データを生成する。いま, 外界 e によって起動される先頭プロセスをプロセス p とするとき, 外界 e の時刻属性 t のデータ $D_{e,t}$ を参照するプロセス p は, それと同一の時刻属性を持つデータ $D_{p,t}$ を生成するものとする。また, プロセス q によって起動される途中プロセスをプロセス p とするとき, プロセス q の時刻属性 t のデータ $D_{q,t}$ を参照するプロセス p は, それと同一の時刻属性を持つデータ $D_{p,t}$ を生成するものとする。

プロセス間のデータ受渡し形態は, 起動との関係で二種類に分けられる。ここで, データを参照するプロセスを x , プロセス x を起動するプロセスを y , y 以外のあるプロセスを z とする。

一つのデータ受渡し形態は, プロセス x がプロセス y の生成データ $D_{y,t}$ を参照する場合である。プロセス x は, プロセス y がデータ $D_{y,t}$ を生成した時点で起動される。このデータ受渡し形態を「同期結合」と呼ぶことにし, 実線矢印 ($y \rightarrow x$) で両者のデータ受渡し関係を表現する。

もう一つのデータ受渡し形態は, プロセス x がプロセス z の生成データ $D_{z,t}$ を参照する場合である。プロセス x は, プロセス z と無関係なタイミングで動作するのであるから, プロセス z によって生成されたデータのうち, その時点で最新のデータ $D_{z,t}$ を参照することになる。このデータ受渡し形態を「非同期結合」と呼ぶことにし, 破線矢印 ($z \dashrightarrow x$) で両者のデータ受渡し関係を表現する。

(c) プロセスの処理時間

プロセスの処理時間は, プロセスごとに一定である

と仮定する⁷⁾。プロセス x の処理時間は P_x で表す。

(d) プロセッサの割当て

プロセスが動作しようとするときには, 必ず十分なプロセッサが割り当てられるものとする。つまり, リソース共用による処理待ちは発生しないと仮定する。

4. 鮮度の求め方

4.1 鮮 度

プロセス 1 は, 外界 0 のデータ $D_{0,t}$ を参照してデータ $D_{1,t}$ を生成し, プロセス y は, プロセス x の生成データ $D_{x,t}$ を参照してデータ $D_{y,t}$ を生成するものとする (ただし, $y=x+1, x \in \{1, 2, \dots\}$)。

この状況において, 以下, 先頭プロセス 1 によって生成されるデータの鮮度と, 途中プロセス y によって生成されるデータの鮮度とを, 同期結合の場合と非同期結合の場合に分けて, 求めることにする。

4.1.1 先頭プロセスの生成データの鮮度

外界 0 の状態が, 時刻 α, β, \dots でそれぞれ $D_{0\alpha}, D_{0\beta}, \dots$ に変化するものとする。

(a) 同期結合の場合

外界 0 とプロセス 1 が同期結合の場合, プロセス 1 は, 時刻 α, β, \dots で起動され, データ $D_{0\alpha}, D_{0\beta}, \dots$ を参照して, 処理時間 P_1 が経過した時刻 $\alpha+P_1, \beta+P_1, \dots$ に, データ $D_{1\alpha}, D_{1\beta}, \dots$ を生成する。

時刻 $\alpha+P_1$ から時刻 $\beta+P_1$ までの間, プロセス 1 の生成データのうちで最も新しいものはデータ $D_{1\alpha}$ である。データ $D_{1\alpha}$ は, データ $D_{0\alpha}$ から得られたのであるから, データ $D_{1\alpha}$ の鮮度は, 外界 0 の状態が $D_{0\alpha}$ から $D_{0\beta}$ に変化する直前の時刻からの経過時間として求められる。

つまり, 時刻 t ($\alpha+P_1 \leq t < \beta+P_1$) におけるデータ $D_{1\alpha}$ の鮮度は, $t-\beta$ である。その鮮度は, 時刻 $\beta+P_1$ で最大になり, 時刻 $\alpha+P_1$ で最小になる。最大値は, 定数 P_1 であり, 最小値は, 時刻 α と時刻 β の時間差の関数 ($P_1+\alpha-\beta$) である。

