数理解析研究所講究録 1375 巻 2004 年 151-157

線形制約式を用いた時間QoS一貫性の検証法
Consistency Checking of Timeliness QoS using Linear Constraints

岡野 浩三　森 一夫　谷口 健一

Kozo OKANO, Kazuo MORI, and Kenichi TANIGUCHI

大阪大学大学院情報科学研究科
Graduate School of Information Science and Technology, Osaka University
{okano,kazuo-mr,taniguchi}@ist.osaka-u.ac.jp

あらまし 分散環境におけるコンポーネントに対する提供 Timeliness QoS とシステム全体における要求 Timeliness QoS を線形制約式で表現し，要求 Timeliness QoS が提供 Timeliness QoS の下で満たされること（一貫性）を検査する手法を提案する．提案手法では，この QoS 検査問題を線形計画法の非可解性問題に帰着させる．

キーワード 時間 QoS，線形不等式，検証，コンポーネント システム

1. はじめに

本稿では線形制約式で表現された Timeliness QoS(時間 QoS) の一貫性を検証する方法を提案する．コンポーネントベースの設計法の一つに，各コンポーネントに対して QoS の仕様記述を与える方法があり [1], UML を用いて設計する際に有用視されている [6].

QoS のうち，時間に関するものを Timeliness QoS という．コンポーネントや，システム全体に関する Timeliness QoS の正しさの検証については，テストオートマトン [3] に基づき，文献 [7], [8] 等において，分散システムを構成するコンポーネントに対して記述された Timeliness QoS の検証を行なう形式的なアプローチを示している．ここでは各コンポーネントは時間オートマトンネット [2] でモデル化される．これらのアプローチに共通する問題点は大規模なシステムに対しては，メモリ使用量の面から検証が困難になる点である．

以上の点を踏まえ，文献 [9] では，下記の方法を提案している．システムが複数のコンポーネントからなると仮定し，それらの各コンポーネントに提供 Timeliness QoS を課す．この提供 Timeliness QoS がシステム全体が満たすべき要求 Timeliness QoS を満たしているかどうかという問いに対して，これらの QoS を時間変数を用いた線形制約式として記述し，線形制約の可解性判定問題に帰着し判定するアプローチをとる．このアプローチの利点は，コンポーネントを提供 Timeliness QoS で抽象化することにより，各コンポーネントを時間オートマトンなどで陽に記述する方法に比べてスケーラビリティが増すことにある．一般に，Timeliness QoS を信号 x の i 番目の発生時刻 x_i を用いて表すと，i については全称子をもった論理式となる．全称子を残したまま線形計画法を適用することはできない．一方，全称子をなくして元の意味通りの式を生成しようとすると無限個の式が必要となる．しかし実際には必要十分な有限個の式があれば，判定することは可能である．したがって提案手法では，Timeliness QoS の時間特性などを利用し，全称子を持たない有限個の線形制約式に変換し，判定する．

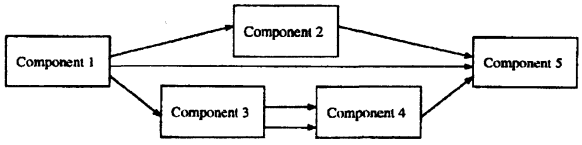

図 1 コンポーネントの接続関係
Fig. 1 Relations of components

本稿では，QoS 一貫性検証方法について詳しく述べる．以後，2. では Timeliness QoS について述べる．3. ではシステムに要求されている QoS の一貫性検証について説明し，4. では導出例について述べる．5. でまとめる．

2. Timeliness QoS

QoS のうち，時間に関するものを Timeliness QoS と

いう．本研究ではこのうちスループット，ジッタ，遅延の3つを扱う．以降ではTimeliness QoSを単にQoSと呼ぶ．

図1はコンポーネントと信号の接続関係を表している．一般に，一つのコンポーネントにはコンポーネントによって定まる n 個の入力および m 個の出力信号 $\mathrm{x}^k(1 \leqq k \leqq n+m)$ が接続しているとする．なお，信号はこのように上付き添字で区別するか，あるいは，誤解がなければ，x，y，zなどの記号を用いる．各信号 x^k に対して，$x_i^k(i \in \mathbb{N})$ という変数を導入する．これは信号 x^k の i 番目の発生時刻（例えばフレーム出力信号における i 番目のフレームの出力時刻）を表す．曖昧さを招かない限り，添字 k を略し，x_i と表す．なお，x^k で信号 x^k の発生時刻系列を意味する．

