工学博士　学位論文

)300

分散 マルチメディア システム における同期方式に関する研究

１９９４年１２月２０日

指導教官　　齊藤 忠夫 教授

東京大学大学院　工学系研究科

電子工学専攻　　27112

大野　隆一

もくじ

図一覧

第 1 章

序論

1.1 はじめに

近年，ネットワーク技術及びコンピュータ技術の発展に伴い，ネットワークを介してマルチメディア情報を送ることで多地点間での会議，教育など様々な活動を支援するシステムの研究・開発が盛んに行なわれてきている．このようなシステムにおいてはネットワークの遅延及び遅延変動，各ノードの負荷変動などの影響を避け，質の良い情報を同時に提供することが重要となる．

ネットワークの遅延変動などは各メディア内のフレーム到着の変動，複数メディア間の同期ずれなどを引き起こす．このようなメディア到着変動は受信側のノードにバッファを設け，受信したメディア情報をいったんバッファに蓄え，ある時間間隔をおいてからユーザに提示することで和らげることができる．

ある一つのノードから他の一つのノードにマルチメディア情報を転送するような場合にはこのような手法でメディア到着の変動を吸収できるが，多地点間の会議や教育を支援するシステムなどのように共通の情報が複数のノードにおいてほぼ同時に提示されるようなシステムにおいては，単に送られてくるメディアの到着変動を吸収するだけでなく，複数のノードで発生する（生の声などの）様々なメディア情報間の同期を取ることも必要となる．

本論文では，このような多地点間で共通の情報がほぼ同時に提示されるような (会議型の) システムにおけるメディア間同期の問題を取り上げる．

このようなシステムで扱われるメディア情報としては以下のようなものが考えられる．

RMS 予めディスク，テープなどの蓄積型情報源に蓄えられていてそこから引き出されるメディア情報（Retrieved Media Stream と名付ける．以下 RMS と略称する．）．教育用教材など．一時停止，再生などの命令により適宜その情報提示位置を変化させながら各ノードで提示される．

LMS 参加者の声などの生のメディア情報（Live Media Stream と名付ける．以下 LMS と略称する．）．

本研究では LMS と RMS を分けて考える．RMS を各ノードでバッファリングし，そのバッファ上の情報提示位置を適切に定めることで各ノード間の情報提示位置の関係が決まる．さらにこのノード間の情報提示位置の関係により LMS と RMS が各ノードで提示される際のタイミングが決まる．つまり，各ノードにバッファリングされるメディア情報を用いることで多地点環境でのメディア間の同期を取ることが可能となる．RMS が

予め存在する情報であるために予めバッファリングしておくことが可能であるのに対し，LMS は生の情報であるためにできるだけ早く提示されることが望ましいといった RMS と LMS の性質の違いがあることからこの方法が適用しやすいことになる．

本論文では，まず本研究で対象とするシステムの構成を示す．次に，予めネットワーク，端末などの資源を確保することで QOS が保証される環境におけるシステムの設計条件を示す．次に，LAN 環境において構築した試作システムにおいて負荷変動に柔軟に対応しつつ情報提示の際のメディア間の同期を維持する機構，及び，その評価を示す．さらに，本研究で行なった検討を踏まえ，分散マルチメディアシステムにおけるメディア間同期を実現するために必要な諸技術について再検討する．

1.2 本論文の構成

以下に本論文の構成を示す.

第 2章では，以降の章での基礎的な知識として，マルチメディアの定義，コンピュータにおけるマルチメディア，マルチメディア通信，マルチメディアにおけるメディア間同期について概説する.

第 3章では，本研究で対象とするシステムの構成を示す．まず，多地点間での会議，教育などの活動を支援するシステムにおける諸技術について触れ，その中での本研究の目的につてい述べる．さらに本システムの目的，構成を示す.

第 4章では，予めネットワーク，端末などの資源を確保することで QOS が保証される環境におけるシステムの設計条件を示す．まず，第 3章で定めた同期のセマンティクスを満たすために各ノードにおいて必要なバッファ量の算出法を示す．さらに，これを用いて，バッファサイズとコマンド応答時間の間の設計条件を明らかにする．また，各ノードにおいてバッファの大きさが定められている場合の設計条件についても検討する.

第 5章では，LAN 環境において構築した試作システムにおいて負荷変動に柔軟に対応しつつ情報提示の際のメディア間の同期を維持する機構を示す．また，本同期機構の評価についても述べる.

第 6章では，本研究で行なった検討を踏まえ，分散マルチメディアシステムにおけるメディア間同期を実現するために必要な諸技術について再検討する.

第 7章では，以上の結果をまとめて結論を述べ，今後の課題を示す.

第 2 章

背景

本章では研究の背景となる基礎的な知識として分散マルチメディアシステムを構成する上での諸技術について述べる．また，マルチメディアにおけるメディア間同期に関するこれまでの研究を紹介する．

2.1 マルチメディアとは

2.1.1 定義

ここ数年，“マルチメディア”という言葉を頻繁に耳にするようになってきた．しかし，この言葉の意味するところは研究者の間でさえ統一されているとは言い難く，使う人の都合により様々に解釈されているのが現状である．

MHEG(Multimedia and Hypermedia information coding Expert Group)[Yasu 91] では，“メディア”という言葉は以下のように定義される．

メディアム 情報が知覚され，表現され，蓄積され，または，伝えられるための手段．

知覚メディアム ユーザにより知覚される情報そのもの．

表現メディアム 相互交換されるデータの種類．符合化された形式で記述される情報そのもの．

表示メディアム ユーザに対して情報を再現させたり（出力装置），ユーザからの情報を取り込んだり（入力装置）するために使われる物理的手段．

蓄積メディアム データを蓄積するための物理的手段．

伝送メディアム データを伝送するための物理的手段．

相互交換メディアム データを相互交換するための手段．これは，蓄積メディアム，伝送メディアム，またはその組合せでよい．

また，“マルチメディア”という言葉は以下のように定義される．

マルチメディア 複数種類の表現メディアを扱う特性．

つまり，この定義ではテレビ放送用の映像情報のように知覚メディアが複数でも表現メディアが一つしかない場合，マルチメディアではなくモノメディアと呼ぶ．

この定義だけでは，メディアが複数あればマルチメディアという印象を与えかねない．そこで，さらに言葉の意味する範囲を狭め，本論文で扱うマルチメディアを以下のように定義する．

メディア 人にどのようにインタフェースするか考えた上で形式付けられた抽象的な情報表現媒体

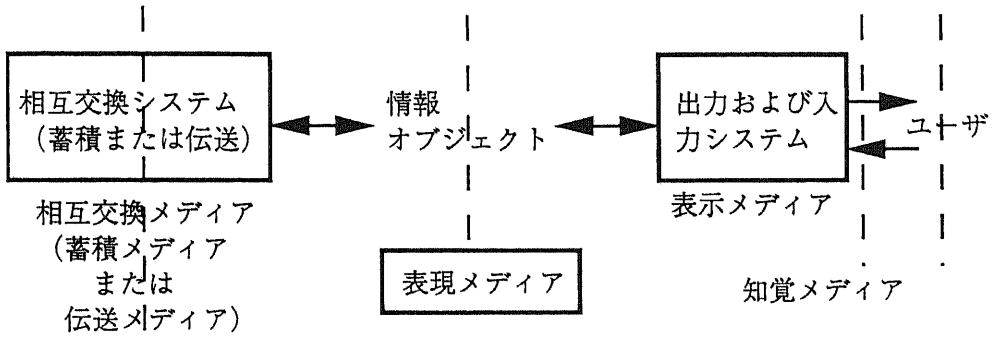


図 2.1: メディアの定義

マルチメディア 予め互いに関連付けられた複数のメディアからなり，人とインタフェースする時にメディア間の同期を取ることで全体として一つの意味をなす情報表現媒体

2.1.2 マルチメディアの構成要素 [Yasu 91, Hara 91]

マルチメディアを構成する個々のメディアとしては，動画，音声など，様々なメディアが考えられる．ディジタル技術の発展により，現在，これらのメディアはすべて 1 と 0 のディジタル情報として同一のコンピュータ，及び，ネットワーク上で蓄積，交換されるようになってきた．

以下に，代表的なモノメディアを示す．

画像

画像は情報量が大きく，蓄積，交換などのために様々な圧縮技術が研究，開発されてきた．

静止画のように，各フレームごとにフレーム内での冗長性のみを利用して圧縮を行なう方式としては JPEG 方式が標準化されている．

また，動画のようにフレーム間での冗長性も利用して圧縮を行なう方式としては MPEG 方式 [Gall 91] が標準化されている．

図 2.2に MPEG における画像間予測を示す． MPEG では映像信号は I(Intra-coded) ピクチャ， P(Predictive-coded) ピクチャ， B(Bidirectionally predictive-coded) ピクチャの 3 種類のピクチャとして符号化される． I ピクチャは予測を使わずに入力信号をその

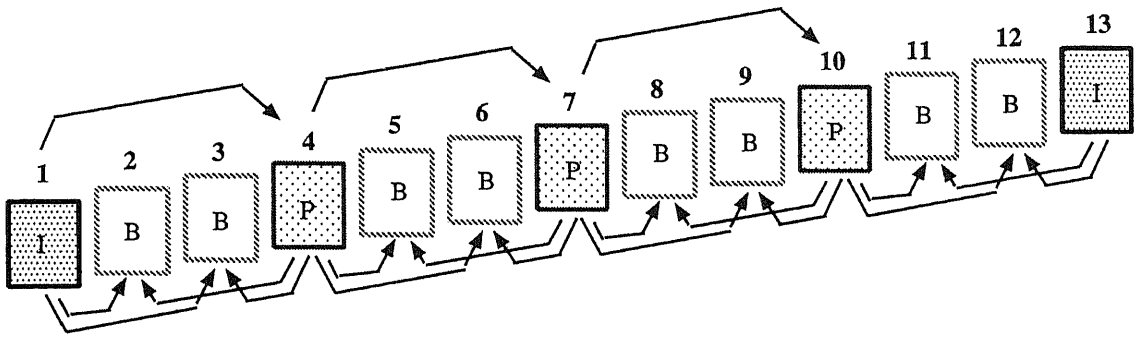


図 2.2: MPEG

まま符号化するフレームで，編集点やデコード開始点として利用できる．P ピクチャは一方向の動き補償予測を用いるフレームで，B ピクチャは双方向予測を用いるフレームである．

音声

これまでの通信ネットワーク（電話回線）で最も良く利用されてきたメディアが音声である．音声はパルス符号変調 (PCM) により 6 4 kb/s のディジタル信号となる．音声は画像に比べ情報量がかなり少ない．

CG

CG(Computer Graphics) は 3 次元世界をモデル化し（数値データで表す），その投影像として画像を生成する．CG は画像生成にかなりの CPU パワーを必要とするため，専用のハードウェアなどが用いられる．蓄積，交換などに必要な情報は command 主体のものとなるためその情報量自体は動画に比べ少なくなると考えられる．

Text

これまでのコンピュータシステムにおいてはキャラクタベースの Text 情報が最も良く利用されてきたメディアと言えるだろう．Text は文字コードで構成され，画像などに比べて扱う情報量もかなり少ない．また，冗長性の無い情報であり，bit 誤りは許容されない．

2.1.3 マルチメディア標準 [ISO 93, Kretz 92, NHK 94, Yasu 91]

現在，国際標準化機構の技術委員会 ISO/IEC JTC 1 においてマルチメディア／ハイパーメディア技術を扱う様々な標準化活動が進められている．

以下に，代表的なマルチメディア関連の標準規格を示す．

HyperODA

ODA(open document architecture, 開放型文書体系) はマルチメディア文書の表現形式の ISO 国際規格である．

文書にはページやページ内の領域などのレイアウト的な構成と，章立て・段落・図などの論理的な構成の 2 面がある．ODA では，文書をレイアウト的および論理的な両視点から分割し，階層的なツリー構造によって構造化する．

HyperODA は ODA の拡張であり，紙に印刷する文書にはない機能として，文書内や文書間で関連する項目を自由に参照できるように情報をリンクできる機能などを追加したものである．

HyTime

HyTime(Hypermedia Time-based structuring language) は文書構造表現の ISO 国際規格である SGML をベースとし，ハイパー文書を表現するための基本モデル，及び，記述言語を定義する ISO 国際規格である．

HyTime の特徴は，ハイパー文書内の構造情報と描出 (rendering) 情報とを区別して扱い，構造情報を交換対象の中心とするところにある．ハイパー文書にとった本質的な情報であるマルチメディアオブジェクト間の結合関係，位置関係，参照関係等については HyTime で記述しておき，装置に依存するような詳細な表示情報の扱いはアプリケーションに任せる．

MHEG

MHEG はサービス，アプリケーションの中で相互交換するマルチメディア・ハイパーメディア情報オブジェクトの符号化表現を規定することを目指した国際規格である．ここでいう相互交換には，蓄積，テレコミュニケーション，放送網も含めたすべてのものが含まれる．

MHEG では，マルチメディア・ハイパーメディア情報を，“動画像，静止画像，音声，文字，図形，MIDI などから構成されるある一つの単位にまとめた情報” ととらえ，マルチメディア・ハイパーメディア情報の特徴であるメディア間の関係づけ機能（同期およびリンクの表現）をサポートした上で，その情報を種々のシステム間で相互交換するための符号の規格化を行なっている．
例えば，JPEG，MPEG 映像などのデータは，MHEG では “コンテントオブジェクト” の要素として扱われる．また，オブジェクト間の関係を記述するオブジェクト（リンク，スクリプト），これらを複合した “コンポジットオブジェクト” を規定している．MHEG オブジェクトの例を図 2.3に示す．

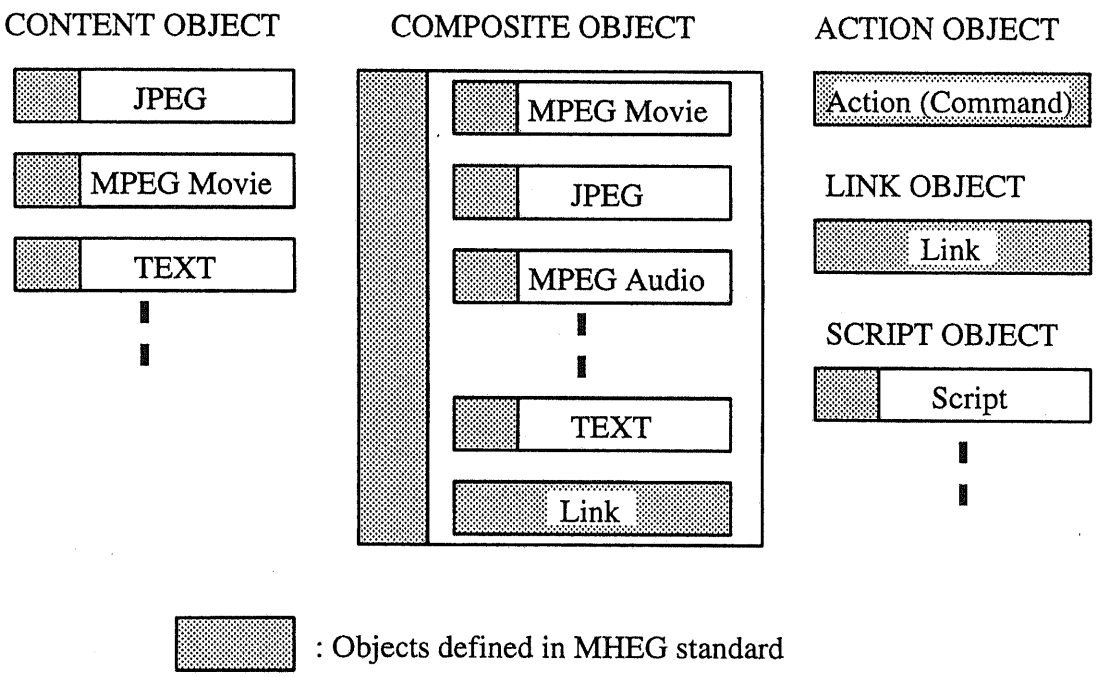


図 2.3: MHEG オブジェクトの例

図 2.3において，リンクは同期関係，条件動作などを記述する．一つの MHEG オブジェクトは指定されたトリガ条件（例えばマウスのクリック）を満たすと，他の MHEG オブジェクトに対してアクション（コマンド）を送る．このようにして図 2.4に示すようなマルチメディア同期の表現が可能となっている．

また，コンテント（コンポジット）オブジェクトを実際にユーザに提示する際の表現動作を規定するものとして以下の 2 つがある．

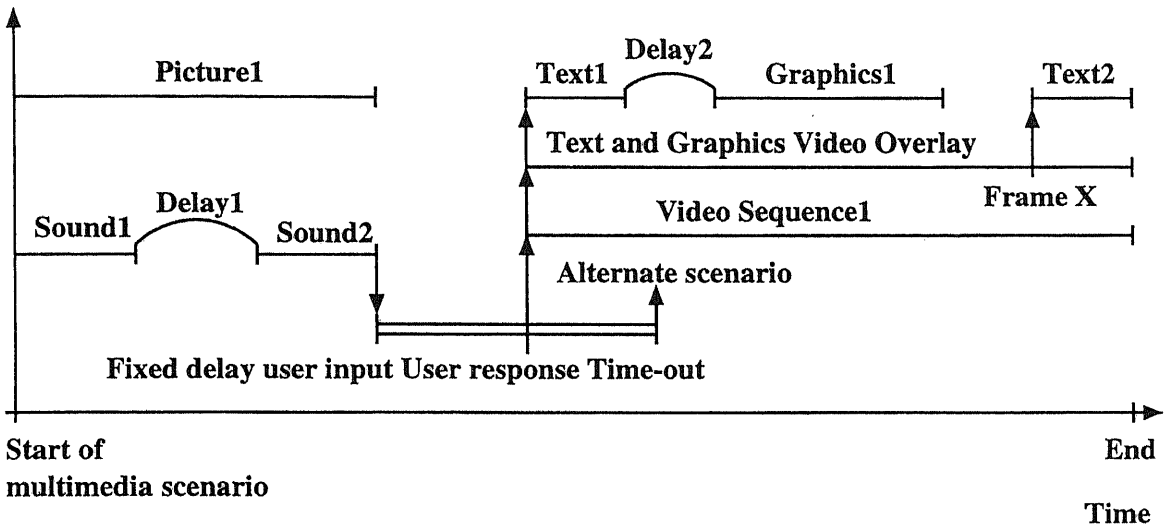


図 2.4: MHEG におけるマルチメディア同期の例

プレゼンタブル (presentable) コンテントオブジェクトの表現動作の規定を行なう.