時刻 α と時刻 β の時間差は, 外界 0 の状態変化の時間間隔である。その時間間隔の最小値は S_0 であるから, $(\beta-\alpha)$ が S_0 に等しいときに, データ $D_{1\alpha}$ の鮮度の最小値が最大になる。

$$F^\circ(0 \rightarrow 1) = P_1 - S_0 \quad (4.1)$$

$$F^*(0 \rightarrow 1) = P_1 \quad (4.2)$$

ここで, $F(0 \rightarrow 1)$ は, 外界 0 のデータを参照するプロセス 1 が生成するデータの鮮度を表し, 矢印 “ \rightarrow ” は, 両者が同期結合であることを表す。 F° は, いか

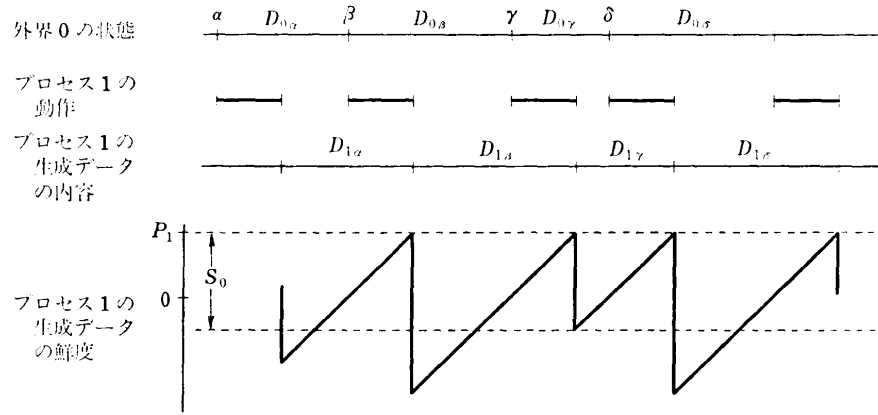


図3 先頭プロセス生成データの鮮度—同期結合の場合

Fig. 3 Freshness of data D_i made by process 1 which is evoked synchronously with the state transitions of external 0.

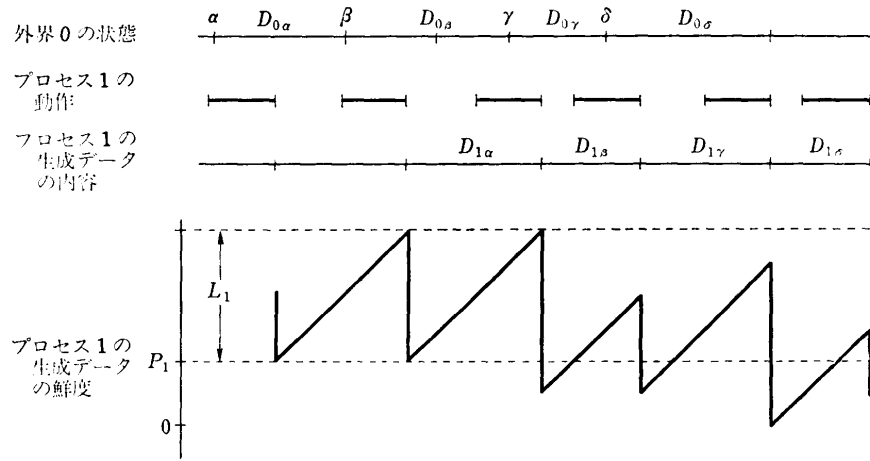


図4 先頭プロセス生成データの鮮度—非同期結合の場合

Fig. 4 Freshness of data D_i made by process 1 which is evoked asynchronously with the state transitions of external 0.

なる条件でも、それより大きくなることはない最小の鮮度を、 F^* は、いかなる条件でも、それより大きくなることはない最大の鮮度を、意味する。

図3に、プロセス1の生成データの鮮度が、時間経過とともに変化する様子を示す。

(b) 非同期結合の場合

外界0とプロセス1が非同期結合の場合、プロセス1は、外界0の状態変化時刻 α, β, \dots とは無関係な時刻 a, b, \dots に起動される。

時刻 a で起動されるプロセス1は、外界0のデータ $D_{0\alpha}$ を参照して、時刻 $a+P_1$ にデータ $D_{1\alpha}$ を生成するものとし、時刻 b で起動されるプロセス1は、 $D_{0\beta}$ 以降の外界0データを参照するものとする。

このとき、時刻 t ($a+P_1 \leq t < b+P_1$) におけるデー

タ $D_{1\alpha}$ の鮮度は、同期結合の場合と同様に、 $t-\beta$ である。その鮮度は、時刻 $b+P_1$ で最大になり、時刻 $a+P_1$ で最小になる。最大値は、 $P_1+b-\beta$ であり、最小値は、 $P_1+a-\beta$ である。