本研究では検証時に，QoSについては時間変数を介した表現（QoS式）を用いて表現する．各QoS式は全称子が添字 i を束縛している．以下で各QoSに対するQoS式を示す．

2.1 スループット

ある期間 T に信号xが少なくとも K 回発生しなければならないという制約は次のようにQoS式で表現される．

$$\forall i \in \mathbb{N}: \quad x_{i+K-1} - x_i \leqq T$$

同様に，ある期間 T に信号xが高々 K 回発生しなければならないという制約は次のように表現される．

$$\forall i \in \mathbb{N}: \quad x_{i+K-1} - x_i \geqq T$$

一般には，スループットは

$$\forall i \in \mathbb{N}: \quad T' \leqq x_{i+K-1} - x_i \leqq T$$

の形で与えられる．

なお，この形で表現されるスループットはNon-Anchoredスループットと呼ばれる[4]．

なお，Anchoredスループットは以下の様に定義できるが，本稿ではこのAnchoredスループットは扱わない．

$$\forall j \in \mathbb{N}: \quad \#\{i \mid jT \leqq x_i < (j+1)T\} \geqq K$$

$$\forall j \in \mathbb{N}: \quad \#\{i \mid jT \leqq x_i < (j+1)T\} \leqq K$$

ここで $\#A$ は集合 A の要素数を表す．

2.2 ジッタ

T 間隔で発生する信号xのジッタ制約のQoS式は次のように表現される．

$$\forall i \in \mathbb{N}: \quad T - m \leqq x_{i+1} - x_i \leqq T + M \quad (m, M: \text{定数})$$

なお，この形で表現されるジッタはNon-Anchoredジッタと呼ばれる．これと対になる概念であるAnchoredジッタは

$$\forall i \in \mathbb{N}: \quad iT - m \leqq x_i \leqq iT + M \quad (m, M: \text{定数})$$

で表現されるが，本稿ではこのタイプは扱わない．

2.3 遅　　延

2つの信号xとyの遅延関係が高々 T であるという制約を表すQoS式は次のように表現される．

$$\forall i \in \mathbb{N}: \quad 0 < x_i - y_{Ki+K'} \leqq T$$

なお，

$$\forall i \in \mathbb{N}: \quad |x_i - y_{Ki+K'}| \leqq T$$

により，発生時刻に前後関係のない2つの信号に関する遅延束縛条件を表すことができる．2つの信号の同期を表すことができる表現であるが本稿では，やはりこのタイプは扱わない．

2.4 信号に対する仮定

信号系列 x に対して，以下の3つの性質を仮定する[2]．

（1） 単調増加性 $\forall i \in \mathbb{N}: x_i < x_{i+1}$

（2） Non-Zenon性 $\forall K > 0: \exists i: x_i > K$

（3） 非負値性 $\forall i \in \mathbb{N}: x_i \geqq 0$

3. システム要求QoSの一貫性検証

3.1 検証方法のあらまし

システム全体として満たすべきQoSの集合を要求QoS，システムを構成する各コンポーネントが提供するQoSの集合を提供QoSと呼ぶ．以下で，要求QoSの提供QoSに対する一貫性を検証する方法のあらましについて述べる．(1) 要求QoSのQoS式集合と，(2) 各コンポーネントにおける提供QoSのQoS式集合，および，(3) コンポーネントの接続関係（遅延制約を含む）から，有限個の線形制約式からなる一貫性検証式を生成する．QoS式には，全称子があるため，QoS式から変数置き換えなどの単純な方法で線形制約式を生成することはできない．本検証手法では，一つのQoS式から，複数の相当する線形制約式を導出し，検証を行う．本検証手法では最終的に次のような検証式を生成する．

$$\left(\bigwedge \text{提供 } QoS \text{ 式から生成された式}\right) \wedge \neg\left(\bigwedge \text{要求 } QoS \text{ 式から生成された式}\right)$$

この式は直観的に，要求QoS式を満たさないような場合があるかどうかを意味する．この式を充足する解があれば一貫性は保証されないし，この式を充足する解がなければ一貫性は保証される．