トゥリー (tree) コンポジットオブジェクトの表現動作の規定に用いられる.

様々なプレゼンタブル, トゥリーなどを用いることで, 同じコンテント (コンポジット) オブジェクトを再利用して異なる情報提示を行なうことが可能となる.

次に, MHEG オブジェクトの状態制御機能について説明する. 図 2.5にオブジェクトの状態管理図を示す. この図における Ready 状態と Not Ready 状態は通信路を介して MHEG オブジェクトをやりとりするような場合のために設けられた状態で, Not Ready 状態は必要なデータがまだ database などの遠隔ノードにあり, ユーザノードには入っていない状態を示している.

Not Running と Running はオブジェクトの表示状態を示している. また, Running の際には, 外部 (ユーザ) からのインタラクションに対応する機能として Select (マウスでのメニュー選択など) と Modify (数値などのテキスト入力) がある.

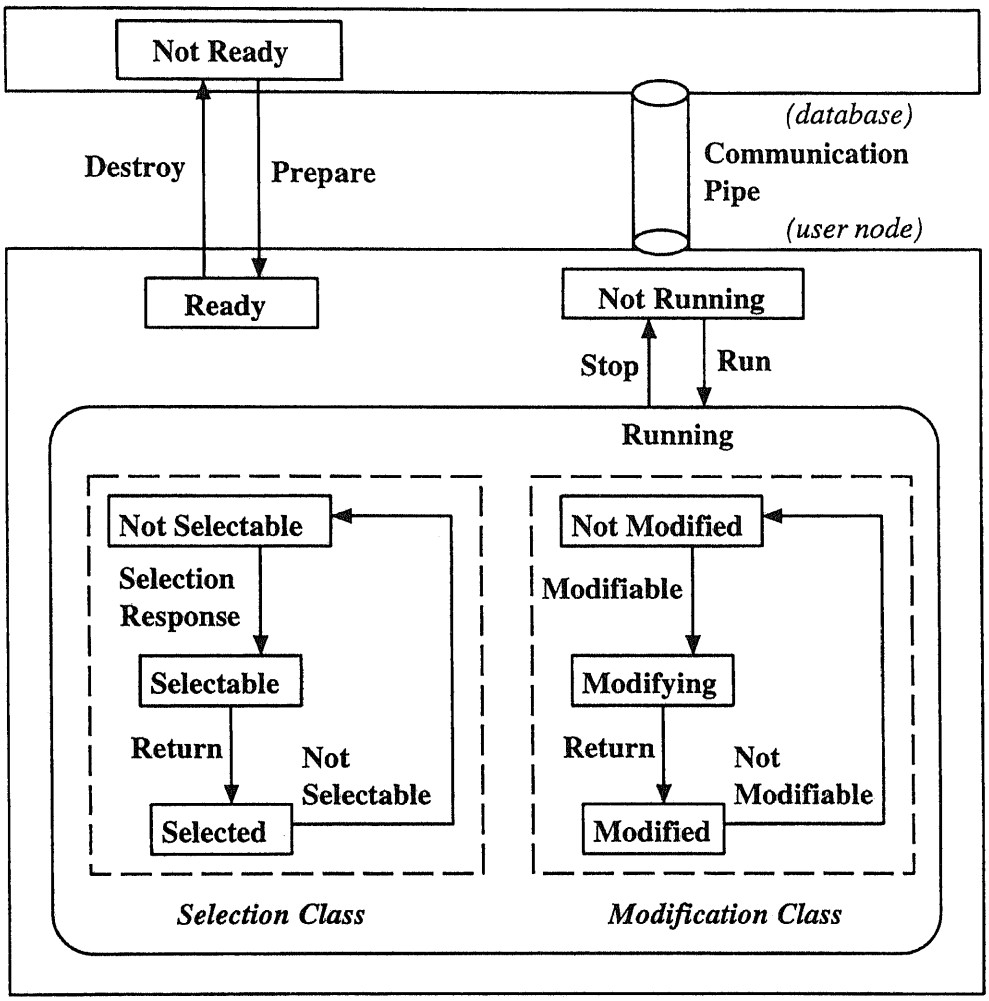


図 2.5: State Transition Diagram of the Preparation Status

次に，相互交換システムにおける MHEG の位置付けを示す．図 2.6において Script と書いてある部分は MHEG オブジェクトに対して同期，リンクよりも複雑な操作を実現するための記述言語として現在標準化が進んでいる．

MHEG object with its content data と書いてある階層が MHEG 標準化案で規定される MHEG オブジェクトの部分である．また，Non MHEG content data と書かれている階層は MPEG,JPEG などの個々のメディアの符号化標準である．

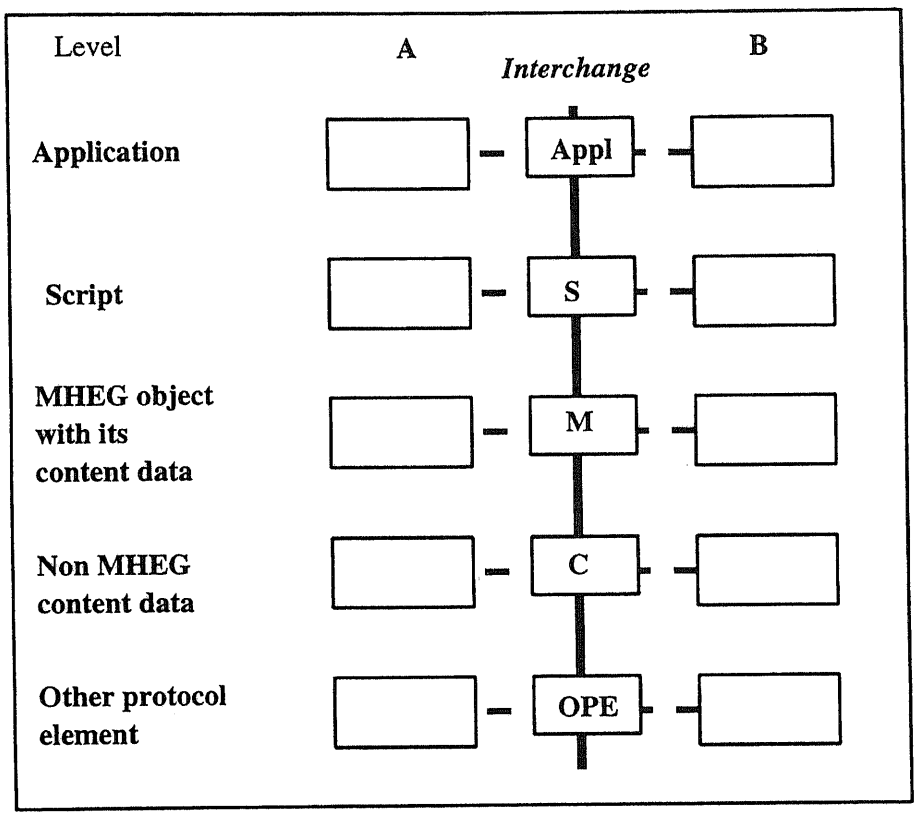


図 2.6: Interchange of MHEG Objects

MHEG object with its content data の部分の相互交換についてさらに詳しく示したのが図 2.7である.

図 2.7において MHEG engine と書いてあるのが MHEG 階層での MHEG object のデコード, 交換などの処理一切を取りし切るプロセスである.

図 2.8に MHEG エンジンの機能（一例）を示す. ただし, MHEG エンジンは MHEG 標準案中で規定されるものではない. つまり, MHEG エンジン自体にどのような機能を持たせるかは MHEG 標準の範囲外である.

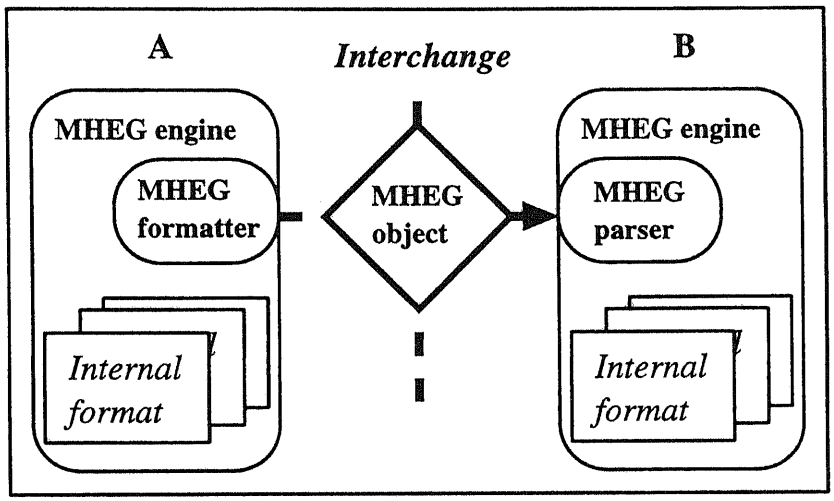


図 2.7: Conceptual Model of a MHEG System

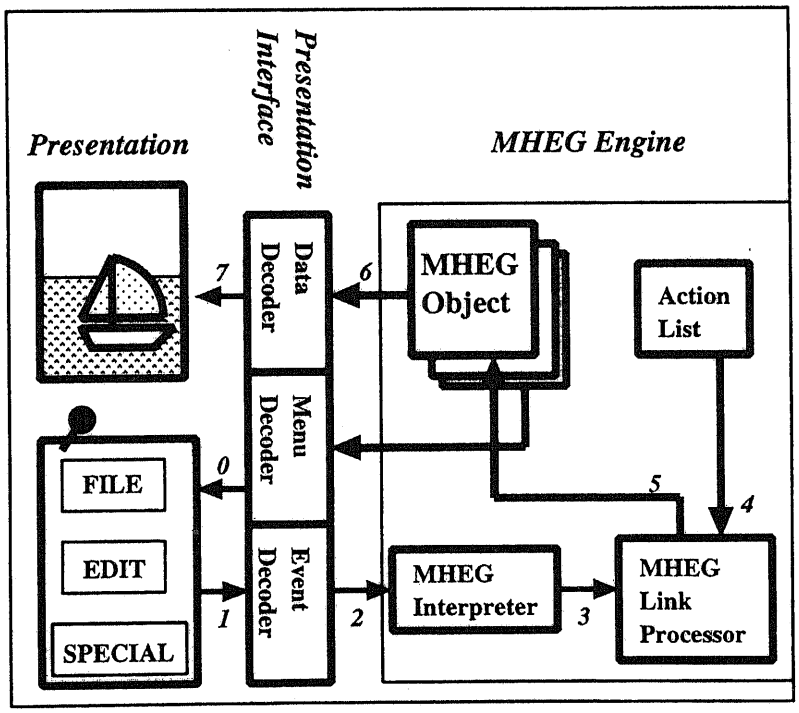


図 2.8: an example of MHEG engine

2.2 コンピュータにおけるマルチメディア

CD-ROM 装置や画像圧縮ハードウェアの普及に伴い，パーソナルコンピュータなどにおいても従来の文字以外に音声，映像などの様々なメディアを容易に扱えるようになってきた．

従来のコンピュータは，その用途が主に数値計算などの計算処理に置かれていたため，高速化の追求が最重要視されてきた．しかし，コンピュータをマルチメディア端末として捉えた場合，ただ速いというだけでなく，ユーザが望む品質のマルチメディア情報を提示することを考慮する必要がある．

例えば，UNIX ワークステーションにおいて動画や音声を再生中に他のプロセスを起動すると，音声の再生が途切れる，動画の再生が乱れる，メディア間の同期が合わなくなるなどといった様々な問題が起こり得る．このようなマルチメディア情報の提示の際のメディア品質 (QOS) の劣化を与えられた (CPU などの) 資源でいかに最小源にくいとめるか，つまり，与えられた資源制約の下 (できるだけ) ユーザの望むような品質のメディア情報を提示することがコンピュータがマルチメディアを扱う上で最も重要な点であると考えられる．

また，将来のマルチメディアシステムではユーザが希望するメディア品質 (QOS) を指定し，その QOS に見合った品質のメディア情報が提供されるといった形態のサービスが実現可能となるであろう．そのようなシステムにおいては，ユーザからの QOS を保証できるだけの資源をシステム側が確保する必要がある．

以下，まず，資源制約の下メディア品質の劣化を最小限に食い止める手法について述べる．次に，CPU などの資源を予め確保することで（ユーザにとっての）QOS を保証するための QOS 保証について述べる．

2.2.1 Graceful degradation

同じアプリケーションを異なる性能のマシーンで動かす場合，数値計算のようなアプリケーションでは計算時間に差が生じることになる．しかし，マルチメディア情報を提示するようなアプリケーションを動かす場合，遅いマシーンでは意図した時間に各メディアの情報が提示されないために，ユーザにとっての QOS が大幅に悪化するなどといった結果が生じる可能性がある．同様なメディア品質の大幅な劣化は，UNIX のようなマルチタスク OS において他のプロセスの影響で負荷が大きくなったような場合にも生じ得る．

このような，性能の低いマシーンや，負荷が大きいマシーンにおいてもできるだけユーザへの QOS を劣化させずにそれなりのメディア情報を提示するためには，CPU 能力の不足に対しどのように情報提示を劣化させるかが重要となる．

アップル社が開発した QuickTime[Asa 91, Apple 92] ではマルチメディア情報を扱うために時間管理の概念を導入し，CPU 能力の不足に対しては動画データの表示を間引くことで様々な性能のマシーンに共通の OS プラットフォームを提供する．IBM 社の MMPM/2 (Multimedia Presentation Manager/2) では CPU 能力の不足時に音声処理と動画処理への CPU 時間を適切に割り当てることで過負荷時に対応する．

以下に，代表的なマルチメディア OS である QuickTime における時間管理機能を紹介する．

QuickTime

QuickTime はアップル社が開発した OS のマルチメディア拡張であり，従来の OS には無かった機能として時間の経過を表現した新しいファイルフォーマットを定義して，それを使って同期制御などの時間を管理するメカニズムを提供する．

Movie ファイル・フォーマット

QuickTime では時間の経過を表現できるファイル・フォーマットとして Movie ファイル・フォーマット (図 2.9) をサポートする．

Movie ファイル・フォーマットでは，マルチメディア・データを表現するデータ本体とそれに伴う情報を分離して表現する．分離して記録することによって，Movie ファイルのところだけをコピーすれば，あたかもマルチメディア・データ全体をコピーしたように扱える．また，データを共有する部分を持った Movie ファイルを定義することもできる．このフォーマットにより，動画データなどのように容量の大きなファイルをコピーする際に動画データ本体のコピーが必要なくなる．

以下，図 2.9中の Movie,Track,Media と書かれている領域について説明する．

Movie ファイルには，その予告編を表すプレビュー (Preview)，アイコン表示などに使う静止画のポスター (Poster) を定義できる．ここには，実際の動画データのどの部分を使うか記述する．

Track と呼ぶ領域は，映像や音声といった個別の情報を記述する．Movie ファイルでは，Track を何本持ってもよい．これにより，一つの Movie ファイルが複数の

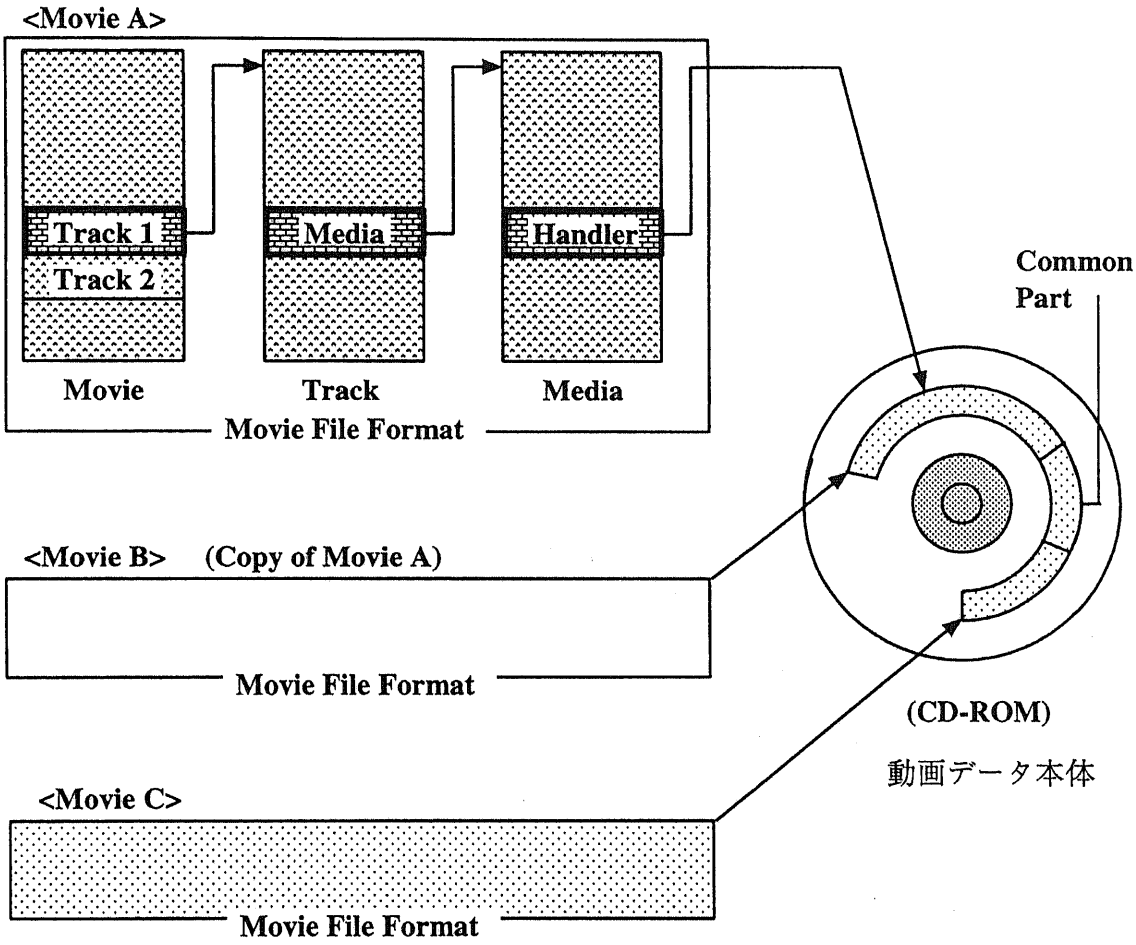


図 2.9: Movie File Format

映像データや音声データを管理することが可能になる.