$\alpha \leq a < \beta$ であるから、 a が β に限りなく近づいたときに、データ $D_{1\alpha}$ の鮮度の最小値が最大になる。また、 $(b-\beta) = (b-a) + (a-\beta)$ であるから、 $(b-a)$ がプロセス1の最大起動間隔 L_1 に等しいときに、データ $D_{1\alpha}$ の鮮度の最大値が最大になる。

$$F^{\circ}(0 \rightsquigarrow 1) = P_1 \tag{4.3}$$

$$F^*(0 \rightsquigarrow 1) = P_1 + L_1 \tag{4.4}$$

ここで、矢印“ \rightsquigarrow ”は、外界0とプロセス1が非同期結合であることを表す。

プロセス1の生成データの鮮度が、時間経過とともに

に変化する様子を、図4に示す。

4.1.2 途中プロセスの生成データの鮮度

プロセス x が、時刻 a, b, \dots でそれぞれ、データ $D_{x\alpha}, D_{x\beta}, \dots$ を生成するものとする。

(a) 同期結合の場合

プロセス x とプロセス y が同期結合の場合、プロセス y は、時刻 a, b, \dots で起動され、データ $D_{y\alpha}, D_{y\beta}, \dots$ を参照して、処理時間 P_y が経過した時刻 $a+P_y, b+P_y, \dots$ に、データ $D_{y\alpha}, D_{y\beta}, \dots$ を生成する。データ $D_{y\alpha}, D_{y\beta}, \dots$ の鮮度は、当然、データ $D_{x\alpha}, D_{x\beta}, \dots$ の鮮度に、処理時間 P_y を加えたものになる。

$$F^\circ(0 \dots x \rightarrow y) = F^\circ(0 \dots x) + P_y \quad (4.5)$$

$$F^\bullet(0 \dots x \rightarrow y) = F^\bullet(0 \dots x) + P_y \quad (4.6)$$

ここで、 $0 \dots x$ は、外界0からプロセス x までのプロセス列を表し、 $0 \dots x \rightarrow y$ は、 $0 \dots x$ にプロセス y を加えたプロセス列を表す。 $F^\circ(0 \dots x)$ は、プロセス x の生成データの鮮度を、 $F(0 \dots x \rightarrow y)$ は、プロセス y の生成データの鮮度を表す。

(b) 非同期結合の場合

プロセス x とプロセス y が非同期結合の場合、プロセス y は、プロセス x のデータ生成時刻 a, b, \dots とは無関係な時刻 a', b', \dots に起動される。

時刻 a' で起動されるプロセス y は、データ $D_{x\alpha}$ を参照して、時刻 $a'+P_y$ にデータ $D_{y\alpha}$ を生成するものとし、時刻 b' で起動されるプロセス y は、 $D_{x\beta}$ 以降のプロセス x 生成データを参照するものとする。

時刻 $a'+P_y$ から時刻 $b'+P_y$ までの間、プロセス y の生成データのうちで最も新しいものはデータ $D_{y\alpha}$ である。データ $D_{y\alpha}$ の鮮度は、時刻 a' におけるデータ $D_{x\alpha}$ の鮮度に、時刻 a' からの経過時間を加えたものになる。データ $D_{x\alpha}$ の鮮度は時刻 b の直前で最大になるから、時刻 a' が時刻 b の直前の場合、データ $D_{y\alpha}$ の鮮度は最大になる。

$$F^\circ(0 \dots x \rightarrow y) = F^\circ(0 \dots x) + P_y \quad (4.7)$$

$$F^\bullet(0 \dots x \rightarrow y) = F^\bullet(0 \dots x) + P_y + L_y \quad (4.8)$$

4.1.3 鮮度の計算式

外界0からプロセス n までのプロセス列が存在するとしたときの、プロセス n 生成データの鮮度を求めることにする。

(a) すべて同期結合のとき

外界0からプロセス n まですべて同期結合のとき、式(4.1)、(4.5)から、次式が求まる。

$$F^\circ(0 \dots n) = \sum_{v=1}^n P_v - S_0 \quad (4.9)$$

(b) 少なくとも1個の非同期結合があるとき

プロセス i とプロセス j ($=i+1$) が非同期結合で、プロセス j からプロセス n まですべて同期結合であるとする。

式(4.6)、(4.8)から、次式が求まる。ただし、プロセス x とプロセス y ($=x+1$) が同期結合のとき $\delta_{xy} = 0$ 、非同期結合のとき $\delta_{xy} = 1$ とする。

$$F^\bullet(0 \dots y) = F^\bullet(0 \dots x) + P_y + \delta_{xy} \times L_y \quad (4.10)$$

