マルチメディアデータベースにおける時間的マルチメディアオブジェクトの同期再生メカニズムとその実装法 *

7Q－3

清光英成　野中和明　増永良文
図書館情報大学 †

1　はじめに

音声オブジェクトや動画像オブジェクトなどの時間の経過と共にその属性値が変化するオブジェクトを時間的マルチメディアオブジェクトと呼ぶ．時間的マルチメディアオブジェクトの共通した性質として必ずその開始時刻 (stp) と終了時刻 (etp) があり，その間の時間幅の内に存在しているということがある．従って，時間的マルチメディアオブジェクトを閉時区間 (time interval) と捉える [1]．複数の時区間の間の関連は Allen によって調べられ equals, meets など１３種が存在する [2]．しかし，我々はこの１３種の関連が空時区間を用いることにより equals と meets のみで全て表せることを示した [3]．また，時間的マルチメディアオブジェクトのクラス定義中で他の時間的マルチメディアオブジェクトとの間の時間的関連 (temporal relationship) を ODMG-93 オブジェクトモデルで提案された参照制約の記述法 [4] を拡張して指定する概念を提案した [5]．本論文ではマルチメディアデータベースにおける時間的関連に基づく同期メカニズムとその実装を論じる．

2　時間的関連 equals と並列同期再生

2.1　個体同期再生

マルチメディアオブジェクトを再生するには所定の持続時間でその再生が完了しなければならない．例えば，ビデオオブジェクトは再生が開始されて t 秒後には 30t+1 番目のフレームを表示していなければならない．この個体同期の概念は時間的マルチメディアデータをただ一つ再生する場合ですら必要な概念である．この問題解決に対する基本的なアプローチは，再生プロセスが各々自己の再生状況を自己監視する機能を備えることである．例えば，ビデオデータの再生は毎秒 30 フレームの連続した静止画像の列をディスプレイに出力することで実現される．さらにビデオデータは圧縮して格納するのが普通である．OMEGA では圧縮されたビデオデータを伸長して表示する操作に自己監視機能を付与している．OMEGA では再生開始時刻を内部時刻 0 として各フレームに本来再生が終了するべき内部時刻 s を与えて以下のアルゴリズムでビデオオブジェクトの再生プロセスでフレームの伸長と表示の時刻を評価する．

(1) 圧縮されている第 i フレームを伸長する．

　(a) 圧縮されている第 i フレームの伸長が内部時刻 $s_i(s_i=i/30(\mathrm{sec}))$ より早く完了したなら伸長したフレーム表示を開始する。

　　(i) フレームの表示終了が s_i より早ければ s_i になるまで待って i に 1 加算して (1) を実行する．

　　(ii) フレームの表示終了が s_i 以後ならば i に 1 加算して (1) を実行する．

　(b) 圧縮されている第 i フレームの伸長が内部時刻 s_i より早く完了しなかったなら，i に 1 加算して (1) を実行する．

上記のアルゴリズムをビデオデータの最後のフレームまで繰り返すことで個体同期再生は実現できる．以下に上記の例を XIL(X Image Library) を用いて JPEG 形式で圧縮して格納されている動画像を再生するプログラムの例を示す．

```
while (xil_cis_has_frame(cis)) {
  xil_decompress( cis, image );
  if ( current_internal_time() < s[i] ) {
     xil_rescale(image, image, scale, offset);
     xil_ordered_dither( image, disimage, colorcube, dmask);
     while ( current_internal_time() < s[i] ) { } }
  i++;
  }
```

個体同期のメカニズムはビデオや音といったメディアが違えば異なるものである．そこで OMEGA ではクラス Video，クラス Sound といった異なるメディアのクラスのメンバ関数として定義しておく．

2.2　開始同期再生

$m_1, m_2, .., m_n$ $(n \geq 2)$ を equals 関連にある n 個の時間的マルチメディアオブジェクトとし，それらを再

*A Synchronization Mechanism of Temporal Objects and its Implementation on a Multimedia Database System.

†Hidenari Kiyomitu, Kazuaki Nonaka and Yoshifumi Masunaga, University of Library and Infoemation Science, 1-2 Kasuga, Tsukuba, Ibaraki 305, Japan