また, 複数の Track をグループ化 (Alternate Group) することもできる. 例えば, ある動画の音声についての英語の Track と日本語の Track を用意する. この二つの Track をグループとして表現する. 再生時にどちらの Track を使うのかも別に記述する.

Media と呼ぶ領域では, データ本体が実際にどの Media に格納されているか記述する. データ本体の物理的な位置情報は Media ハンドラに記述する.

時間管理のメカニズム

Movie ファイル・フォーマットには，映像や音声などのデータにそれぞれ TimeScale（基準となる時間の尺度），Start Time（開始時間），Duration（総時間）と呼ぶ時間に関する三つの情報を記述している．QuickTime はこれらの情報を基にパソコンの基準クロックを使って時間を管理する（図 2.10).

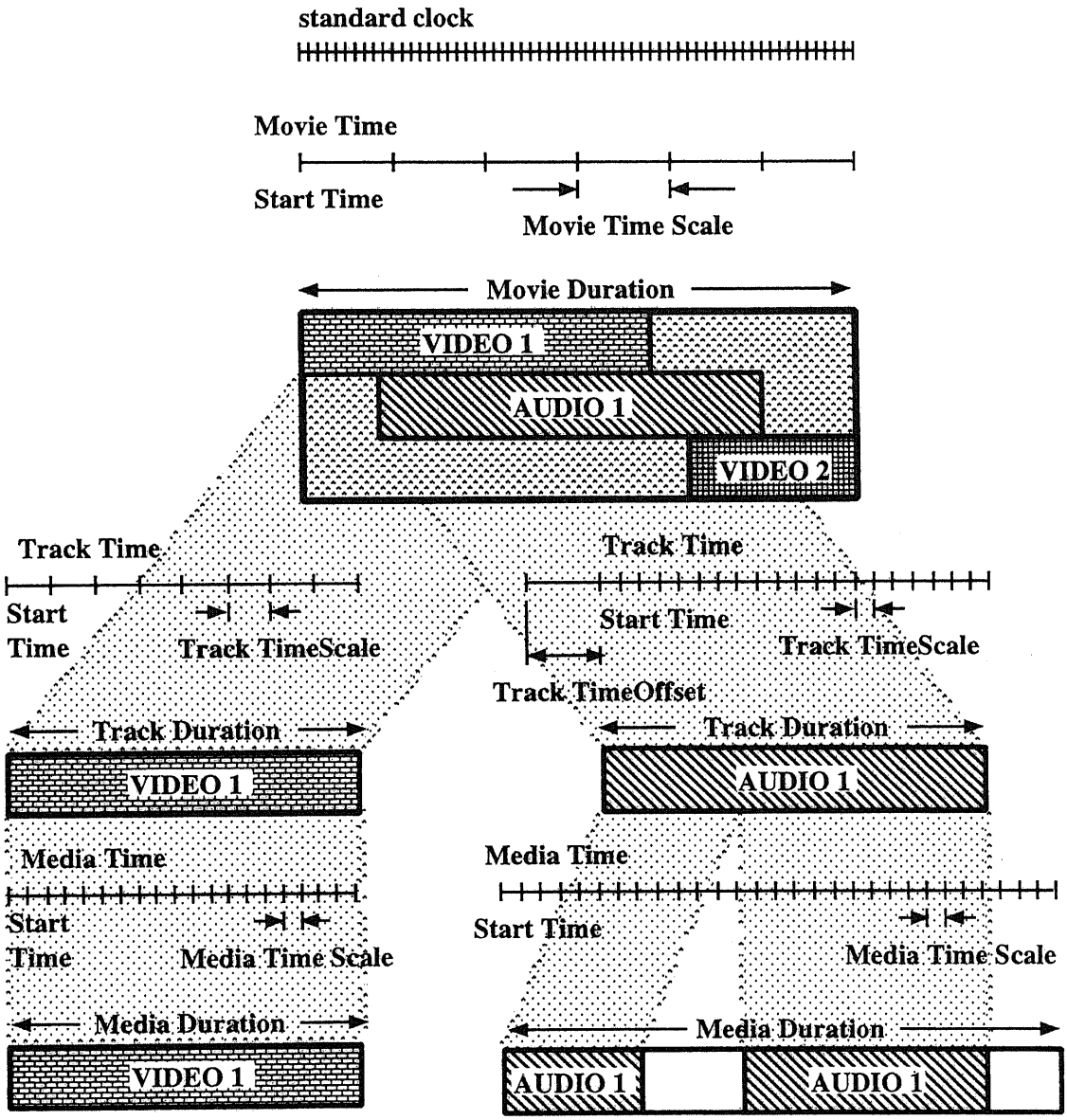


図 2.10: 時間管理のメカニズム

ただし，TimeScale は，すべてのデータの間で同じ尺度というわけではない．QuickTime は，各 TimeScale をまとめたうえで新しい TimeScale(Movie TimeScale)

を設定し直して，時間を管理する．

さらに，QuickTime は既に述べたように，処理速度が遅いパソコンでは動画データを間引いて再生することで違和感がないように情報提示を行なう仕組みを備えている．

2.2.2 QOS 保証

ユーザへの情報提示の際のメディア品質を保証するためには，予め CPU，ネットワークなどの資源が（要求された QOS を満たすのに十分な量以上）使用できることが前提となる．

特に，UNIX のようにマルチタスクの環境では複数のアプリケーションが同じ資源を共用することになる．このような環境では，各々のアプリケーションに CPU などの資源をどのように割り振るかにより，各アプリケーションがユーザの要求する QOS を満たすことができるかが決まってくる．

図 2.11にユーザからの QOS 要求をシステム側でどのように保証するかの大まかな枠組 [Toku 93] を示す．

まず，ユーザ（アプリケーション）が，OS に対して（ユーザにとっての）QOS を要求する．この要求に応じて，OS が CPU 資源を割り当てる．OS はネットワーク・プロトコルに要求を伝える．ネットワークプロトコルが伝送路の資源を確保する．さらに，このような要求に対して資源が確保できるか否かを，各部がそれぞれの上位部に伝える．

資源確保ができない場合の対応としては，資源割り当てを拒否する，既に起動されているアプリケーションも含めたアプリケーション全体の QOS を少しずつ劣化させることで新たに起動されたアプリケーションに資源を割り振る，などといった幾つかの選択枝が考えられる．

このような QOS 保証を実現するためには，CPU などの資源を各タスクにどれだけ割り振れば各タスクがどのような挙動を示すかといったことを予測できることが大切である．そのためには，コンテキスト切替え，プロセス間通信，共有資源へのアクセスの際の同期などの個々の処理にかかる時間が予測可能であることはもちろんのこと，各資源を各タスクに割り当てる際のスケジューリングが各タスクの QOS を反映して，各タスクの（時間制約などの）QOS を満たす方向に働くことが重要となる．

以下，このような QOS 保証を実現する上で欠かせない CPU のスケジューリング方式について述べる．

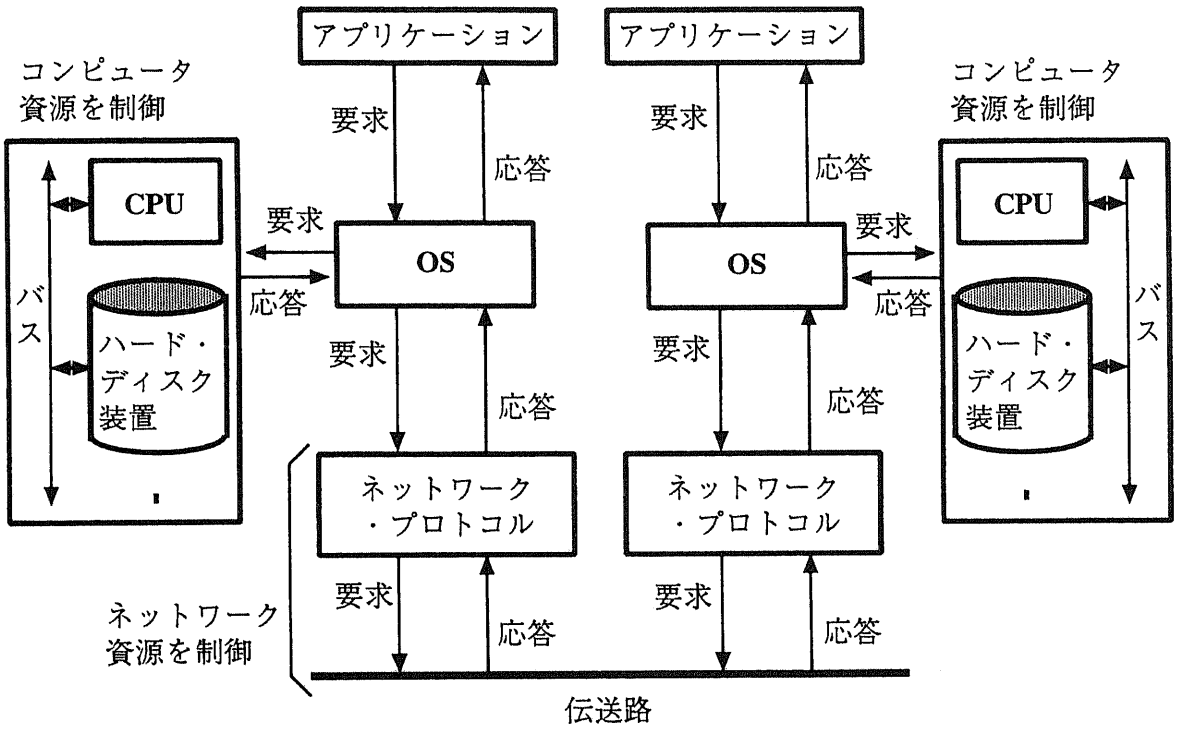


図 2.11: QOS 保証

スケジューリング方式

従来の UNIX ベースのオペレーティングシステム [Bach 86, Leff 89] では各プロセスは時間交代で CPU 資源を使用する．CPU を使うタスクが交代する順番は，ラウンド・ロビン方式に基づく．一定時間が経過すると，次のタスクに CPU を受け渡す．スケジューラは各タスクを平等に扱い，時間的な制約が厳しいプログラムに CPU 資源を優先的に割り当てるといった機能はない．つまり，ユーザからの QOS を反映させるべく，CPU 資源を適切に各タスクに割り振ることはできない．

それに対し，リアルタイム OS[Toku 89] では各タスクにデッドラインを設定でき，デッドラインを設定したタスクの処理を優先させることができる．

現在，このようなリアルタイム OS をベースとし，QOS を保証する機能を付け加えることでマルチメディアに対応した OS を構築しようといった研究が幾つかの研究機関で行なわれている [Toku 92, Ander 90, Ander 93]．

このような OS における代表的な CPU のスケジューリング方式として rate monotonic スケジューリング [Lecho 89] がある (図 2.12)．

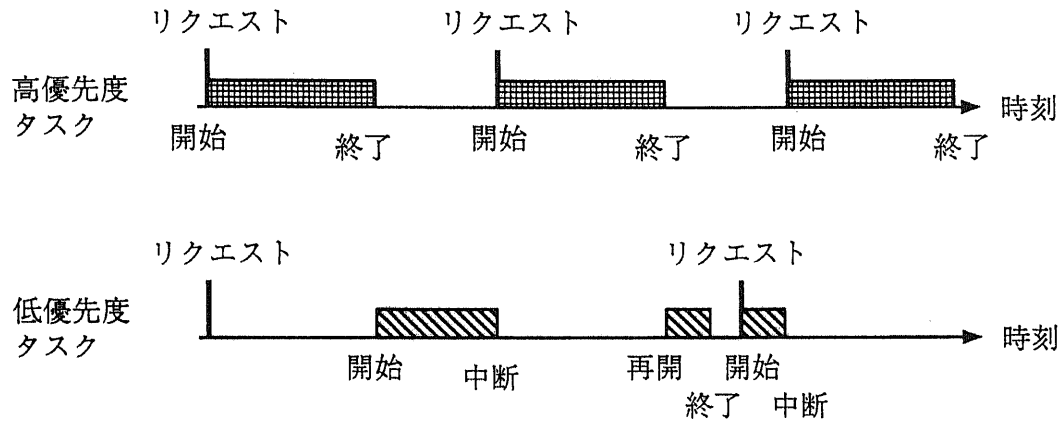


図 2.12: Rate Monotonic スケジューリング

rate monotonic スケジューリングはセンサ情報の収集のようにリクエストが周期的に発生するタスクを対象とし，周期の短いタスクほど高い優先度を与える preemptive なスケジューリング方式である．ここで，preemptive とは高い優先度のタスクが低い優先度のタスクを一時的に中断して実行できることを意味する．(UNIX は non-preemptive なスケジューリング方式．)

rate monotonic スケジューリングは固定優先度の周期タスクスケジューリングにおいて最適であることが証明されている．また，周期タスクの数を m，タスク $\tau_i(1 \leq i \leq m)$ の実行時間，周期をそれぞれ C_i，T_i とした時，プロセッサ利用率 $\sum_{i=1}^{m} \frac{C_i}{T_i}$ の上限が以下のように導出されている．

$$U = m(2^{1/m} - 1)$$

rate monotonic スケジューリングはスケジューラビリティの解析システムに実際に用いられており，比較的実用的なアルゴリズムである．しかし，タイマ割り込みなどの OS が提供するサービスにかかる時間を考慮していない，などといった指摘もあり，より現実に即したアルゴリズムが今後求められるであろう [Ara 93]．

2.3 マルチメディア通信

光ファイバディジタル伝送，LSI 等の様々な技術の発展により，通信においても従来の音声，低速データ以外にも画像などを含む様々なメディア情報を伝送することが可能となってきた．

従来，こららの非電話系サービスはパケット網，ファクシミリ網といったように電話網とは別に，それぞれ別々に作られてきた．

ディジタル技術の発展に伴い，電話系，非電話系を含めた様々なサービスにおいて扱われる音声，画像などの様々なメディアを 0 と 1 のディジタル情報で扱うことが可能となり，様々なサービスを一つの通信網で行なうことを目的とした研究開発が盛んに行なわれてきている．

このような通信網で扱われるメディアとしては従来の固定レート音声 (64kbps) 以外にビットレートの変動の大きい動画，大容量ファイル伝送なども考えられ，様々な特性を示す様々なメディアが同じ通信網を共用することになる．

また，音声，動画ではある程度の伝送誤りは許されるが，データでは許されないなどといったように，要求される品質もメディアごとに異なるものとなる．表 2.1に通信からみた各種メディアの品質上の特徴を示す．

各種メディア		実時間性	情報誤り	網の信頼性への要求(メディアの特徴)
音声		高い (数十ミリ秒)	許容 (10^2～10^3) *	高い(記録，再現性がない)
画像	会話型動画像	高い (数十ミリ秒)	許容 (10^2～10^3) *	高い(記録，再現性がない)
	放送型動画	会話型より低い	許容 (10^2～10^3)	実時間性を要求する場合は 高い信頼性が必要
	静止画	会話型は高い	非許容 (再送は可能)	高い(再送は可能)
データ		会話型は高い	非許容 (再送は可能)	高い(再送は可能)

* セル損失率 (帯域圧縮符号化方式によっては誤りは許容できない．)

表 2.1: 通信からみた各種メディアの品質上の特徴

マルチメディア通信においては，様々なビットレート特性示すメディアの性質を利用して一つの伝送路の利用効率を上げると同時に，アプリケーションからオペレーティングシステムを通して要求されるメディア品質 (QOS) を保証できるようにネットワーク資

源を各コネクションに割り当てることが今後重要になると考えられる．

以下，まず，様々な特性を示す複数のメディアを一つの伝送路に効率良く多重化するための多重化方式について述べる．次に，ネットワークに対して要求されるメディア品質 (QOS) を保証するための QOS 保証について述べる．最後に，2.1節で示した，本論文で想定するマルチメディアを伝送する場合について触れる．

2.3.1 多重化方式 [Tomi 92, Koi 93, Aki 91]

様々な通信速度，特性を示す複数のメディア情報が同じ伝送路を共用するような場合，これらのメディア情報をどのように多重化して伝送するかが重要となる．

様々な通信速度の情報を同じ伝送路上に多重化して伝送する多重化方式としては，時間位置多重を使った同期方式と，ラベル多重を使った非同期方式の 2 つがある (2.13)．