3.2 準　　備

問題を定義するためにいくつか定義をここで行なう．

システムを S を (多重辺を許す) 有向グラフ (V, E) と定義する．ここで V はコンポーネント集合（システムの入力/出力地点を表す特殊な要素 In, Out を含む），$E = V \times V$ はインターフェイス（信号の接続関係）集合である．各 $e \in E$ の始点 $o(e)$, 終点 $i(e)$ に対して信号が割り当てられる．簡単のため，信号名に重複はないものとする．信号の集合を X, 始点, 終点の集合をそれぞれ $Io = \{o(e) \mid e \in E\}$, $Ii = \{i(e) \mid e \in E\}$ とする．$Ii(Io)$ から X への写像 $Ii \to X$ $(Io \to X)$ を $Sig_i(Sig_o)$ とする．QoS 式の集合を C とし，各制約は

$\forall i \in \mathbb{N}$: $v_1 \leqq \alpha - \beta \leqq v_2$

と表現する．ここで v_1, v_2 は定数で，α, β は $(e, ix)(e \in X$ は信号名，ix は添字式) で表される信号発生時刻変数とする．一般性を失うことなく，束縛添字変数を i とする．したがって，α, β は，$(e, Ki + K')$ の形でと表現することが可能である．

特にスループット/ジッタ制約については，β 側の添字 i を信号の基準添字，β と α の添字の差を範囲 K' とする．遅延式の場合は両方の添字を対応する信号の基準添字と呼ぶ．

要求 QoS 式の集合は $Re \subset C$ で表し，提供 QoS 式の集合は $Pr \subset C$ で表す．要求 QoS 式の集合 Re は，In, Out に接続する各インターフェース e について In については $Sig_o(e)$, Out については $Sig_i(e)$ である信号についての QoS 式からなる集合である．一方，提供 QoS 式の集合 Pr 中の QoS 式はそれらの信号に関するものではない．

また写像 R を $V \cup E \to C$ と定義する．これはあるコンポーネント/インターフェイスに対応する制約集合を表す写像である．

3.3 問題定義

問題は以下のように与えられる．

[入力] システム S, 要求 QoS 集合 Re, 提供 QoS 集合 Pr, 制約写像 R

[問題] 制約写像 R によって定義づけられる対応関係のもとで，システム S に対する要求 QoS Re を提供 QoS Pr が満たすかどうか

この問題を QoS 式を用いて定義すると

$$(\bigwedge_{p \in Pr} p) \wedge \neg (\bigwedge_{r \in Re} r) \qquad \text{(P1)}$$

の真偽を判定する問題と読み替えることができる．

この式 (P1) が真であれば「要求 QoS Re を 提供 QoS Pr は満たさない」し，一方，式 (P1) が偽あれば「要求 QoS Re を 提供 QoS Pr は満たす」．

3.4 検証可能な QoS とシステムのクラス

本手法で検証可能な QoS とシステムのクラスを示す．

QoS 式は，式中には変数が 2 つだけ存在し，その 2 変数の添字は $(i, i+K)$ または (i, Ki) の関係に限定されている（K はその式にのみ依存する定数）．また (i, Ki) の関係が用いられるのは遅延に関する QoS 式のみである．

本研究で扱うシステムは図 1 のように閉路のない有向グラフとしてみなせるものに制限する．

一貫性判定の制限および，システムの制限として，以下の 4 つを課す．

[クラス制限]

（1） 要求 QoS がジッタ制約であるとき，どこかのコンポーネントでジッタ制約が書かれていなければならない．

（2） 要求 QoS にスループット制約が含まれるとき，どこかのコンポーネントでスループット制約かジッタ制約が書かれていなければならない．また，そのスループット制約における添字の差は，要求制約のそれより小さくなくてはならない．

（3） 要求 QoS に遅延制約が含まれるとき，対象システムの入出力間を結ぶ経路のなかで，遅延関係がすべて与えられているような経路が少なくとも 1 つはなくてはいけない．