また、式(4.2)、(4.4)から、次式が求まる。

$$F^\bullet(0 \dots 1) = P_1 + \delta_{01} \times L_1 \quad (4.11)$$

式(4.10)、(4.11)から、次式が求まる。

$$F^\bullet(0 \dots i) = \sum_{v=1}^i (P_v + \delta_{xv} \times L_v) \quad (4.12)$$

プロセス j からプロセス n まですべて同期結合であり、プロセス i とプロセス j は非同期結合であるから、式(4.5)、(4.7)、(4.12)から、次式が求まる。

$$\begin{aligned} F^\circ(0 \dots n) &= F^\circ(0 \dots j) + \sum_{v=j+1}^n P_v \\ &= F^\bullet(0 \dots i) + \sum_{v=j}^n P_v \\ &= \sum_{v=1}^n (P_v + \delta_{xv} \times L_v) - L_j \end{aligned} \quad (4.13)$$

プロセス j からプロセス n まですべて同期結合であるから、 L_j は L_n に等しい。ゆえに、プロセス n の生成データの鮮度は、次のようになる。

$$F^\circ(0 \dots n) = \sum_{v=1}^n (P_v + \delta_{xv} \times L_v) - L_n \quad (4.14)$$

この $F^\circ(0 \dots n)$ は、プロセス n によって生成された各データの、それぞれの生成時刻での鮮度のうちで最大の鮮度を、表している。この F° をデータの(最悪)鮮度 F とする。

4.2 反応時間

プロセス1が外界0のデータを参照し、プロセス y がプロセス x の生成データを参照するものとする(ただし、 $y=x+1$, $x \in \{1, 2, \dots\}$)。

この状況において、以下、先頭プロセス1と途中プロセス y の反応時間を求めることにする。

4.2.1 先頭プロセスの反応時間

(a) 同期結合の場合

外界0とプロセス1が同期結合の場合、外界0の状態が変化した時刻にプロセス1が起動される。そのため、反応時間はプロセス1の処理時間 P_1 になる。

$$R(0 \rightarrow 1) = P_1 \quad (4.15)$$

ここで、 $R(0 \rightarrow 1)$ はプロセス1の反応時間を表す。

(b) 非同期結合の場合

外界0とプロセス1が非同期結合の場合、プロセス1は、外界0の状態変化時刻とは無関係な時刻に起動される。そのため、最悪の場合、前回の起動直後の状態変化を今回の起動時に処理することになる。プロセス1の反応時間は、同期結合の場合に比べて、プロセス1の最大起動間隔 L_1 だけ長くなる。

$$R(0 \rightarrow 1) = P_1 + L_1 \quad (4.16)$$

4.2.2 途中プロセスの反応時間

(a) 同期結合の場合

プロセス x とプロセス y が同期結合の場合、プロセス x がデータを生成した時点でプロセス y が起動される。そのため、プロセス y の反応時間は、プロセス x の反応時間にプロセス y の処理時間 P_y を加えた値になる。

$$R(0 \rightarrow x \rightarrow y) = R(0 \rightarrow x) + P_y \quad (4.17)$$

(b) 非同期結合の場合

プロセス x とプロセス y が非同期結合の場合、プロセス y は、プロセス x のデータ生成時刻とは無関係な時刻に起動される。そのため、最悪の場合、つまりプロセス x がデータを生成する直前にプロセス y が起動された場合、そのプロセス x 生成データは次回のプロセス y 起動時に処理されることになる。プロセス y の反応時間は、同期結合の場合に比べて、プロセス y の最大起動間隔 L_y だけ長くなる。

$$R(0 \rightarrow x \rightarrow y) = R(0 \rightarrow x) + P_y + L_y \quad (4.18)$$

4.2.3 反応時間の計算式

式(4.15)、(4.16)、(4.17)、(4.18)から、プロセス n の(最悪)反応時間が求まる。

$$R(0 \rightarrow n) = \sum_{y=1}^n (P_y + \delta_{xy} \times L_y) \quad (4.19)$$