時間位置多重はインタフェース上に固定長のフレームを定義し（実際は 125 micro second フレームが用いられる），多重化すべきコネクションが使用するタイムスロットの，そのフレーム内で占める時間位置を固定して，その位置情報でコネクションを識別する方法である．この方法は，回線モードとして電話サービスを中心に利用されている．一方，ラベル多重は各パケットのヘッダ内の識別子（ラベル）によりチャネルを識別する方法である．この方法は，パケットモードとして，主にデータサービスの分野で使われている．

様々なメディア情報を多重化する場合には，回線交換方式は多元速度情報の扱いが複雑で，かつ，バースト性情報の伝達が非効率といった難点がある．一方，パケット交換方式は，発生する情報量に応じてパケットを割り当て，この有意なパケットのみをネットワークが転送することによってネットワークのリソースを有効に利用することができる．つまり，様々な特性を持つメディア情報に柔軟に対応できる．しかし，スイッチングのための処理などのために高速化が困難であるといった難点もあわせ持つ．

そこで，回線モードとパケットモードの技術的限界を克服する方式として ATM が研究されてきた．

ATM はラベル多重の一種であり，パケットモードにおいて，パケット長を 53byte の固定長にすることでスイッチングなどの処理を全てハードウェアで行なうことによりパケットモードの利点を保持しつつ，高速化を図った方式と位置付けることができる．

図 2.14に ATM の統計多重化効果を示す．STM では通信チャネル個別に帯域を割り付けるため，バースト性の大きなメディア情報を転送する場合には図 2.14に示すように大

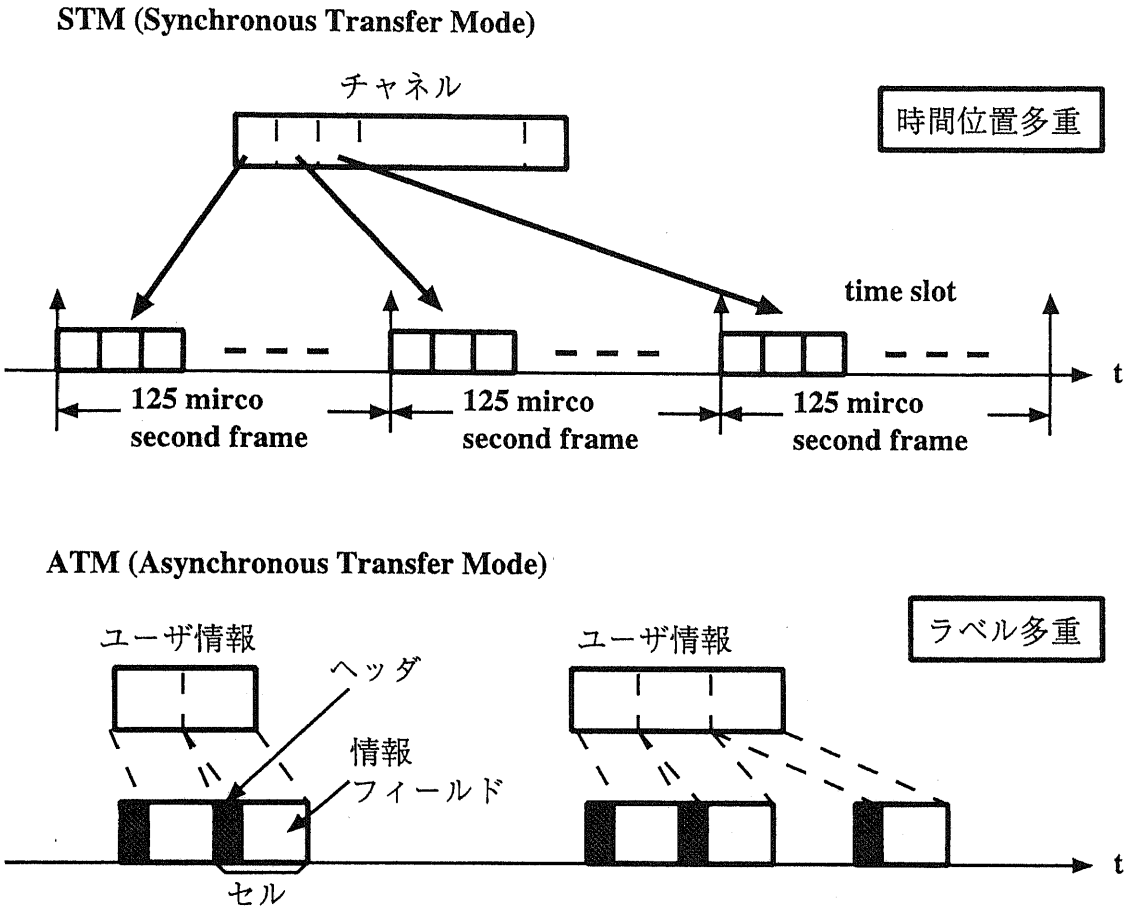


図 2.13: 各種多重化方式

きな伝送帯域を必要とする割に伝送路の利用率は低い．ATM では動きが大きい画像，有音部と無音部からなる音声などの様々な性質を持ったメディア情報を複数重ね合わせて伝送帯域を有効利用している．さらに，ピークが重なりあってしまった時のために若干のバッファ回路を挿入することで統計多重化効果を高め，伝送帯域を STM よりも小さくすることが可能となる．

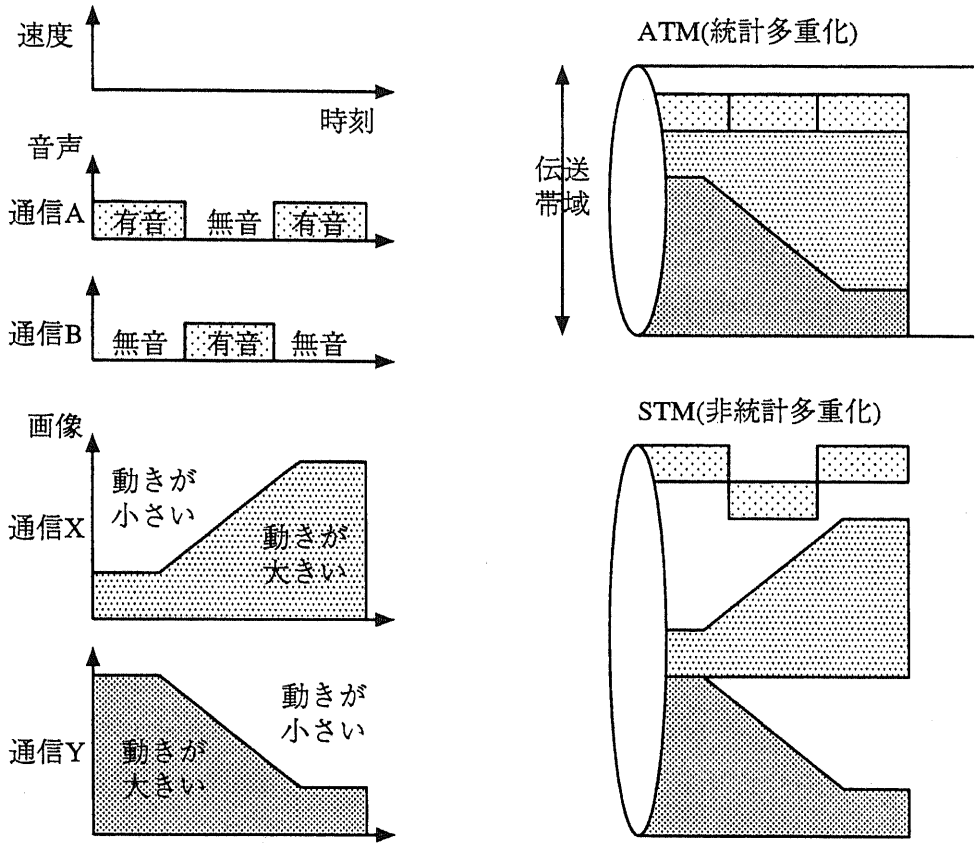


図 2.14: 統計多重化効果

2.3.2 QOS 保証 [Toku 93]

アプリケーションからオペレーティングシステムを通してネットワークに要求されるメディア品質 (QOS) としては以下のようなものが挙げられる.

- スループット
- 遅延時間
- 遅延分散 (ジッタ)
- パケット損失率
- ビット誤り率

これらの (ネットワークに対する)QOS を保証するためには，まずネットワーク自体が帯域 (bandwidth) の確保や，優先制御などの QOS を保証する上で前提となる機能を備えている必要がある．つまり，ネットワークとして Ethernet のように帯域の確保，優先制御などができないネットワークを用いている限り，その上にどのようなプロトコルがのっても QOS を保証することはできない．

また，いくら ATM，FDDI などのように帯域確保などが可能なネットワークが用いられても，その上にのるプロトコルが資源確保などの機能を備えてなければ QOS を保証することはできない．このような，QOS 保証が可能なプロトコルとしては ST-II(Stream Protocol II)，CBSRP(Capacity-Based Session Reservation Protocol) などがある．

ST-II はコネクションオリエンティドなプロトコルであり，コネクション設定時にデータの平均サイズ，バースト時のスループット，ラウンドトリップの最大許容時間，遅延時間の分散，エラー率などを指定することができる．これらの QOS に対する要求パラメータは次々と経路上のルータを伝わり，その過程で各ルータにおいてネットワークなどの資源が確保されていく．この際に，要求された QOS を保証するのに十分な資源を確保できなければコネクションは成立しない．

CBSRP も ST-II と同様，コネクション指向オリエンティドなプロトコルである．CBSRP は単一 LAN 上 (FDDI 上) に実装されており，ST-II のようなルータでの帯域予約は必要ない．CBSRP は新たなコネクションを張るのに十分な資源を確保できない場合，既に張られている (優先度の低い) コネクションの QOS を下げることで新たなコネクションに対する QOS を提供できるように動的に QOS を協調制御する機能を備えている．

QOS 保証は例えば QOS として一つのスループットの値（最大値など）を指定する場合，それに必要な帯域を確保してしまうため，ATM における統計多重化効果を生かすことはできないようにみえる．しかし，QOS として幅のある値（[最低限保証してほしい値～本当に欲しい値] など）を指定する場合には，それぞれのセルに優先度を付けることなどにより，うまく統計多重化効果が生かせるときには本当に欲しい QOS をネットワークを有効利用しながら提供でき，そうでない時にも最低限保証してほしい QOS を提供できる．つまり，必ずしも多重化と QOS 保証は互いに競合するわけではなく，セルの優先度付けなどにより，QOS を保証しつつネットワークの有効利用も図ることが可能であると考えられる．

2.3.3 狭義のマルチメディア通信

通信においては，一つのネットワークが複数の異なる種類のメディア情報を扱うことをマルチメディア通信と呼んでいる．ここで，これらのメディアは必ずしも互いに関連付けられている必要はない．2.3.1節，2.3.2節などではこのような広い意味でのマルチメディア通信を実現する上での手法を見てきた．

本節では，2.1節で述べたような，マルチメディアを互いに関連付けられた情報として捉えた場合の（狭い意味での）マルチメディア通信について触れる．

このような狭い意味でのマルチメディアを伝送する場合には，従来，MPEG のようにそれらのメディアを送信側で一旦多重化し，一つの stream としてネットワークに送り出すといった方法が取られてきた．このような方法は全てのパケットが同じように扱われる従来型の（モノメディア）ネットワークにおいてはメディア間の同期合わせが楽な点などで有利な方法である．

しかし，2.3.1節，2.3.2節で述べたような（広い意味での）マルチメディアネットワーク基盤が整備された環境ではそれぞれのメディア情報を別々の stream で送り出す方が各メディアの QOS に応じたメディア送出のスケジューリング，パケットの廃棄が可能になるなどといった点で一つの stream に多重化してから送り出すよりも有利になると考えられる．

以下に，各々のメディア情報を別々の stream を用いて送出する場合の利点，欠点を示す．

利点
- メディアの特性を生かした転送が行なえる．

  一つの stream に多重化してから送り出す場合，冗長な情報を含みある程度の伝送誤りは許されるが時間的制約に厳しい動画，伝送誤りの許されないデータなどの様々なタイプのメディアが一つの stream として伝送されることになる．このような場合，この stream の QOS としては構成要素である各メディアの内最も厳しいもの（時間的制約に厳しく，伝送誤りは許されない，等）を指定することとなる．つまり，各メディアの特性を生かすことができず，必ずしも必要でないリソースまで確保してしまうことになる．

  各メディアを別々の stream で送り出すような場合には，各メディア (stream) ごとに QOS を指定でき，メディアの特性を生かした通信が可能となる．

- 多重化が不要．

送信ノードでの multiplex，受信ノードでの demultiplex が不要となる．

- 中間ノードでのメディア変換が可能．

  例えば，ネットワーク中の中間ノードにおいて音声情報のみを日本語から英語に変換したり，音声情報をテキスト情報に変換して受信ノードに送り出すといった，高次のメディア変換機能を利用することが可能になる．

欠点

- 受信ノードでのメディア間の同期合わせが困難．

  受信ノードには各メディアが別々のコネクションを介して送られてくるため，メディア間のタイミングのずれは一つの stream を介して送る場合と異なり予測できない．つまり，メディア間同期を取るための手法が必要となる．

以上述べてきたように (狭い意味での) マルチメディア情報の転送には 2.3.1節， 2.3.2節で述べた（広い意味での）マルチメディアネットワーク技術，及び，メディア間の同期をとるための機構が必要となる．

2.4 同期

マルチメディアを（2.1節で定義したように）互いに関連付けられた複数のメディアの集まりと捉えた場合，この“関連付け”をどのように行なうかが重要となる．マルチメディアにおける同期とはまさにこの“関連付け”のことであり，マルチメディアがマルチメディアであるための本質をなす．

マルチメディアにおける同期は以下に示すように分類できる．

メディア内同期 単一メディア Stream 内でのメディア情報（フレーム）の実際の提示間隔と予定した提示間隔とのマッチング．

例えば，ネットワークを介して動画情報が転送されるような場合を考えると，ネットワークで生じる遅延のばらつきにより，送信側でのメディア情報の送出間隔が一定であっても，受信側でのメディア情報の提示間隔が一定になるとは限らない．このようなばらつきは通常受信側のバッファにより吸収される．バッファの大きさは，通常，ばらつきの最大値を見積もることで決定されるが，バッファを大きくするとメディア情報の送出時点から実際に提示されるまでの遅延が大きくなるため，遅延制約との trade-off も考慮する必要がある．

メディア間同期 複数のメディア間の情報提示タイミング，または，情報提示位置の関係．オブジェクト表現レベルでの同期と，Stream レベルでの同期に分類できる．

オブジェクト表現レベルでの同期 MHEG におけるコンテントオブジェクトのような一つのメディアからなる一塊のメディア情報同士を関係付ける際の同期．あるシナリオに従って様々なシーンやナレーションが提示されるような場合に，複数のシーンやナレーション間の提示時間の関係，及び，スクリーン上での位置関係などを規定する．

Stream レベルでの同期 (lip-sync) オブジェクト表現レベルでの同期よりもさら細かい粒度の同期として Stream レベルでの同期 (lip-sync) がある．Stream レベルでの同期はメディアフレーム提示間隔 (e.g. 33 ミリ秒) などのレベルで各メディアの同期を取るもので，代表的な例として唇の動きと話し声の間の間の同期を取る lip-sync がある．

後述の R&L 同期も Stream レベルでの同期である．

2.4.1 オブジェクト表現レベルでの同期

MHEG における同期

MHEG[Yasu 91] ではオブジェクト表現レベルでのメディア間同期を以下のように規定している.

基本形同期 同期表現の最初のレベルで, 2 つの基本オブジェクト (オブジェクト 1, オブジェクト 2) を複合オブジェクトの構成要素として定義する. 基本形同期の表現としては関連したモードの識別子のみが含まれる. すなわち, オブジェクトの 2 つの識別子, 2 つの整数パラメータ T_1 と T_2, 1 つの論理パラメータ S である. 基本形同期手法は次に示す形の複合オブジェクトに対して適用される.

$S = 0$ の場合 (シーケンシャルモード) 複合オブジェクトが起動された参照表示時間から T_1 時間後にオブジェクト 1 が表示され, オブジェクト 1 の表示の終了から T_2 時間後にオブジェクト 2 が表示される.

$S = 1$ の場合 (パラレルモード) 複合オブジェクトが起動された参照表示時間から T_1 時間後にオブジェクト 1 が表示され, この参照時間から T_2 時間後にオブジェクト 2 が表示される. この様子を図 2.15に示す.

条件つき同期 “オーディオシーケンス” や “オーディオビジュアルシーケンス” などのシーケンスとして表示する “時間軸を持つ” オブジェクトという基本オブジェクトの特別なサブクラスに適用される同期方式である. これらのオブジェクトタイプの表示過程で “開始マーク”, “終了マーク”, “通過マーク” という 3 つのタイプの “同期マーク” によりイベントが得られる. これらのマーク, あるいは, マークの組合せは “マーク状態” という関連オブジェクト表示の実時間で決められる値を持つ変数とともに定義される. 2 つのタイプ (“基本的条件”, “結合条件”) の “同期条件” によりイベントが生成される. 基本的同期の場合, 与えられたテスト値とマーク状態変数値との単純比較 (等しい, より低い, より高い) を行なう. 結合条件の場合, “and” および “or” の基本的条件の論理的組合せにより条件設定を行なう. 条件つき同期は, 上記に示したようにマーク状態変数にもとづいた基本的条件あるいは結合条件の条件によりオブジェクト表示が起動される基本オブジェクトに適用される. この様子を図 2.4に示す.