（4） 対象システムのコンポーネント接続関係について閉路はない．

制限 (1) をおく理由は，ジッタ制約からスループット制約を推論することはできるが，逆はできないためである．例えば，$\forall i \in \mathbb{N}$: $m \leqq x_{i+1} - x_i \leqq M$ から $\forall i \in \mathbb{N}$: $x_{i+10} - x_i \leqq 10M$ は容易に推論できるが（発信間隔が M 以内ならば，10 回目の発信時刻は最初の発信時刻からは $10M$ 以内に起こる），逆は推論できない (10 回目の発信時刻が最初の発信時刻からは $10M$ 以内に起こるからといって，発信間隔が M 以内であるとは言えない）．制限 (2) は主に提案する手法のアルゴリズムにおけるクラス制約である．(2) は，最終的に検証に用いる式のサイズを有限に押さえることに影響している．制限 (3) も，本質的には制限 (1) と同様の理由により設けている．なお，本稿では同期の Timeliness QoS を扱わないとしているが，絶対値の定義に基づき，場合分けを行なうことにより，理論的には対応可能である．

3.5 検証アルゴリズム

与えられた問題のインスタンスに対して検証式に必要な線形制約式集合を生成する関数 generateFormulae() を次に示す．関数中に現われる $c.\alpha.e$ は QoS 式 c で使われる信号発生時刻変数 α の信号名 e を表す．

[定義 1] QoS 式の線形表現式

QoS 式の全称子を無視し，添字付き変数 x_i を通常の変数 x に置き換えて得られる式（ただし，異なる添字を持つ変数は別変数に置き換える）をもとの QoS 式の線形表現式と呼ぶ． □

[入力] S, Re, Pr

[出力] Re, Pr に対応する線形表現式集合 lRe, lPr

```
function generateFormulae(S)
(T, lRe, lPr) ← (∅, ∅, ∅)
for each c ∈ Re
  T ← (c に現われる信号名，基準添字，範囲)
  lRe ← lRe ∪ (c について全称子を外した線形制約式)
for each {c | c ∈ R(e) ∧ e = (v, Out) ∈ E}
  if c の添字関係が (i, i + K) then
    // (α, p, q) が T に存在
    T ← T ∪ (β, p + K, q)
  else // c の添字関係が (i, Ki)
    // (α, p, q) が T に存在
    T ← T ∪ (β, Kp, Kq)
for each システムの最後のコンポーネント v
  lPr ← lPr ∪ generateCompFormulae(v, T, ∅)
return (lRe, lPr)
```

この関数では，まず全ての要求 QoS 式に現われる信号名，基準添字，範囲の 3 項組を表 T に登録する．システム出力につながるインターフェイスに関する遅延制約があった場合，遅延式に使われる添字によってずれが生まれる．例えば，あるコンポーネントの出力 x と入力 y の間に遅延制約

$0 \leq x_i - y_{i+2} \leq 10$

があるとき，このコンポーネントの出力以降で現われる添字 i に対し，入力以前の添字 i には何の関係もなく，$i+2$ が関係する．式生成時にはこのずれを考慮しなければならない．そこで，この関数ではシステム出力/入力で使われる添字と各コンポーネントの QoS 式で使われる信号変数の添字とのずれを表に保持している．最後にシステムの最後のコンポーネント，すなわちコンポーネントの出力が他のコンポーネントではなく，直接システムの出力に接続されているものについて generateCompFormulae() を呼び出している．generateCompFormulae() を次に示す．

[入力] コンポーネント v, 添字範囲情報表 T, 作業中の線形表現式集合 r

[出力] コンポーネント v に対応する 線形表現式集合 r

```
function generateCompFormulae(v, T, r)
if v がシステムの最後のコンポーネント or 全ての
コンポーネント v'(v' | (v, v') ∈ E) が式を生成済 then
  for each 遅延制約と出力側のスループット/ジッタ制約 c
    r ← r ∪ c に相当する範囲 n までの制約式 ... (*)
  if 遅延の制約 c が存在 then
    if c の添字関係が (i, i + K) then
      T ← T ∪ (c.β.e, 基準添字, n)
    else // c の添字関係が (i, Ki)
      // (c.α.e, p, q) が T に存在
      T ← T ∪ (c.β.e, Kp, Kq)
  for each 入力側のスループット/ジッタ制約 c ∈ R(v)
    r ← r ∪ c に相当する範囲 n までの制約式 ... (*)
  for each コンポーネント v'(v' | (v', v) ∈ E)
    generateCompFormulae(v', T, r)
return r
```