4.3 鮮度と反応時間の関係

(a) すべて同期結合のとき

式(4.9)と式(4.19)から、次の関係が求まる。

$$R(0 \rightarrow n) = F(0 \rightarrow n) + S_0 \quad (4.20)$$

(b) 少なくとも1個の非同期結合があるとき

式(4.14)と式(4.19)から、次の関係が求まる。

$$R(0 \rightarrow n) = F(0 \rightarrow n) + L_n \quad (4.21)$$

5. 鮮度と反応時間の計算例

5.1 電話での天気予報案内

電話で177を回すと天気予報が聞ける。このサービスがなされるためには、多くの観測点(外界 e_1)の気象データを集めて天気予報を計算する仕事(プロセス x)と、電話(外界 e_2)からの要求を受けて予報を返答する仕事(プロセス y)とが必要である。

処理の方法として二種類考えられる。一つは、電話からの要求があった時点でプロセス x を起動し、プロセス x はその時点で最新の気象データを収集して予報データを生成し、その予報データをプロセス y が返答する方法である(方法I)。もう一つは、周期的にプロセス x を起動し、プロセス x は要求の有無にかかわらず予報データを次々と生成し、電話からの要求があった時点で最新の予報データをプロセス y が返答する方法である(方法II)。

外界 e_1 の内容は時々刻々変化しているため、外界 e_1 とプロセス x は非同期結合である。方法Iの場合、外界 e_2 とプロセス x は同期結合、プロセス x とプロセス y は同期結合である。方法IIの場合、外界 e_2 とプロセス y は同期結合、プロセス x とプロセス y は非同期結合である。各方法における外界とプロセスの結合形態を、図5に示す。

各方法の鮮度と反応時間は、次式のようになる。

(方法I)

$$F_1(e_1 \rightarrow x \rightarrow y) = P_x + P_y$$

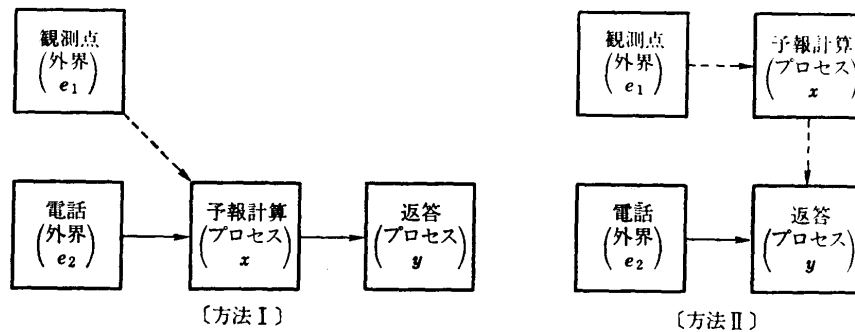


図5 電話での天気予報案内システムのソフトウェア構成

Fig. 5 A software structure of weather forecast information system by telephone.

$$R_I(e_2 \rightarrow x \rightarrow y) = P_x + P_y$$

(方法Ⅱ)

$$F_{II}(e_1 \rightarrow x \rightarrow y) = P_x + P_y + L_x$$

$$R_{II}(e_2 \rightarrow y) = P_y$$

ここで、プロセス x の処理時間 P_x と最大起動間隔 L_x を共に1時間、 P_y を20秒とすると、方法Ⅰでは、鮮度と反応時間がいずれも約1時間となり、最小鮮度の予報データが得られるが、電話をかけてから1時間も待たされてしまう。方法Ⅱでは、電話をかけてからわずか20秒で返答されるが、そのとき得られる予報データは最悪2時間も過去の気象データをもとにして作られたものになってしまう。

電話での天気予報案内の場合、長い時間待つわけにはいかないから方法Ⅰは成り立たず、実際には方法Ⅱが採用されている。鮮度を犠牲にして、反応時間の短縮を計っているのである。

5.2 列車の進路制御

各列車の各駅への進入進路は、運行計画として前もって定められている。ただし、この計画は、列車の運行が乱れてしまったときに、指令員によって変更されることがある。

列車が駅の手前のある地点に到達すると、計画に定められた進入進路に対応する信号機が、停止現示から進行現示に変わるよう制御される。地点到達後、許容応答時間内に信号機が制御されれば問題はないが、許容応答時間より遅れて制御されると、列車が減速せねばならなくなる。