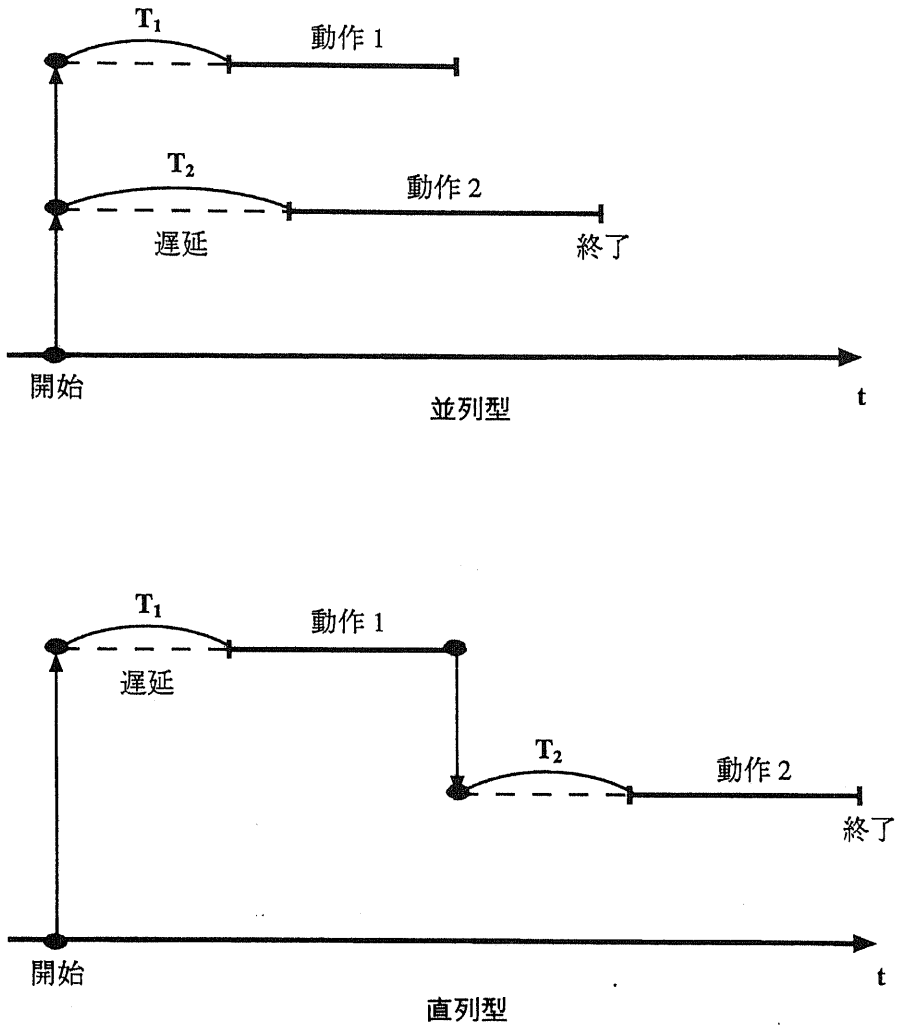


図 2.15: 基本形同期

巡回形同期および連鎖形同期 巡回形同期は与えられたマーク状態変数の変化点ごとに繰り返し表示を行なう 1 つの基本オブジェクト（入力または出力）に適用される．連鎖形同期は表示が連鎖的につながって行なわれ，全体として 1 つの複合オブジェクトを形作るような 1 つ以上の基本オブジェクトに適用される．

2.4.2 Stream レベルでの同期

MPEG

MPEG ではオーディオ，ビデオの同期アルゴリズム (AV 同期アルゴリズム) を規定している．ただし，MPEG における AV 同期アルゴリズムは送信側でオーディオ，ビデオの符号化入力をマルチプレックスしてから送りだし，受信側で再びデマルチプレックスする場合を想定している．

また，以下に示す同期方式ではセグメントと呼ばれる単位を用いている．ビデオ符号化ストリームの場合は 1 ～数フレーム分を，オーディオ符号化ストリームの場合は 1 ないし複数の AAU(オーディオアクセスユニット) を 1 セグメントとする．ただし，AAU は複数サンプル (ex. 1152 サンプル) 分をグループ化したオーディオ符号化単位である．

AV 同期アルゴリズムは 2 つあり，1 つは暗黙的同期 (implicit synchronization) といわれるアルゴリズムで，AV マルチプレックスの状態でほぼ同一時刻に対応するようにする．このアルゴリズムでは復号化遅延の差があっても何らかの手段で吸収されることを仮定している．このアルゴリズムでは高精度 (オーディオの 1 サンプルに対応する時間) の同期はできない．もう 1 つのアルゴリズムはタイムスタンプ (time-stamp) と呼ばれる同期のための補助情報を用いて高精度な同期を行なうものである．タイムスタンプは AV の符号化データに付与される時刻情報である．

暗黙同期 1 (簡易法) だいたい同一時刻，同一時間長に相当する各 AV のセグメントがマルチプレックスされているとすれば，復号化出力時に各セグメントの最初の部分が同一時刻に対応しているとみなしてもほぼ同期がとれる．図 2.16にこの同期法を示す．A セグメント 1 と V セグメント 1 の間に T_1 の時間差があったとし，それを復号化時に同一時刻に出力すると，T_1 の同期誤差が発生する．しかし，この T_1 の最大値を問題とならない範囲に規定すればよい．AV 同期誤差の主観評価実験によればオーディオが 20ms 先行ないし 50ms 遅延の範囲ではほとんど検知されない．

このアルゴリズムによる同期は，ランダムアクセス後等の最初の復号化開始時のみ行ない，それ以後のセグメントは PLL(Phase Locked Loop) 等による継続同期で行なわれる．したがって，もし図 2.16において今はセグメント 1 での同期を示しているがセグメント 2 から復号化を開始すれば，同期誤差は T_2 ということになる．

暗黙同期 2 (高精度法) 上記方法でのセグメントは，1 つないし複数の AAU(オーディオアクセスユニット) やビデオフレームで構成されるとしていたが，セグメント間の

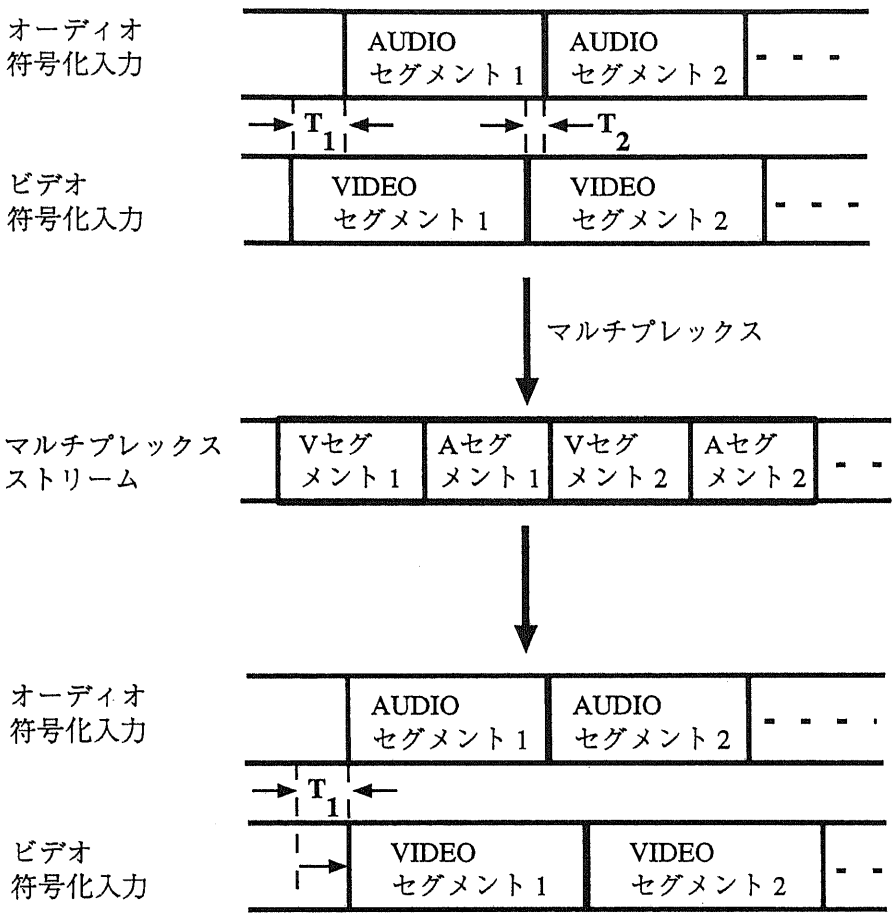


図 2.16: AV の暗黙同期

時間対応関係をより同一に近付けることができれば，同期誤差は小さくなる．本アルゴリズムではたとえば基本とする V セグメント境界で AAU を分割してマルチプレックスする．この様子を図 2.17に示す．ただしオーディオ符号化に可変長符号化要素が含まれている時には V セグメント境界時刻に合うように AAU の分割位置を正確に算出する必要がある．この計算精度によって同期誤差が左右される．

TS (Time Stamp: タイムスタンプ) による同期 AV の各マルチプレックスされるセグメントごとに，その先頭部分に対応する時刻情報 TS を付与し，それも含めてマルチプレックスする．その様子を図 2.18に示す． TS-A1， TS-V1 等は各先頭部に対応する TS である．この TS の精度は 90kHz のクロックを用いており，クロック周波数は 1 オーディオサンプル以上の時間精度を得ること， NTSC， PAL 両ビデオ

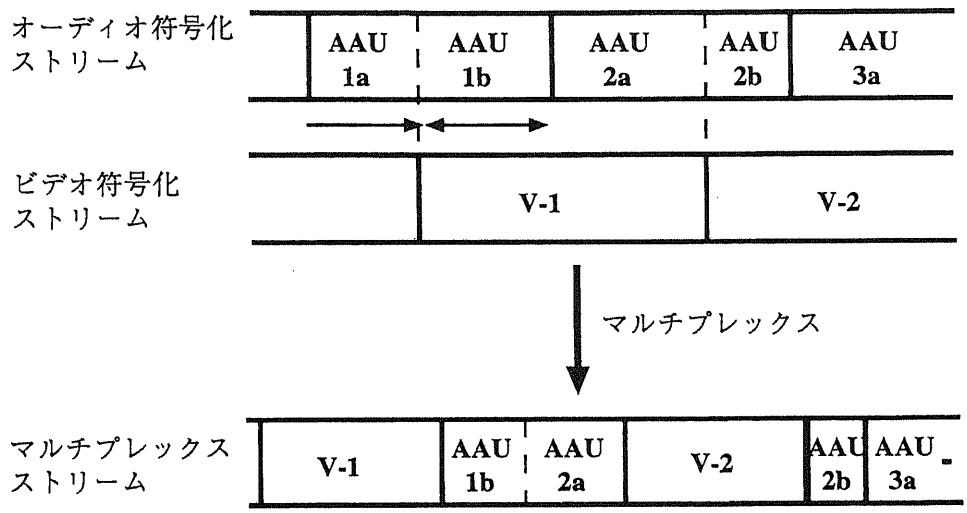


図 2.17: 細分割マルチプレックス

符号信号のフレーム周波数に同期できることの 2 つの条件を満たす周波数が採用された.

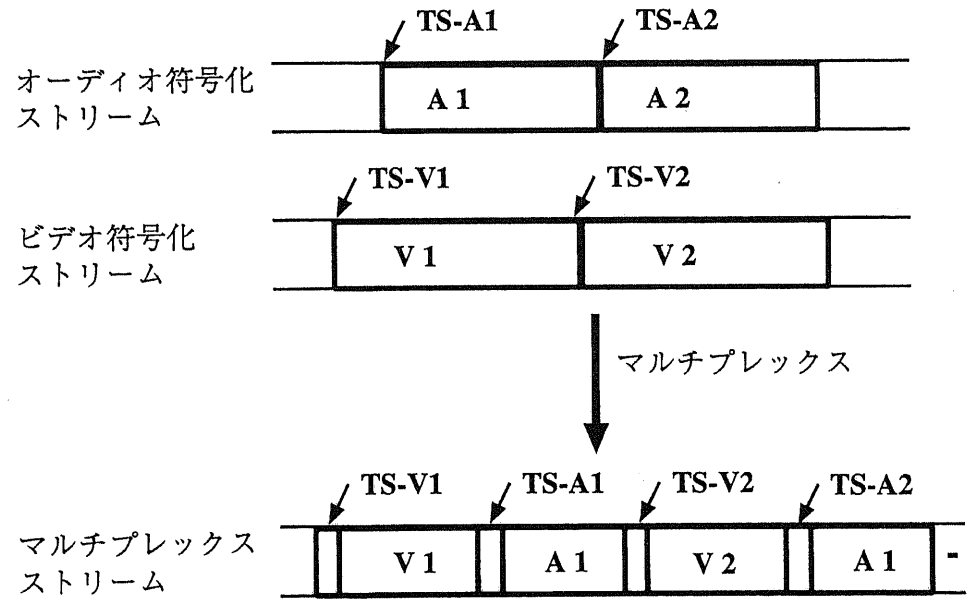


図 2.18: TS を含んだマルチプレックス

TS を用いて AV 同期を行なう原理を図 2.19に示す. AV の各 TS の差分を算出し,

その結果に応じてオーディオかビデオのどちらか一方の復号化出力を遅延されて同期する．実際のハードウェアでは，復号器出力の遅延はメモリ量が多くなるため，符号化ストリームを遅延させて (復号化開始時間を制御) 同期するのが一般的である．

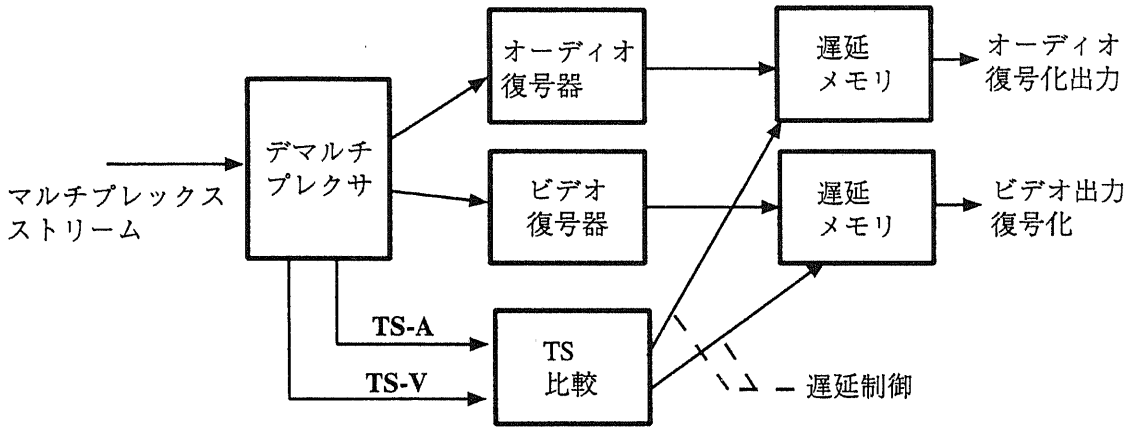


図 2.19: TS による同期法

Acme[Ander 91]

Acme(Abstraction for Continuous Media) は UC Berkeley(University of California at Berkeley) で開発された連続メディア入出力サーバである．

図 2.20に Acme サーバ内での連続メディア (CM,Continuous Media) の流れを示す．

Acme は複数の CM データストリームを同時に扱う．出力データは（論理デバイスごとに存在する）ネットワークコネクションから読み出され，必要に応じて型変換され，バッファリングされ，ミキシングされ，物理デバイス（ビデオディスプレイ又はスピーカ）に出力される．入力データは物理デバイス（ビデオカメラやスピーカ）から集められ，複製され，バッファリングされ，必要に応じて型変換され，コネクションに書き出される．

Acme では，CM データストリームを一続きの unit （video frame や audio sample) の集まりとしてモデル化している．各 unit はストリームの始まりからの時間位置を示すタイムスタンプを持つ．

Acme の同期は LTS(Logical Time System) と呼ばれ，同じタイムスタンプを持つデータ unit がほぼ同時に提示，又は，取り込まれることで LTS 内の複数の論理デバイスへ

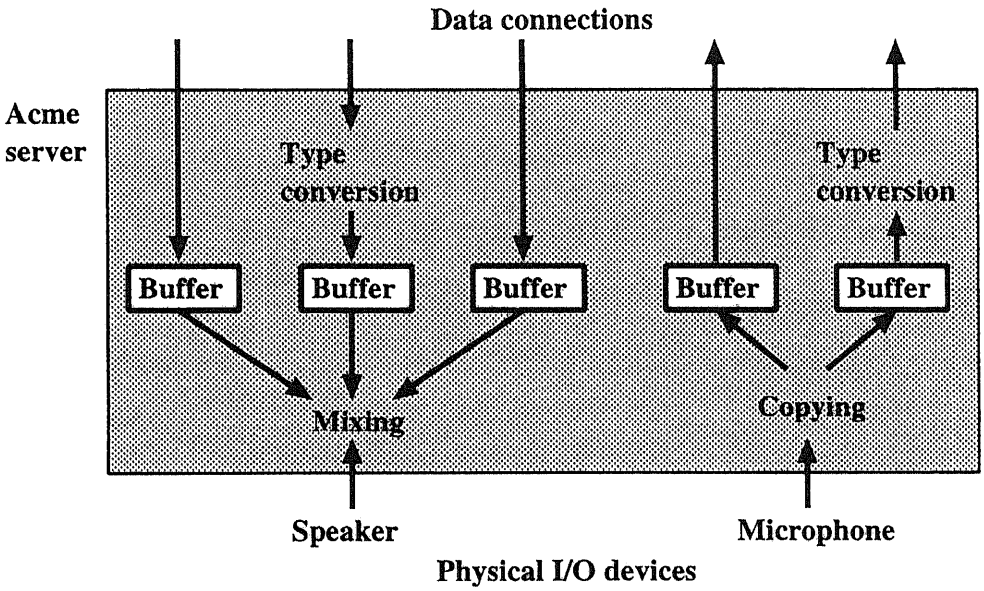


図 2.20: Acme server

の入出力の同期が取られることとなる.