関数 generateCompFormulae() は引数に与えられたコンポーネント v や表 T からシステムの最初のコンポーネントまでの全てのコンポーネントについて制約式を生成する再帰関数である．まず処理するコンポーネントの出力インターフェイス先のコンポーネントが全て処理済かどうかを検査し，処理済ならば，自コンポーネントの式生成を行う．このコンポーネントが複数の出力インターフェイスを持つ場合，各インターフェイス先のコンポーネントから複数回呼び出されるために，最後のインターフェイス先から呼び出されたときのみ生成処理を行うことを意味する．関数内部では，まずコンポーネントの出力信号のスループット/ジッタ制約式と遅延制約式について生成し，次に遅延式があれば generateFormulae() と同じ方法で表に追加する．それからコンポーネントの入力信号の制約式について生成する．自コンポーネントについて全ての制約式を生成した後で，入力インターフェイス先の全てのコンポーネントについて generateCompFormulae() を再帰呼出しする．

式生成 (*) について簡単に述べる．(*) が処理される時点で生成する制約式で使われる信号名，基準添字，範囲の組が表に既に登録されている．これはスループット/ジッタの場合は 1 組であるし，遅延の場合は 2 組である．以下ではスループットの場合についてのみ説明するが，他の場合でも考え方は同じである．制約式が

$v_1 \leqq x_{i+k} - x_i \leqq v_2$ であり，表に (x,j,n) が登録されているとする．このとき，

$v_1 \leqq x_{j+k} - x_j \leqq v_2$

$v_1 \leqq x_{j+2k} - x_{j+k} \leqq v_2$

$v_1 \leqq x_{j+3k} - x_{j+2k} \leqq v_2$

$\cdots$

$v_1 \leqq x_{j+mk} - x_{j+(m-1)k} \leqq v_2 \ (mk = n)$

を生成する．

これらの式から時間性質を考慮にいれることにより

$mv_1 \leqq x_{j+mk} - x_j \leqq mv_2$ を得る．

なお，$mk = n$ でない場合は

$v_1 \leqq x_{i+k} - x_i \leqq v_2 \Rightarrow 0 \leqq x_{i+k'} - x_i \leqq v_2 (0 < k' < k)$

という一般性質を用いて，最後の式の代りに

$v_1 \leqq x_{j+mk-l} - x_{j+(m-1)k} \leqq v_2 \ (mk - l = n)$

を生成し，上と同様に

$mv_1 \leqq x_{j+mk} - x_j \leqq mv_2$

を得る．

[定義 2] 問題のインスタンス I に対し generateFormulae() で生成される線形制約集合 lRe, lPr をもとに作成される

$$(\bigwedge lPr) \wedge \neg(\bigwedge lRe)$$

の式を I の検証式 $V(I)$ と呼ぶ． □

I の検証式は線形制約式の非可解性判定問題の複数の問題インスタンスに帰着することができる．この各部分問題は線形計画法のパッケージを用いることにより実用的に高速に解くことができる．

3.6 検証方法の正しさ

再び式 (P1) を考える．

一般に，要求 QoS 式に現われる全称子で束縛されている変数 i については変数定数 k で代表し，議論することによって，全称子をはずすことができる．一方，提供 QoS 式に現われる全称子で束縛されている変数 i については，変数定数 k を用いた式 $k+1, k+2, k+3$ を順次代入して得られる必要十分な数の線形制約式で提供 QoS 式を代替表現することによって，この式と意味的に等価な式を得ることができる．したがって，検証方法の正しさは提供 QoS 式から上記アルゴリズムによって得られる複数の線形制約式が検証に必要なだけ得られることとその数が有限で押さえられることを示すことによって議論できる．

まず，QoS 式に対していくつかの性質を考察する．

$\forall i \in \mathbb{N}: \ m \leqq x_{i+K} - x_i \leqq M$

が成り立っている場合に

$\forall i \in \mathbb{N}: \ m \leqq x_{i+K+1} - x_{i+1} \leqq M$

も成り立つ（$\forall$ の意味定義より）．一般に任意の 0 以上の定数 c について

$\forall i \in \mathbb{N}: \ m \leqq x_{i+K+c} - x_{i+c} \leqq M$

が成り立つ．このことと，時間に関する加算性を考慮にいれると任意の 0 以上の定数 n について以下が成立する．

$\forall i \in \mathbb{N}: \ nm \leqq x_{i+nK} - x_i \leqq nM$ (1)