この状況において、図6に示すように、指令員(外界 e_1)からの計画変更要求を受けて計画を変更する仕事(プロセス x)と、検出装置(外界 e_2)からの処理要求を受けて列車の地点到達を検出する仕事(プロセス y)と、地点到達検出時に、その時点で最新の計画に従って信号機を制御する仕事(プロセス z)とが存

在するものとする。

信号機制御に使った計画データの鮮度と、列車の地点到達から信号機制御までの反応時間を求める。

$$F(e_1 \rightarrow x \rightarrow z) = P_x + P_z$$

$$R(e_2 \rightarrow y \rightarrow z) = P_y + P_z$$

ここで、プロセス x, y, z の処理時間をそれぞれ7秒、1秒、2秒とすれば、鮮度は9秒、反応時間は3秒になる。

列車の地点到達時刻を t とすると、時刻 $(t + P_y - P_x)$ (上例では $t - 6$ 秒) までに出力された計画変更要求は必ず信号機制御に反映されるが、その時刻以降に出力された計画変更要求は信号機制御に反映されない。

指令員の立場から、この意味を考察してみると、列車の地点到達以前に進入進路を変更しようとしたにもかかわらず目的を達成できない、ということが起こりうるわけである。

6. む す び

リアルタイムシステムの応答時間を評価する尺度として、処理要求から返答までの経過時間という従来の尺度(反応時間)に加えて、出力されるデータの新鮮さという新概念の尺度(データ鮮度)を導入した。

性能評価の世界にデータ鮮度の概念を導入したことにより、鮮度を犠牲にすれば反応時間を短縮できることとか、制御時刻以前に出力された計画変更指示が制御に反映されない可能性があることなど、並行処理の本質にかかわる事象を、無理なく説明できるようになった。

データ鮮度は、リアルタイムシステムを設計する上での重要な評価関数の一つであり、データ鮮度の概念が、リアルタイムシステムの設計、特にモニタリング機能を含むプロセス制御システムの設計に、役立つものと考えている。

参 考 文 献

- 1) 松本吉弘: リアルタイムシステム, 昭晃堂, 東京 (1984).
- 2) 福岡和彦: データ体内時計を用いた並行プロセス同期方式, 情報処理学会論文誌, Vol. 25, No. 3, pp. 372-378 (1984).
- 3) Martin, J. T.: *Programming Real-Time Computer Systems*, Prentice-Hall, Englewood Cliffs, N. J. (1965).
- 4) Yourdon, E.: *Design of On-line Computer Systems*, Prentice-Hall, Englewood Cliffs, N. J. (1972).

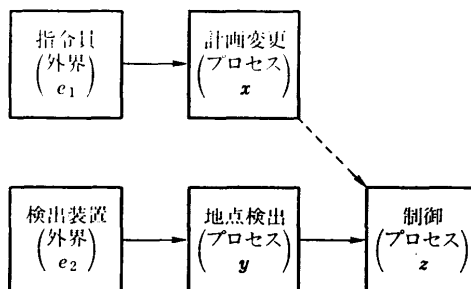


図6 列車の進路制御システムのソフトウェア構成
Fig. 6 A software structure of train route control system.

- 5) Manacher, G. K.: Production and Stabilization of Real-Time Task Schedules, *J. Ass. Comput. Mach.*, Vol. 14, No. 3, pp. 439-465 (1967).
- 6) Leinbaugh, D. W.: Guaranteed Response Times in a Hard-Real-Time Environment, *IEEE Trans. Softw. Eng.*, Vol. SE-6, No. 1, pp. 85-91 (1980).
- 7) Liu, C. L. and Layland, J. W.: Scheduling Algorithms for Multiprogramming in a Hard-Real-Time Environment, *J. Ass. Comput. Mach.*, Vol. 20, No. 1, pp. 46-61 (1973).

(昭和 61 年 10 月 6 日受付)

(昭和 62 年 4 月 15 日採録)



福岡 和彦 (正会員)

昭和 21 年生。昭和 43 年東京大学工学部精密機械工学科卒業。同年(株)日立製作所中央研究所入社。現在、同社システム開発研究所にて、ワークステーション基本ソフトウェアの研究に従事。並行処理システムに興味をもつ。著書「制御用計算機におけるリアルタイム技術」(共著, コロナ社)。IEEE 会員。