各 unit のタイムスタンプは LTS が持つ current value（初期値 0）に従って付けられる. LTS の current value はほぼ実時間に比例して増加する値である. その時間粒度はデバイス割り込み時間（数十ミリ秒）である.

Acme は skip と pause により論理デバイスの速度を合わせる (図 2.21). skip は実時間よりもデバイスの速度を上げる. pause は実時間よりもデバイスの速度を落す. 出力デバイスの場合, デバイス割り込みにより起動されるルーチンは出力バッファからデータを捨てることにより skip する. これは, バッファ中に少なくとも 2 つのデータブロックがある場合のみ許される. pause を行なう場合, 割り込みルーチンはバッファからデータを取り出さず, 一区間の間, null データを提示する（つまり, 前回提示されたビデオフレームを提示する, もしくは, 無音データを提示する.）. 入力デバイスの場合, skip はバッファに null データを書き出すことで, pause は現在のデータブロックをバッファに書き出さずに捨て去ることで実現される.

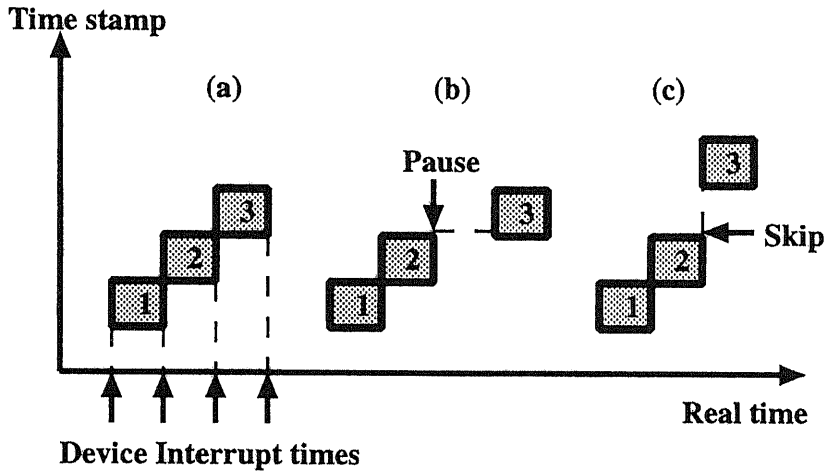


図 2.21: Stair-step diagrams for logical devices with pausing(b), skipping(c), and neither(a).

Harmony[Fuji 93]

Harmony は大阪大学で開発された分散ハイパメディアシステムであり，ハードディスクなどに格納された映像，音声，テキスト，アニメーションなどの複数のメディア情報からなるハイパメディア情報を Ethernet でつながれた 2 台のワークステーション上に提示するものである（映像，音声，テキストは SPARC station 10 上に，アニメーションは SPARC station IPX 上に提示される．）．

Harmony は複数のプロセスから構成されており，メディアの種類ごとにメディアオブジェクトの表示等を行なうクラスプロセスとこれらのプロセスにプレゼンテーションの指示を行なう Harmony サーバが互いにプロセス間通信を用いて通信しているという点で分散型のアプリケーションである．Harmony サーバは，利用者からの操作を受け付けるユーザインタフェース部 (Harmony/UI)，オブジェクト間のリンクを扱うリンク管理部 (Harmony/LM)，オブジェクトの管理を行なうデータベース部 (Harmony/DB) から構成されている．

Harmony では以下に示す 3 つの同期モデルを規定している．

開始点同期 オブジェクト表現レベルでの同期であり，映像と音声を同時に開始したい場合など，複数のメディア情報が同時刻に表示される場合の同期を開始点同期と呼ぶ．マルチメディア情報はデータ構造や容量，蓄積形式の違いによって，表示の指示が

なされてから実際に表示されるまでにかかる時間が異なってくる．例えば，光ディスク装置上の映像は，再生が始まる前に頭だしといった操作が必要である．このように，実際に情報が表示されるまでに行なわれる準備にかかる時間を考慮しなければ，表示にずれが生じることになる．更に，Harmony のような分散システムでは，遠隔のワークステーション上のプロセスに情報を表示させる場合，メッセージの伝送遅延をも考慮に入れなければならない．

Harmony では予めプレゼンテーションのためのシナリオを解析しておき，個々のオブジェクトの表示開始時刻を計算しておくことで開始点同期を実現している．また，利用者からの操作などにより（サブ）シナリオの開始時刻が決定（変更）される場合には，その開始時刻を改めて計算し直し，各クラスプロセスに知らせることで対応している．

実時間再生 実時間再生とは，映像，音声，アニメーションといった時系列情報を表示する場合に，例えば，内容が３０秒間であれば，表示が開始されてから正確に３０秒後に終了することを意味する．ディジタルサウンドプロセッサや光ディスク装置などのハードウェアを用いて音声や映像を扱う場合には，再生時の実時間性が保証される．しかし，ハードディスク上に保存されている映像を扱う場合やソフトウェアによってアニメーションを実現する場合などにおいては，ソフトウェアによって実時間性を保証する機構が必要となる．更に，ワークステーションの負荷が変動するため，それに追随する機能も必要になる．例えば，ワークステーションの負荷が増大するとプロセスの実行速度が遅くなるため，表示を間引くなどして対応しなければならない．

Harmony での実時間性の保証への対応を以下に示す．オブジェクトの表示再生において実時間性を保証するために，時系列メディアを扱うクラスプロセスでは，表示再生中に定期的にオブジェクトの進み具合を確認するペースマネージャと呼ばれる再生状況を監視する機構をクラスプロセス内部に組み込む．ペースマネージャは，あらかじめ設定された処理速度 (pace) に基づき，図 2.22のように定期的に表示割合いを確認する．ワークステーションの負荷の増大により表示が遅れている場合には，ペースマネージャが確認を行なった時点で表示を正しいところまでとばすことで対応する．これは QuickTime と同様の方法である．

適応連続同期 アニメーションや映像を教育的な目的に使う場合などでは，実時間性を優

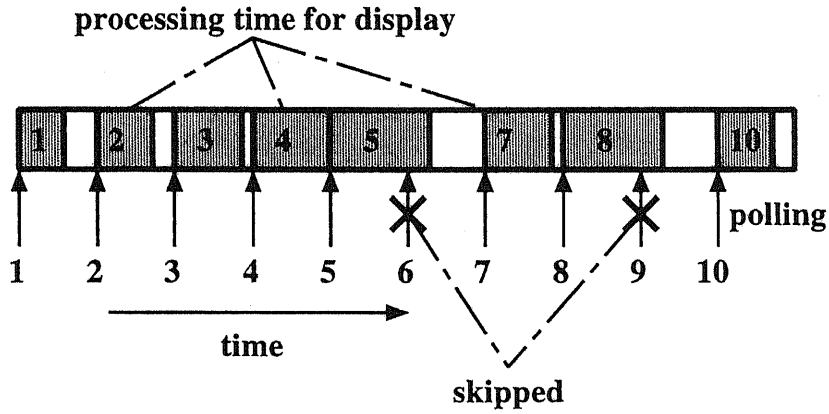


図 2.22: Real-time processing

先して表示を間引くことよりも内容を欠落させずに表示することが望まれることがある．この場合，それに同期する他のメディアの情報も合わせて進行を遅らせる必要がある．このような複数のマルチメディア情報の同期を実現するためには，ワークステーションの負荷の変動に対応して，情報の再生間隔を変化させると共に他の情報も同期して再生間隔を変更する機構が必要となる．

Harmony での適応型連続同期への対応を以下に示す．実時間再生の場合と同様に適応連続同期の場合も，各クラスプロセスのペースマネージャにおいて表示の進み具合を監視する．適応連続同期を実現するために，イベントの種類として，オブジェクトの表示中の再生の遅れを知らせる (pace Changed) を加える．また，適応連続同期のグループオブジェクト内にあるオブジェクトの再生が遅れた場合にグループ内の他のオブジェクトに再生速度を遅らせるように指示するメッセージ (set Pace) を用意する．

適応連続同期は以下のような手順で行なわれる (図 2.23)．

[step1] 個々のクラスプロセスは，オブジェクトの表示が開始されると通常の再生速度で再生する．

[step2] マスターオブジェクトにおいて実際の表示がその pace に追いつかなくなった場合，マスターオブジェクトのクラスプロセスはプレゼンテーションマネージャに pace Changed のイベントを送る．この時，マスターオブジェクトが

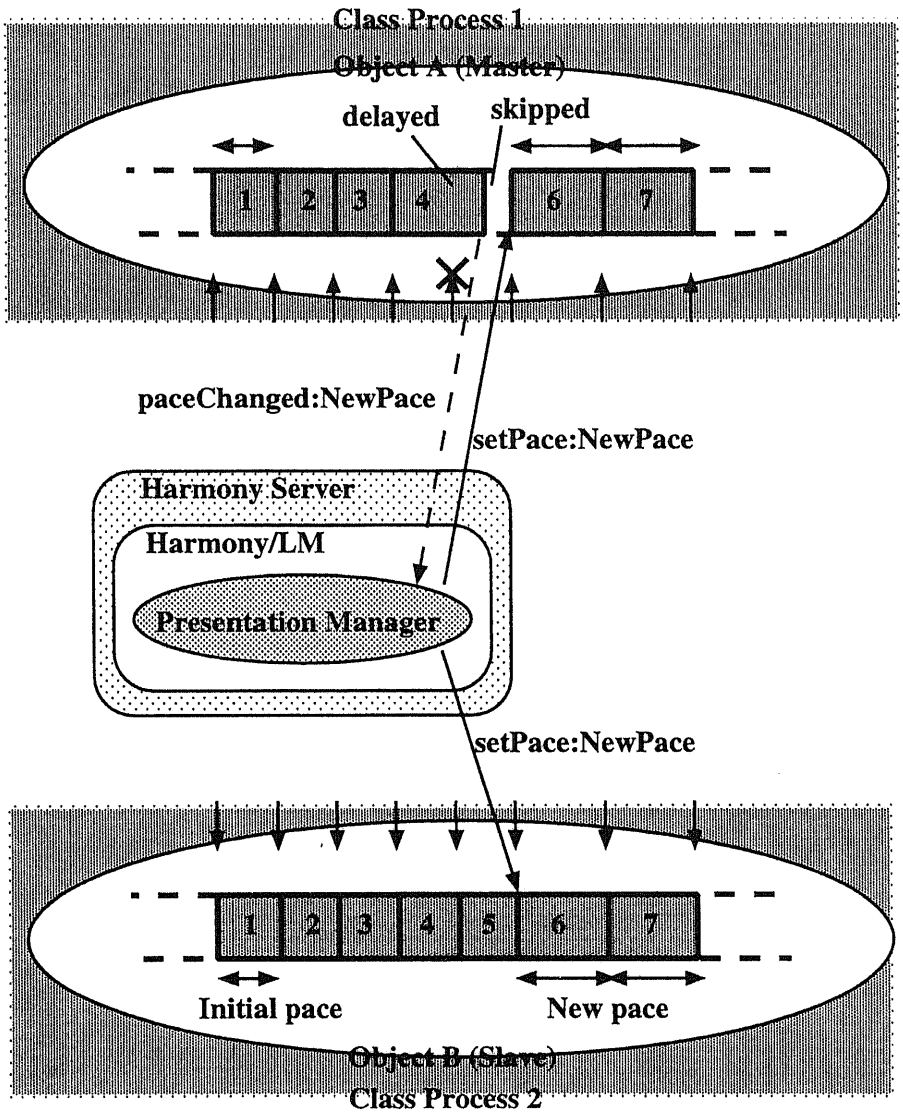


図 2.23: Adaptive continuous synchronization

要求する pace を pace Changed イベントの引数として渡す．この時点では表示を飛ばして pace を守る．

[step3] プレゼンテーションマネージャは同期しているすべてのオブジェクトに pace Changed イベントから受けとった pace の値を引数として set Pace メッセージを送る．同時に，これに続くオブジェクトの開始時刻が無効になるため，サブシナリオの開始時刻の変更を各クラスプロセスに通知する．

[step4] メッセージを受けとったクラスプロセスは引数の値に従って再生割合を変更する.

[step5] マスターオブジェクトでの表示に余裕ができた場合には, step 2 と同様に pace Changed イベントを送り, pace を元の値に近付けるようにする.

第 3 章

遠隔会議システムにおける R&L 同期

本章では遠隔会議システムにおける同期について，その問題点を示し，本研究の目的を明確化する．また，本研究で対象とするシステムの構成を示し，R&L 同期を定義する．

3.1　遠隔会議システムにおける諸技術

遠隔会議システムにおいては，その構成要素となる端末，ネットワークが参加者の映像などのマルチメディア情報を扱うのに十分高速，広帯域であると同時に，発言権の管理などといった，遠隔会議システム独自の機能も必要となる．

以下に，特に遠隔会議システムに必要とされる諸技術を示す．

- 会議開始前に必要となる諸技術

  会議召集 会議の召集，参加者への通知，出欠の確認など．電子メイルなどを用いて行なわれる．

  コネクション設定 会議を始めるにあたって，参加者間にコネクションを張る必要がある．

- 会議中に必要となる諸技術

  グループ通信 参加者間で動画，音声などを含むマルチメディア情報をやりとりするためには効率的なマルチキャスト機能などを含むグループ通信機能が欠かせない．また，様々なメディア間の同期を取ることで，各ノードにおいて意味のある情報を提示することも必要となる．

  排他制御 システム中に一つしかない（または，数に上限がある）資源，権利などに対しては排他制御が必要となる．排他制御にはビデオ装置などの蓄積型情報源に対する操作権の管理，発言権の管理，議長権の管理，電子黒板などの共有情報に対する変更の排他制御などがある．

  入退出管理 会議への入退出管理も遠隔会議システムに必要な機能の一つである．入退出の際には，コネクションの設定や解放，他参加者への通知などが必要となる．

本論文ではこれらの諸技術の中で特にグループ通信を行なう上で欠かせない技術であるメディア間同期について述べる．

3.2　研究の目的

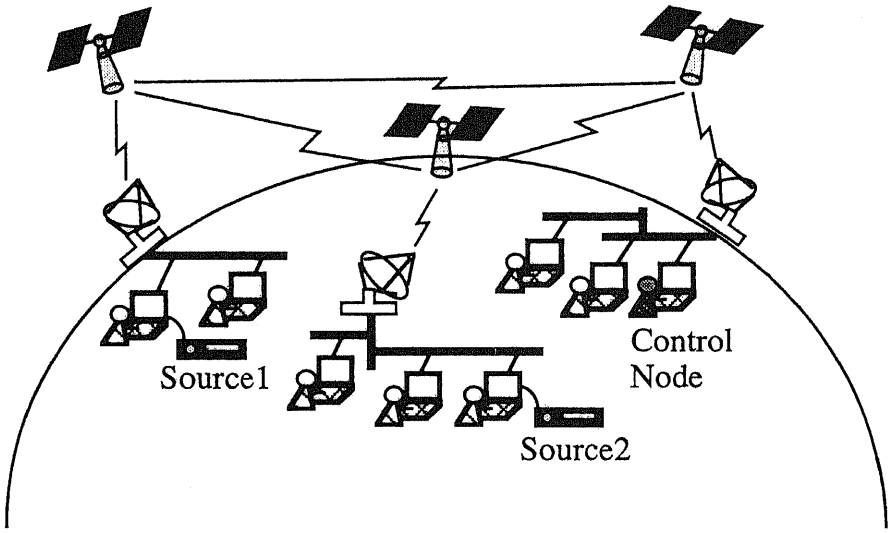


図 3.1: マルチメディア遠隔提示システム

マルチメディア遠隔提示システムで用いられるネットワークとしては B-ISDN，衛星回線など様々なものを考えることができるが，どのようなネットワークを用いるにしろネットワークの伝送遅延を避けることはできない．このようなネットワークの伝送遅延により以下のような問題点が生じる．

- ネットワーク中に分散配置されているディスク，テープ，ビデオカメラなどの様々な情報源から送られてくる複数のメディア情報を各ノードで提示する際にメディア間のタイミングのずれが大きくなる．
- ディスク，テープなどの蓄積型情報源に対して一時停止などといった操作命令を出した際に，指示してから実際にその命令が情報提示に反映されるまでの遅延が大きくなる．

情報提示を行なうノードにおいてメディア情報をメモリ中にキャッシュすることは上記の 2 つの問題点のいずれにも有効な手法となる．

従来のメディア間同期に関する研究 [Rama 93, Lit 91, Stein 90, Ander 91] はビデオ・オン・デマンドのように一つの蓄積型装置から取り出されたマルチメディア情報を一つのノードで提示する際にバッファを用いてメディア間の到着時点のずれを吸収する手法に関するものが主で，多地点間の会議や教育を支援するシステムなどのように共通の情報が複数のノードにおいてほぼ同時に提示されるようなシステム [Ahuja 92, Clark 92, Ahuja 88] においては，メディア間同期はあまり問題にされてこなかった．

本論文では，このような多地点間の会議や教育を支援するシステムにおけるメディア間同期の問題を取り上げる．

例えば，遠隔教育システムを考えた場合，そこで使われるメディアとしては以下のようなものが考えられる．

教材
- 教育用に作られたビデオ教材．例えば，NHK の教育番組のようなもの．
- 教師による手作りの教育用教材．例えば，生徒のテニスのフォームに教師がコメント付けしたものなど．

  また，遠隔教育ということを考えると，各生徒が自分のテニスのフォームを各自のハードディスクに入れておき，それに対し教師，もしくはプロのテニスプレーヤーなどがコメント付けを行なうなどといったことも考えられる．この場合，後から付けられるコメントはディスク容量，セキュリティーなどの関係で生徒のテニスフォームとは別のハードディスクに入れられるかもしれない．

このような教材は，例えば，教師により一時停止，再生などの操作を受けながら各ノードで提示される．

コメント　教師による説明，生徒の説明，教師・生徒間の議論など．

ここで，教材が格納されているノードとは別のノードにいる人が発言した場合などを考えると，何も制御しなければ発言者以外のノードでは教材中のメディアよりも発言者の声が遅れて聞こえてしまう．

そこで、本研究ではディスク，テープなどの蓄積型情報源から引き出されたメディア情報（RMS）とその説明を行なう参加者の声などの生のメディア情報（LMS）を分けて考える．各ノードでは RMS がバッファリングされ，その RMS 上の情報提示位置を適切に定めることで各ノード間の情報提示位置の関係が決まる．さらにこのノード間の情報提示位置の関係により LMS と RMS が各ノードで提示される際のタイミングが決まる．

つまり，各ノードにバッファリングされるメディア情報を用いることで多地点環境での RMS と LMS との間のメディア間同期を取ることが可能となる．RMS が予め存在する情報であるために予めバッファリングしておくことが可能であるのに対し，LMS は生の情報であるためにできるだけ早く提示されることが望ましいといった RMS と LMS の性質の違いがあることからこの方法が適用しやすいことになる．

本研究の目的は，様々な環境においてこのような RMS と LMS の間の同期手法を確立することである．

3.3　システムの概要

3.3.1　システムの概念

システムの概念を図 3.2に示す.