生するプロセスを p_1, p_2, ..., p_n とする．それぞれの再生プロセスを記述した関数は引数として再生開始時刻 sp を代入するための共有メモリへのポインタをとる．$p_i(1 \leq i \leq n)$ が m_i の再生準備を完了したならば属性 status（初期値 0）に 1 を代入して他の全ての再生プロセス $p_j(1 \leq j \leq n, j \neq i)$ が m_j の再生準備を完了したかどうかを関数 status_check() を実行してチェックする．関数 status_check() は，局所変数 c（初期値 1）を用いて全ての m_j に対して c=c × m_j.status を計算する．c の値が 1 にならなければ，$p_j(j \neq i)$ のうち少なくとも 1 つは m_j の再生準備を完了していないことになるので，sp の値を初期値のまま m_i の内容表現オブジェクトの個体同期再生プロセスを実行する．このプロセスは sp が初期値の場合は他の値が代入されるまで開始されないようにプログラムされている．c の値が 1 になれば，p_i が最後に再生準備を完了したことになるのでその時刻を sp に代入して m_i の内容表現オブジェクトの個体同期再生を開始させる．これは複数の個体同期再生プロセスに同一の開始時刻を与えることにより，その開始時刻を用いて複数の個体同期再生プロセスが並列に実行されることで実現される．なお，特に同じ形式の音声オブジェクトの開始同期再生は，例えば，8KHz, 8bit でサンプリング・量子化された 2 つの u-law 音声オブジェクト u_1, u_2 は，u_1 の波形に u_2 の波形を重ねた u-law 音声オブジェクト u_{12} を equals(u_1, u_2) 演算により生成して再生するので開始時刻のずれはなくなった．これらの開始同期を保証するメカニズムは時間的関連が定義されたクラスでメンバ関数として定義している．

3 時間的関連 meets と直列同期再生

2 つの時間的マルチメディアオブジェクト m_1 と m_2 の間に meets 関連が定義されているなら m_1 と m_2 の再生はまず m_1 が再生を開始され，その終了を受けて直ちに m_2 の再生が開始されなければならない．これを直列同期再生という．勿論 m_1 と m_2 の再生は個体同期の対象でもある．直列同期再生時の問題点は再生の切り替わりのタイミングを正確に保証できるかどうかという点である．つまり，m_1 の再生が終了した時点で直ちに m_2 の再生を開始できるか，その間に遅れをとることはないか，ということである．問題を 2 つに分けて考える．

(1) m_1 と m_2 が共に同一のメディアデータである場合．

(2) m_1 と m_2 が異なるメディアデータの場合．

解決法は再び空時区間を使用して統一的な対処をするということである．つまり，(1) の場合，meets を合成演算子と捉え，時間的複合マルチメディアオブジェクト meets(m_1, m_2) を生成する．すると，これは直ちに個体同期再生の対象となり，また個体同期が正常に作動すれば結果として m_1 から m_2 への再生の切り替えがきちんと行われたこととなる．(2) の場合，一般に m_1 の再生を終了してから m_2 の再生に切り替わるまでに遅れが生じよう．そこで m_1, m_2 とそれぞれメディアの等しい空時区間オブジェクト n_1, n_2(n_1.stp=m_2.stp ∧ n_1.etp=m_2.etp, n_2.stp=m_1.stp ∧ n_1.etp=m_1.etp) を用いると meets(m_1, m_2) *iff* equals(meets(m_1, n_1), meets(n_2, m_2)) であるので meets 演算を行って開始同期再生すればよい．つまり，異なるメディアの直列同期再生は空時区間オブジェクトを用いると個体同期メカニズムと開始同期メカニズムがあれば実現できるのである．

4 まとめ

本論では時間的マルチメディアオブジェクトの同期再生について議論した．この議論の基本となるのは、時間的複合マルチメディアオブジェクトの同期再生は，空時区間オブジェクトを使用すれば，開始同期再生メカニズムと個体同期メカニズムを備えていることが必要かつ十分であるということである．従って，個体同期再生の実装例を示し，開始同期再生のメカニズムを明らかにした．

< 謝辞 >
本研究に貴重なご意見を数多く頂いた群馬大学金森吉成教授と金森研究室の諸氏に感謝します．また，日頃有益な御議論を頂く増永研究室の諸氏に感謝します．

参考文献

[1] Masunaga, Y. : A Temporal Expansion to Multimedia Object Model in OMEGA, Preoceedings of DASFAA'95, pp. 430-440, 1995.4

[2] Allen, J. : Maintaining Knowledge about Temporal Intervals, Communications of the ACM, Vol.26, No.11, pp.832-843, 1983.

[3] 増永良文, 清光英成 : 時間的マルチメディアオブジェクトの同期表現と実装法, Proceedings of Advanced Database System Symposium'95, pp. 79-85, 1995

[4] Cattell, R. (ed.) : The Object Database Standard, ODMG-93 Release 1.1, 176p. , Morgan Kaufmann, 1994.

[5] 増永良文, 清光英成 : マルチメディアオブジェクト間の時間的関連記述の一フレームワーク，電子情報通信学会論文誌, Vol. J79-DII, No. 4 に掲載予定