次に

$\forall i \in \mathbb{N}: \ m \leqq y_i - x_{Ki+K'} \leqq M$

の式に付いて考える．このときやはり

$\forall i \in \mathbb{N}: \ m \leqq y_{i+1} - x_{K(i+1)+K'} \leqq M$

さらには任意の 0 以上の定数 c について

$\forall i \in \mathbb{N}: \ m \leqq y_{i+c} - x_{K(i+c)+K'} \leqq M$ (2)

が成り立つ．ただし，この場合は，時間に関する加算性を考慮にいれても，以下の式は導出できない．「任意の 0 以上の定数 n について $\forall i \in \mathbb{N}: \ nm \leqq y_i - x_{Ki+Kn+K'} \leqq nM$」

[定義 3] ある QoS 式に対して，添字 i に $i + C$(C は整数定数) を代入して得られた QoS 式をもとの QoS 式のバリアントと呼ぶ．ある QoS 式が与えられたとき，そこから（整数定数値を変えて）複数のバリアントを作成し，各添字付きリテラルに 1 対 1 対応をするように変数割り当てをしながら，それぞれの式に対する線形表現式を得る．これらの線形表現式の集合を，もとの QoS 式に対して，添字の対応を保存した線形表現式集合と呼ぶ． □

[命題 1] クラス制限を満たす問題のインスタンス I に対して構成される式 (P1) と I の検証式 V を考える．(P1) の各 QoS 式に対して，添字の対応を保存した V 中の線形表現式集合が存在する．

[略証 1] 関数 generateCompFormula() での線形表現式の作成方法より，題意は満たされる． □

この命題は，直観的には，与えられた QoS 式に対して意味的に矛盾する線形表現式は生成されないことを保証している．

次の命題は，これらの線形表現式が十分な数だけ生成されることを保証する．

[命題 2] クラス制限を満たす問題のインスタンス I に対して構成される式 (P1) と I の検証式 $VI()$ は等価であるのに十分な長さ (情報量の) の $V(I)$ がアルゴリズムによって生成されている．

[略証 2] 要求 QoS 式に対する線形表現式が generateFormulae() の最初の for each 文で生成される．一般性を失うことなく要求 QoS 式が 1 つとして以下，議論を行なう．要求 QoS 式に用いられる 2 変数に対応する線形表現式上の変数を，x, x' と表す．提供 QoS 式か

ら変数 x, x' に関する，式 (1) と等価な線形表現式が generateFormulae() の最後の for each 文で生成される．

以下，コンポーネントの構成の深さ数 m に関する帰納法を用いる．

$m = 1$ の場合，すなわちコンポーネント数が 1 のとき，変数 x, x' の関係が 1 つのコンポーネントについて閉じている．すなわち，ある変数 w, y があって，w, y に対するスループット/ジッタ制約があり，変数，w, y と変数 x, x' が遅延制約で推移的に関連づけされているか，変数 x, x' 間に遅延制約で推移的に関連づけされているかのいずれかである．前者の場合，クラス制限の (1)，(2) により，変数 w, y に対するスループット/ジッタ制約の QoS 式が存在し，これに対応する添字の範囲と基準点の選び方は generateCompFormulae() で保証されている．後者の場合も変数 x, x' に関する制約を提供 QoS の線形表現式から得られることが確認できる．

$m = K$ のコンポーネントについて generateCompFormulae() が変数 x, x' の接続関係の推移閉包の関係のうちすでに訪問したコンポーネントについては線形表現式をすでに得ていると仮定する．generateFormulae() では接続関係をたどりながら，添字の開始インデックス表現，範囲を構成している．クラス制限の (3) と帰納法の仮定より，追加で生成される式は generateCompFormulae() の再帰処理 1 つ分に相当する．すでにこの段階で対象になるコンポーネントが上記で得ている線形表現式の変数と遅延制約で関係がない場合は，すでに変数 x, x' に関する検証に十分な制約を得ている．すでに得ている変数制約から遅延制約によって関係がある場合は，generateCompFormulae() はその関係式を必要な添字関係にして追加している．そのコンポーネントにジッタ/スループット制約によって関係がさらに追加される場合は添字の範囲までの各インスタンスについて成り立つことを確認できれば，添字変数の一般性より十分である．これは，各 QoS 式の添字の範囲の最大公約数の範囲のインスタンスを得れば等価性判定に十分であることを意味する．クラス制限の (1)，(2) より，この最大公約数として要求 QoS の範囲の値を取れば十分である．また．QoS 式の性質 (1)，(2) 式より，generateCompFormulae() で生成する式は (*) で生成する分で十分である． □