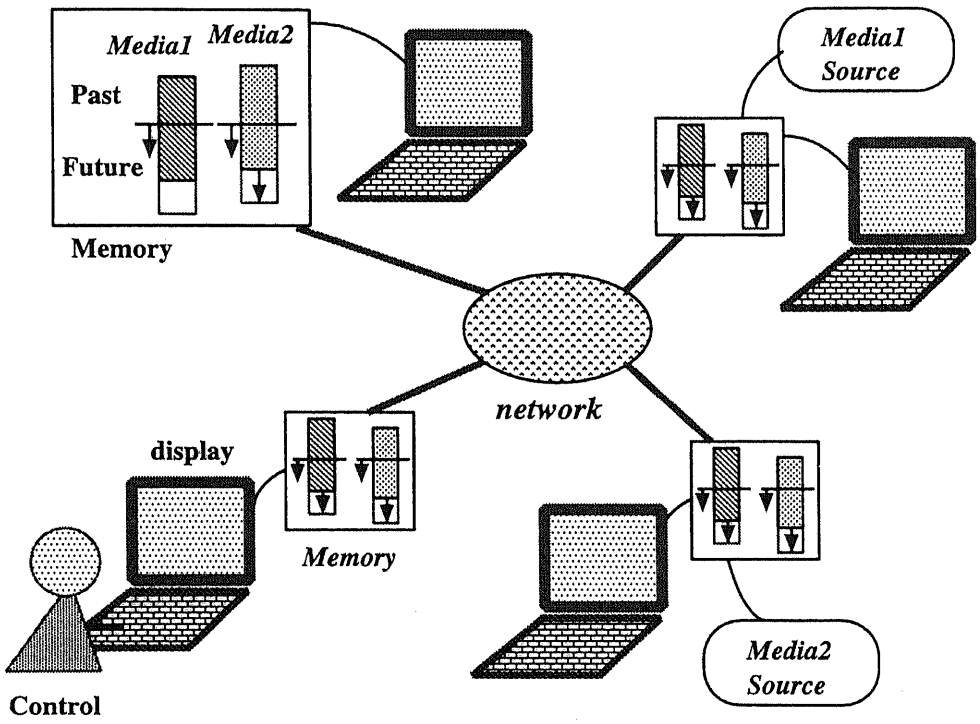


図 3.2: システムの概念

各ノードにはディスプレイ装置などのマルチメディア表示用の装置，及び，メディア間のずれの吸収などに用いられるメモリが備わっている．また，システム中の幾つかのノードにはメディア情報の入ったディスク，テープなどの情報発生源となる装置が備わっている.

本システムの基本的な動作を以下に示す.

- システム中の一つのノードの利用者が操作者となる．操作者ノードは再生，巻き戻しなどの指示を出す．この操作命令はシステム中の各ノードにマルチキャストされる.
- 情報発生源ノードでは操作者ノードからの指示に従いメディア送出位置を変化させる.

- 各情報提示ノードでは情報発生源ノードから送られてきたメディア情報を一旦バッファ中に蓄える．各ノードでは操作者ノードからの指示に従い情報提示位置を変えつつも，メディア間同期を保ちながらメディア情報の提示を行なう．

上記の動作により，本システムでは各ノードにおいて 3.4.2節で述べる R&L 同期のタイミングを満たしながら同一のメディア情報が提示される．

ここで，操作コマンド，各情報源からのメディア情報などは各ノード間ごとに別々にコネクションを張り別々のタイミングで送り出すことも考えられるが，マルチキャストを用いた方が全体としてのネットワーク使用率が低く抑えられ，また，情報送出ノードでの処理が簡単であるといった理由でマルチキャストにより (同時に) 送り出すこととする．

3.3.2　メディア操作の概要

主なメディア操作の概要を以下に示す．但し，操作を行なえる人は一時に一人に限るとし，操作権がノード間を移動するといった方法で操作権の委譲が行なわれるとする．

再生： 再生命令と再生開始位置からなる命令パケットを各ノードに送る．各ノードでは，通知された再生開始位置から再生を開始する．

一時停止： 停止命令と停止位置からなる命令パケットを各ノードに送る．各ノードでは，通知された停止位置の画面を表示し，停止する．

巻き戻し： 巻き戻し命令を各ノードに送る．巻き戻し中は操作者ノード以外で画面が表示される必要はない．

3.3.3　想定する環境

ネットワーク ネットワークとしては B-ISDN のように予め資源を確保することで伝送遅延時間などがある範囲内に収まることを保証できるものから Ethernet(10Mbps) のようにネットワークの負荷変動により伝送遅延時間の変動が予測できないものまで様々なものが考えられる．

端末 端末としてはこのシステムのための専用端末などのように本システムの正常な動作を保証するだけの資源があらかじめ確保できるものや，ワークステーションのよう

に他のプロセスの挙動により利用可能な資源が変動するものなど様々なものが考えられる．

情報源 動画，音声などのような等間隔で情報が発生するメディアの入ったハードディスク，ビデオテープ装置などを考える．

3.3.4　同期に必要な付加情報

本システムでは，すべてのメディア情報源でのメディアフレーム発生間隔を 33 ミリ秒といった共通の値とする．各フレームにはフレーム番号が付けられ，同じフレーム番号を持つメディアフレームを同時に提示することでメディア間の同期を取る．

以下に同期情報を付加すべきメディア情報を示す (図 3.3)．

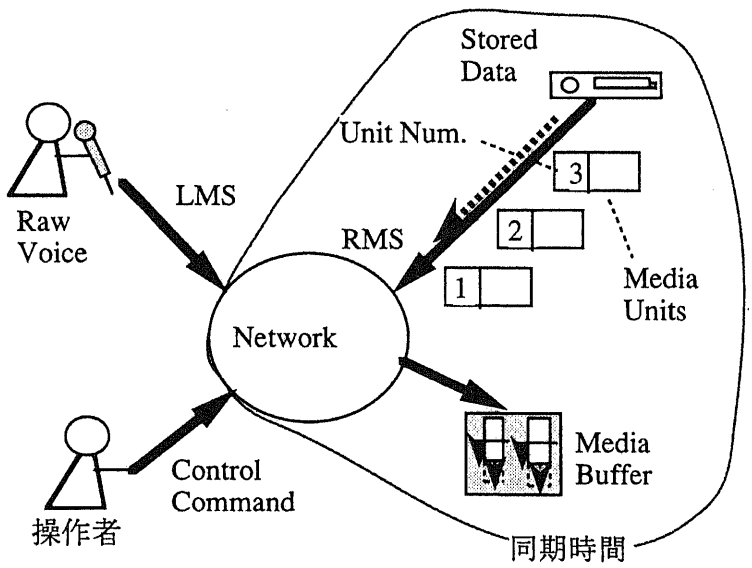


図 3.3: 付加すべきメディア情報

情報源中に蓄積されている情報 等しい情報量のフレームが順番に並んでいるような場合には，１フレーム目の先頭の位置から他のフレームの位置もわかるので同期情報は必要ない．しかし，各フレームの情報量が異なる場合，フレーム番号と各フレームの先頭位置の対応関係を示す情報が必要となる．

RMS RMS 中の各フレームにはフレーム番号が付く．フレーム番号は 1 つの RMS 中における各フレームと 1 対 1 に対応している．例えば，先頭のフレームから 1,2,3,... といった順番で付けていく．

LMS LMS 中の各フレームが送信された時に提示されていた RMS 中のメディアフレームのフレーム番号をその LMS 中のフレームに付加する．

各情報提示ノードでは LMS 中のフレームもバッファリングし，RMS も含めて等しいフレーム番号のフレームを同時に提示することでメディア間同期を取る．ただし，LMS 中のフレームを受信した時点で RMS 中の情報提示位置が受信フレームのフレーム番号を過ぎていた場合，同期情報に関わらず受信フレームは即座に提示される．

すなわち 3.4.2節の同期のセマンティクスで sync_with_one, sync_with_multi でない場合には同期情報は必要ない．

操作命令パケット 操作命令を示す情報には停止位置，再生開始位置などを示すフレーム番号と命令が発行された時間を示す情報が付加される．

バッファリングされている情報 バッファリングされている各フレームにもフレーム番号を付加する．

3.4　R&L 同期

3.4.1　RMS と LMS

本節では，RMS と LMS の違いについて考える．

直観的に言うと，予め存在しているメディア情報が RMS であり，その場で発生する情報が LMS である．しかし，会議の進行を記録しておいて後で見るような場合は LMS がある時点を境にして RMS に変わったことになる．

以下，2 つの属性について RMS と LMS の違いを述べる．

1. システムに入っていく瞬間の属性．
   つまり，ハードディスク，マイクなどからコンピュータ内に入っていく瞬間の属性．

   RMS 既に同期情報が付加されている．

   LMS 入ってきた瞬間に RMS に従った同期情報が付加される．

2. 同期時間軸との関係．
   ここで，同期時間軸とは同期情報（フレーム番号）の並びであり，逆再生などの命令に従ってその進行が変化する時間軸とする（図 3.4）．

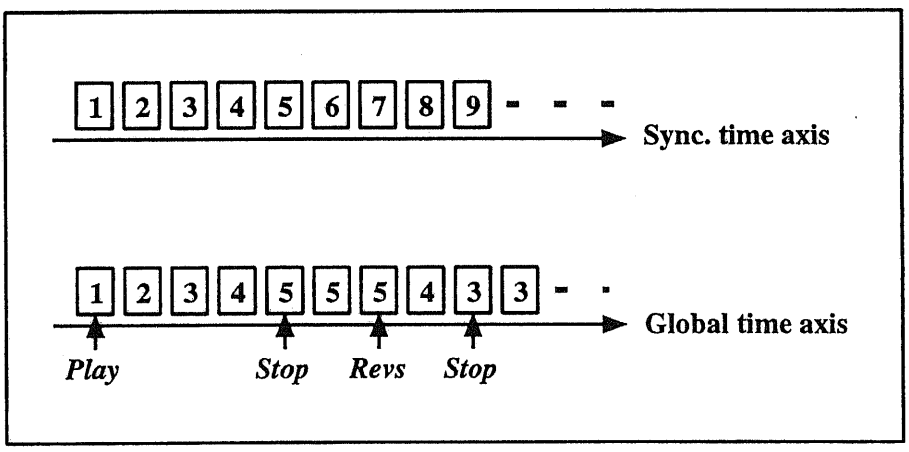


図 3.4: 同期時間軸と絶対時間軸

RMS 同期時間軸上に張りついている．

LMS 一度提示されたら消滅する（LMS の一過性）．

authoring システムなどでは LMS がシステムに入った瞬間に同期時間軸上で固定されることになるので，システムに入った瞬間に LMS が RMS に変わることになる．

また，会議の進行を記録しておき，後で見るような場合は会議の進行中の絶対時間軸が，後で見る際の同期時間軸になる．

以上をまとめると，同期時間軸に張りついているメディア情報が RMS で，同期情報を参照しながらも一過性を保ちつつ提示されるメディアが LMS であると考えることができる．

3.4.2　R&L 同期のセマンティクス

3.4.1節で分類した 2 種類のメディア間の同期関係を以下のように定義する．

R&R 同期 ビデオ・オン・デマンドにおいて各メディアを別々のコネクションを用いて送る場合のような，複数の RMS 間の同期関係．

R&L 同期 遠隔会議，教育システムなどにおけるビデオ教材と，それを説明する人の声との間の情報提示のタイミングのような，RMS と LMS との間の同期関係．

さらに，LMS と RMS の情報提示の際のタイミングに基づき R&L 同期のセマンティクスを以下のように定義する．ここで，RMS 間の同期はすべて厳密に取るものとする．

sync_with_one RMS と操作者からの LMS との同期．すべてのノードにおいて，操作者ノードにおいて LMS が取り込まれたのと同じタイミングで，LMS が RMS と同期して提示されることとする．これは，操作者以外のノードにおける再生点を操作者ノードでの再生点よりも，LMS の圧縮，伝送，及び，伸長にかかる時間の分だけ遅らせることで実現できる (図 3.5)．

このセマンティクスは，例えば，遠隔教育を支援するシステムにおいて教師が教育用教材を操作しながら授業を進めていくような場合に適用される．

sync_with_all RMS と全ての参加者からの LMS との同期．すべてのノードでの再生点が各時刻において等しいこととする (図 3.6)．

このセマンティクスは，例えば，遠隔会議を支援するシステムにおいて全ての参加者が会議資料を見ながら議論に参加するような場合に適用される．

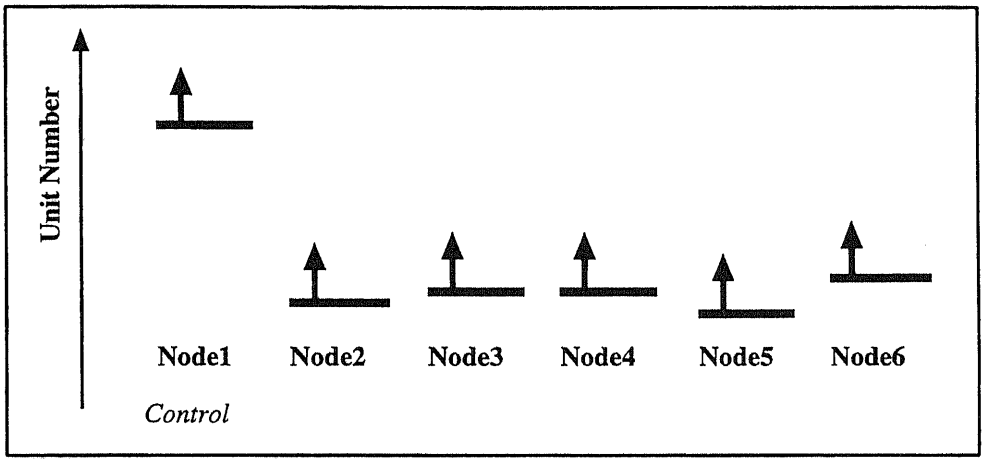


図 3.5: IPPs in the case of sync_with_one

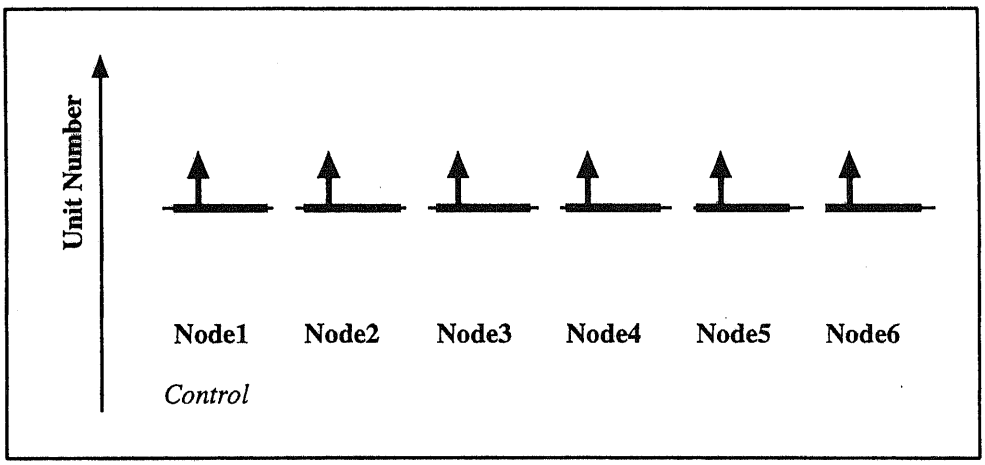


図 3.6: IPPs in the case of sync_with_all

sync_with_multi RMS と操作者を含む複数の参加者 (発信者ノード) からの LMS との同期.

すべての発信者ノードでの各時刻における再生点が等しく，それ以外のノードでは，発信者ノードにおいて LMS が取り込まれたのと同じタイミングで，LMS が RMS と同期して提示されることとする (図 3.7).