[命題 3] クラス制限を満たす問題のインスタンス I に対して構成される式 (P1) と I の検証式 $V(I)$ は等価である．

[略証 3] 命題 1,2 より題意が満たされる． □

3.7 生成される線形制約表現式のサイズ

ここでは，各 QoS 式に対しどれくらいの線形制約式が生成されるかについて解析する．

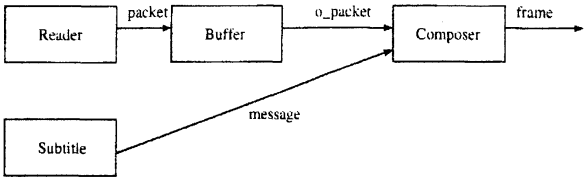

図 2 例題：ビデオプレーヤーシステム
Fig. 2 Examle: Video Player System

まず遅延式について考える．generateFormulae() がシステム出力側から逆向きにたどりながら行われているために，遅延 QoS 式の線形制約式を生成する際には，表には信号 x について (信号名 x，基本添字 j，範囲 n) が存在する．このとき生成される式は

$0 < x_j - y_{Kj+K'} \leq T$, $0 < x_{j+n} - y_{Kj+Kn+K'} \leq T$

の 2 つだけでよい．

次にスループット/ジッタ式について述べる．ただしジッタについてはスループット式の一般形

$\forall i \in \mathbb{N}: \ T' \leq x_{i+K-1} - x_i \leq T$

において $K = 2$ の場合であるのでスループット式についてのみ述べる．スループット QoS 式について線形制約式を生成する際，やはり表には信号 x についての (信号名 x，基本添字 j，範囲 n) が存在する．このとき generateFormulae() の (*) の部分よりもとの QoS 式の範囲を n' とすると n/n' 個が生成される．この値は generateFormulae() より，遅延 QoS 式についての 3 項組が生成された generateCompFormulae が呼び出されるまでの系列内で現れた遅延制約式のうち，添字関係が (i, Ki) であるものの K の総積に等しい．

以上より生成される線形制約表現式の個数は，QoS 式の数 n，全ての遅延制約式のうち添字関係が (i, Ki) であるものの K の総積 $k = \prod K$ とするとき $O(kn)$ で抑えられる．

4. 線形制約式導出例

導出の例題として，図 2 のビデオプレイヤーシステムを考える [9]．このシステムは Reader, Buffer, Subtitle, Composer という 4 つのコンポーネントから構成される．Reader はビデオストリームを生成し，パケット信号を出し，Buffer は Reader からパケットを受け取り Composer に送るバッファである．Subtitle はビデオストリームに付加する字幕を生成する．最後に Composer が Buffer と Subtitle からそれぞれパケットと字幕を受け取りフレームを生成する．この例では簡単のために，1 パケットが 1 フレームから生成されるとする．

システム要求 QoS とコンポーネント提供 QoS は表 1 とする．

表 1 各コンポーネントの提供 QoS
Table 1 Provided QoS for each component

コンポーネント名	提供 QoS
Reader	$\forall i \in \mathbb{N}: 100 \leqq RdrO_{i+1} - RdrO_i \leqq 150$
Buffer	$\forall i \in \mathbb{N}: 0 \leqq BufO_i - RdrO_i \leqq 400$
Subtitle	$\forall i \in \mathbb{N}: 500 \leqq SubO_{i+5} - SubO_i$
Composer	$\forall i \in \mathbb{N}: 50 \leqq ComO_i - BufO_i$ $\forall i \in \mathbb{N}: 50 \leqq ComO_i - SubO_i$

本来ならばコンポーネント間の通信部分（インターフェイス）にも遅延時間がかかるとして導出すべきであるがこの例では簡単のために全コンポーネント間において遅延が 0 とし，さらに，2 つのコンポーネント間の出力信号と対応する入力信号の名前を同じにしている．

各 QoS を簡単に説明する．Reader では出力信号のスループットを，Subtitle では出力信号のジッタをそれぞれ提供している．これはそれぞれ内部の動画や字幕データをフレーム用に分割したものを出力するために最低限の処理時間を要するが，処理時間に制限を課していることを意味する．Buffer は遅延の QoS を提供している．現時点ではバッファ容量などは考慮せず，単に Buffer にパケットが入ってから出るまでの時間制約を記述している．Composer は 2 種類の信号を受け取り，そのデータを合成したものをフレームとして出力する．そのため出力時刻は 2 つの入力に依存するので，2 つの遅延制約を提供している．

このときシステム要求 QoS：
$\forall i \in \mathbb{N}: 1200 \leqq SysO_{i+10} - SysO_i \leqq 1500$
の検証式を生成した．この QoS はプレイヤーシステムのスループット制約を記述している．

先に述べたように，全称子を除いた式だけでは検証することができず，この例では新たに，

- $100 \leqq RdrO_{i+K} - RdrO_{i+K-1} \leqq 150$ $(K = 2, 3, ..., 10)$
- $0 \leqq BufO_{i+10} - RdrO_{i+10} \leqq 400$
- $500 \leqq SubO_{i+10} - SubO_{i+5}$
- $50 \leqq ComO_{i+10} - BufO_{i+10}$
- $50 \leqq ComO_{i+10} - SubO_{i+10}$

を生成することで検証が可能となる．

5. おわりに

複数のコンポーネントからなるマルチメディアシステムなどのリアルタイムシステムに対し，それらの各コンポーネントに提供 QoS を課す．この提供 QoS がシステム全体が満たすべき要求 QoS を満たしているかどうかを問う問題に対して，これらの QoS を線形制約式として記述し，線形制約の非可解性判定問題に帰着することにより，QoS の一貫性を検査する方法論について述べた．

要求 QoS が「システムが指定ジッタを満たすときに，要求遅延は指定値を満たす.」などというように，条件付きで指定される場合も，前提条件部を提供 QoS と見なすことによって，本手法の適用を行なうことができる．したがって本手法で単純な QoS ばかりではなく，やや複雑な QoS も扱うことができる．

現在検証系を Lindo パッケージ [10] を用いて試作している．簡単なビデオ会議システムの例題に適用し，約 100 個の式，50 個程度の変数の制約化で 80msec. で一貫性検査ができることを確かめている．

信号のインターフェースに閉路が存在する場合に対応することなどのクラス拡張が今後の課題である．

文 献

[1] R. Staehli, F. Eliassen, J. Aagedal and G. Blair "Quality of Service Semantics for Component-Based Systems," 2nd Int'l Workshop on Reflective and Adaptive Middleware Systems, pp. 153–157, 2003

[2] R. Alur and D.L. Dill: "A Theory for Timed Automata," In Theoretical Computer Science 125 pp.183–235, 1994.

[3] L. Ageto, P. Bouyer, A. Burgueño and K. G. Larsen: "The Power of Reachability Testing for Timed Automata," In Proceedings of 18th Conference of Fundamental of Software Technology and Theoretical Computer Science (FST and TCS '98), LNCS 1530, pp. 245–256, 1998.

[4] H. Bowman, G. Faconti and M. Massink: "Specification and verification of media constraints using UPPAAL," In Proceedings of Design, Specification and Verification of Interactive Systems '98, Markopoulos and P. Johnos, editors, pp. 261–277 Springer, 1998.

[5] B. Bordbar, J. Derrick and A. G. Waters: "A UML approach to the design of open distributed systems," In Chris George and Huaikou Miao, editors, Formal Methods and Software Engineering, LNCS2495, pp.561–572, 2002.

[6] UML Profile for Modeling Quality of Service and Fault Tolerance Characteristics and Mechanisms, Request for Proposal, available at http://www.omg.org

[7] B. Bordbar and K. Okano, "Verification of Timeliness QoS Properties in Multimedia Systems," 5th International Conference on Formal Engineering Methods (ICFEM '03), LNCS 2885, pp. 523–540, 2003

[8] 加藤雄一郎，山口弘純，岡野浩三，谷口健一，"拡張時間オートマトン群による実時間システムの記述および検証，" 電子情報通信学会技術研究報告，Vol. 102, No. 616, pp. 13–18, 2003.

[9] 森一夫，岡野浩三，谷口健一，"分散マルチメディアシステムにおける Timeliness QoS 一貫性検証と時間制御コード自動導出，" 電子情報通信学会技術研究報告，Vol. 103, No. 583, pp. 13–18, 2004

[10] Lindo linear Programming Solver http://www.lindo.com/