(sync_with_one，sync_with_all は sync_with_multi の特別な場合と考えることもできる．)

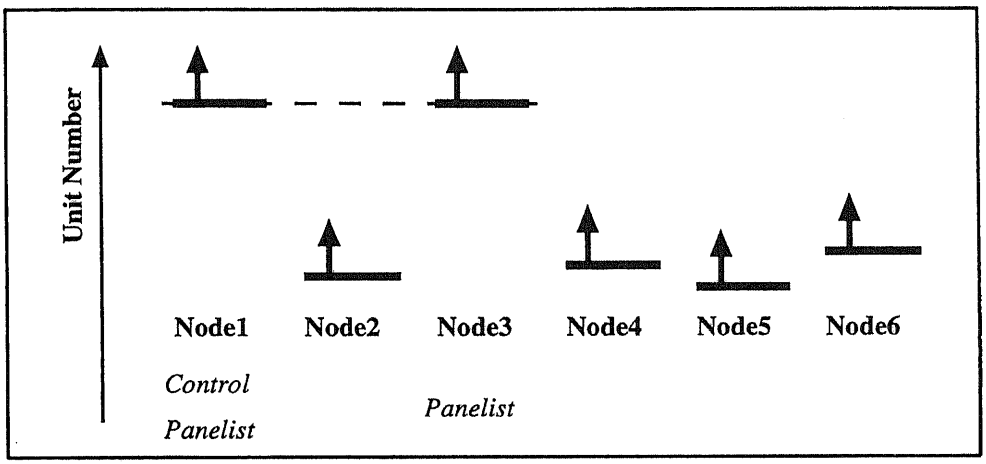


図 3.7: IPPs in the case of sync_with_multi

このセマンティクスは，例えば，遠隔討論会を支援するシステムにおいて討論参加者が討論会資料を見ながら議論を進めていくような場合に適用される．

sync_with_none RMS と LMS との同期は考えず，RMS 間の同期のみ保証する．

3.4.3　同期の粒度

3.3.4節で述べたように，本論文ではすべてのメディア情報源でのメディアフレームの発生間隔を共通の値としている．

2.4.2節で述べたように，数十ミリ秒の同期誤差はほとんど検知されないとされており，メディアフレームの発生間隔を数十ミリ秒に取れば，検知される範囲内の同期はほぼ取れることになる．

また，音声のようにサンプリングレートが高いメディアの場合情報源からのメディアフレームの発生間隔を動画フレームの発生間隔よりもかなり小さくとることができる．このような場合にも一時停止，再生命令などでの停止位置，再生開始位置としては動画などの（発生間隔の大きな）メディアのフレーム位置を用いることとする．つまり，図3.8の P と記してあるところが一時停止位置などとして用いられることなる．

図3.8においては，例えば Media2 の 4，5，6 フレーム目，Media3 の 3，4 フレーム目を Media1 の 2 フレーム目と同時に提示することがメディア間の同期を取ることになる．

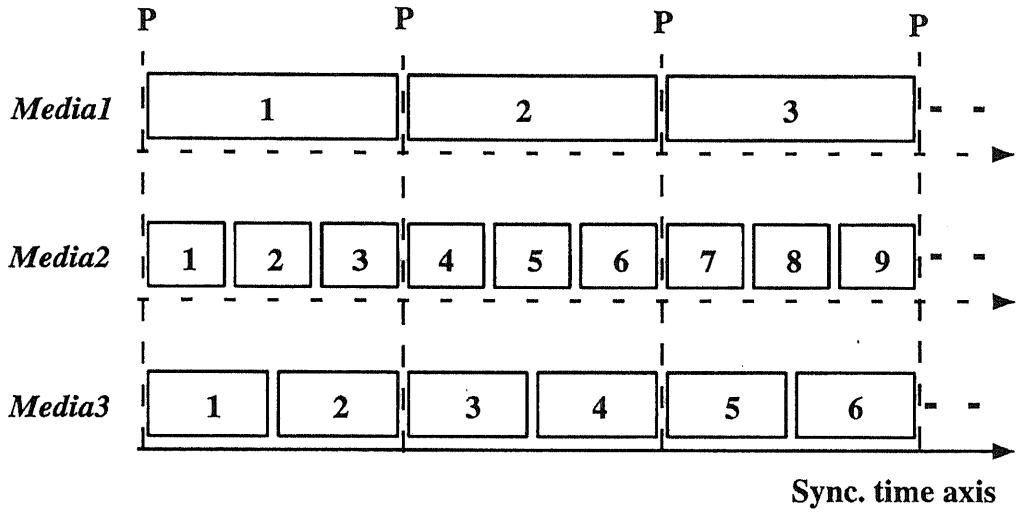


図 3.8: メディアごとに発生間隔が異なる場合

以下，第 4章，第 5章においては基本的にすべてのメディア情報源でのメディアフレームの発生間隔が等しい場合を想定して議論を進めていく．

メディアフレームの発生間隔が異なる場合についてはそれぞれの章の 4.3.3節，及び，5.5.3節で触れる．

3.4.4　本研究の位置付け

本節では，MHEG 階層での本研究の位置付けについて述べる．

R&L 同期は Stream レベルでの同期であり，MHEG のようなオブジェクト表現レベルでの同期よりもより粒度の細かい同期である．また，この同期は遠隔会議システムのような多地点での活動を支援するようなシステムにおいてのみ必要であり，VOD のような個人活動を支援するシステムには必要ない．つまり，本研究は MHEG 階層上で図 3.9に示すような位置に位置付けられる．

また，RMS として MHEG オブジェクトを用いる場合，各 MHEG オブジェクトを予め同期時間軸に張りつけておくことで R&L 同期を実現できる．

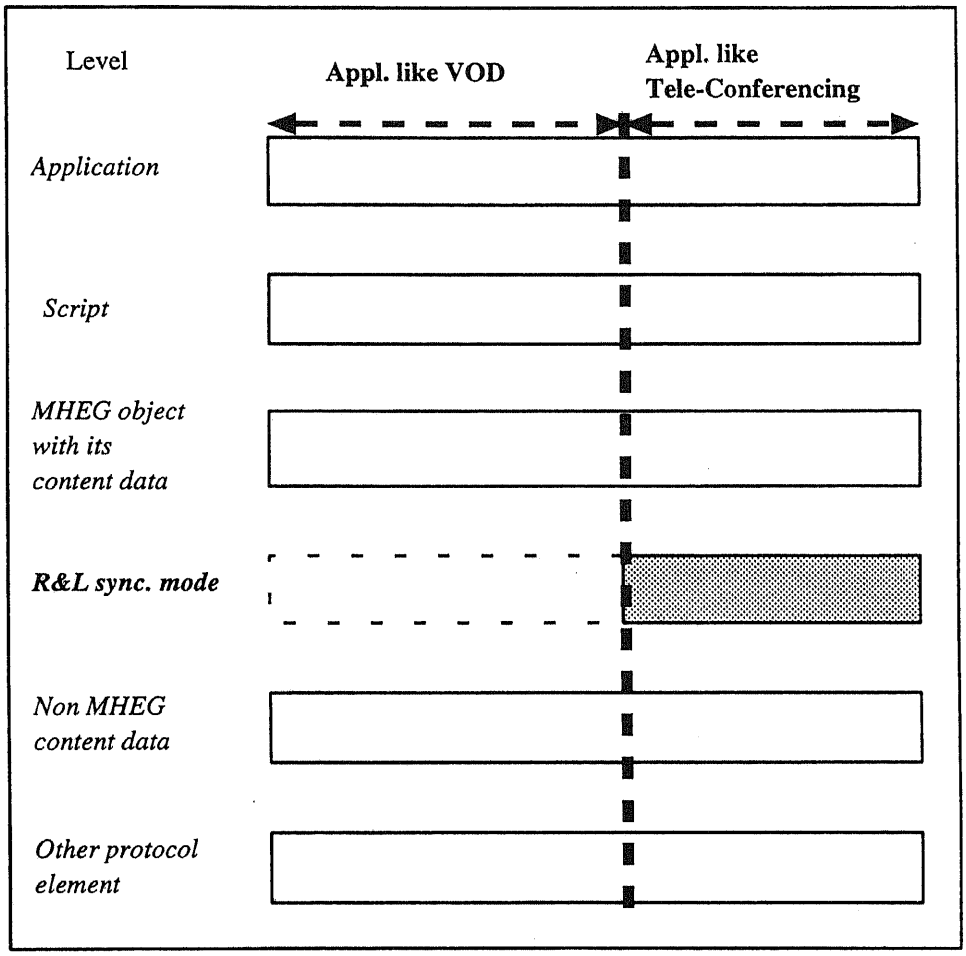


図 3.9: R&L Synchronization mode in the MHEG hierarchy

一例として，図 2.4で示したシナリオを同期時間軸に mapping した場合を図 3.10に示す．図 2.4中のユーザ選択待ちの部分は図 3.10では待ち時間 0 として，分岐後のそれぞれのシナリオに対して同期時間軸の unit 番号が付けられる．そして，MHEG におけるユーザ入力待ちの部分は *R&L sync.mode* のレベルで強制的に一時停止することで実現する．図 3.10の例では 12 unit 目で一時停止し，ユーザの選択を待つ．

このような，R&L 同期，及び，同期時間軸レベルでのシナリオに従った情報提示などといった *R&L sync.mode* での機能は，MHEG での MHEG engine と同様に，R&L engine で実現することとする．

図 3.11に R&L engine の構成の一例を示す．（本例は 5章で述べる試作システムを実現する場合を想定している．）

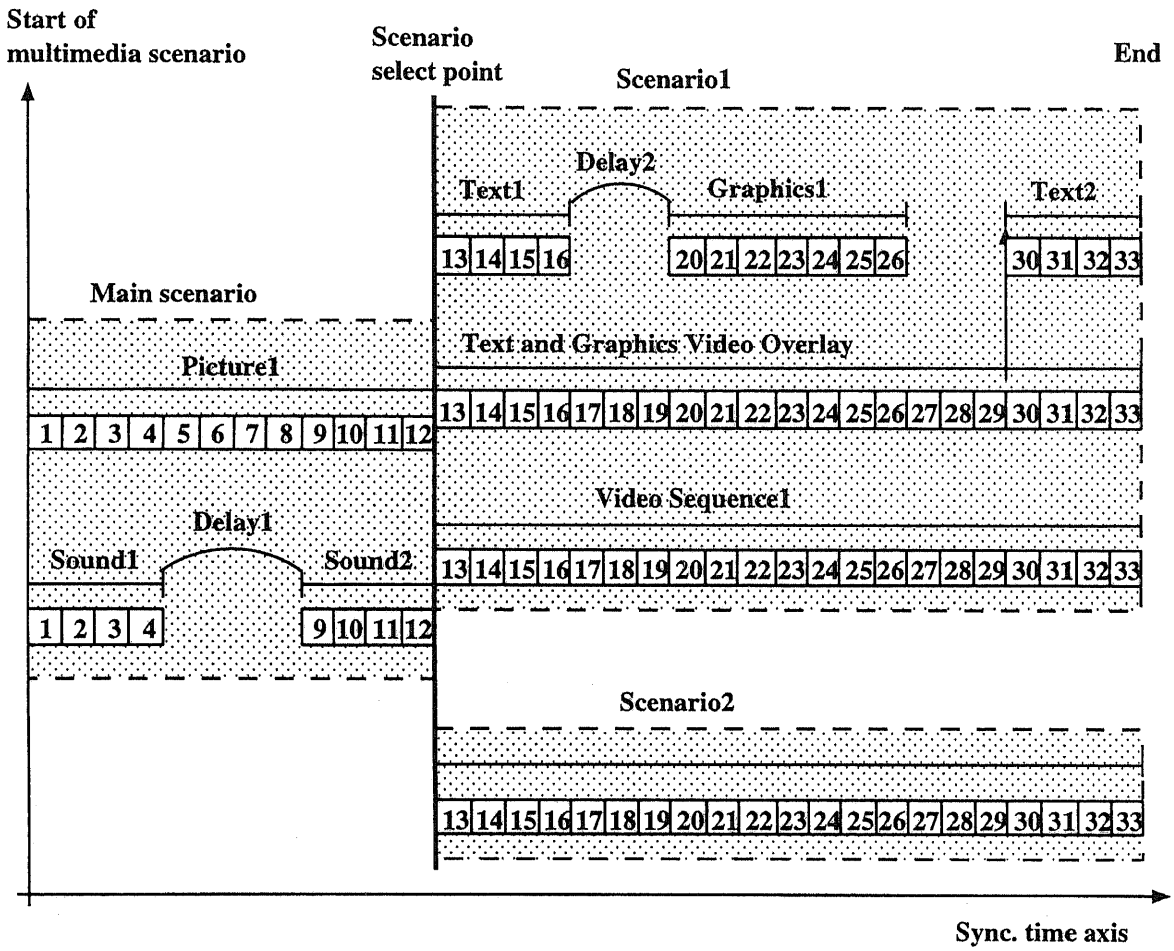


図 3.10: Mapping from MHEG to Sync. time axis

図 3.11において，Scenario List の部分がユーザからの選択待ちによる一時停止位置などを保持している．また，Calc.CDP(Calculate Current Delivery Position) はメディア unit 送出位置の算出，及び，現在位置の保持を行なう．Calc.CPP なども同様の働きをする．

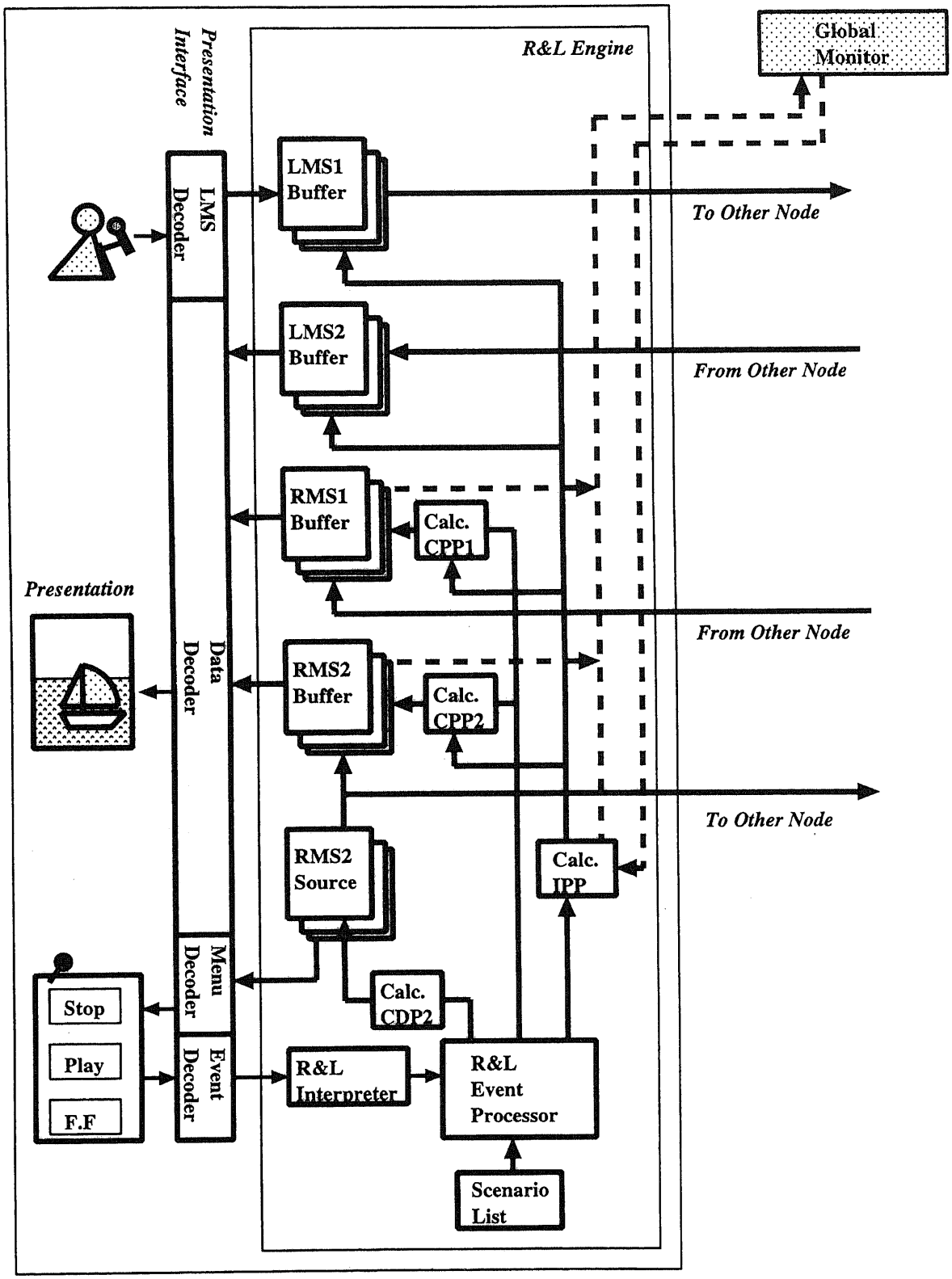


図 3.11: R&L engine

第 4 章

QOS が保証される環境でのバッファ量の検討

本章では予めネットワーク，端末などの資源を確保することでそれらの資源の QOS が保証されるような環境において本システムを設計する際の設計条件について検討する．本検討にでは，命令に対する待ち時間が与えられた際に必要なバッファ量の算出手法，各メディアごとのバッファ量が与えられた際の同期判定及び待ち時間の算出手法，及び，各ノード全体のバッファ量が与えられた際の同期判定及び待ち時間の算出手法の 3 つの手法を示す．また，これらの手法を用いて実際に本システムの設計を行ない，その結果について検討を加える．

4.1 システム条件

4.1.1 想定する環境

ネットワーク ネットワークとしては予め資源を確保することで伝送遅延時間などがある範囲内に収まることを保証できる場合を考える.

端末 端末としてはこのシステムのための専用端末などのように本システムの正常な動作を保証するだけの資源があらかじめ確保できる場合を考える.

情報源 動画, 音声などのような等間隔で情報が発生するメディアの入ったハードディスク, ビデオテープ装置などを考える.

4.1.2 システム条件

想定するシステムとしては図 4.1のような一つの操作者ノード, 複数の情報発生源ノード, 及び, 情報提示ノードからなるものを考える.

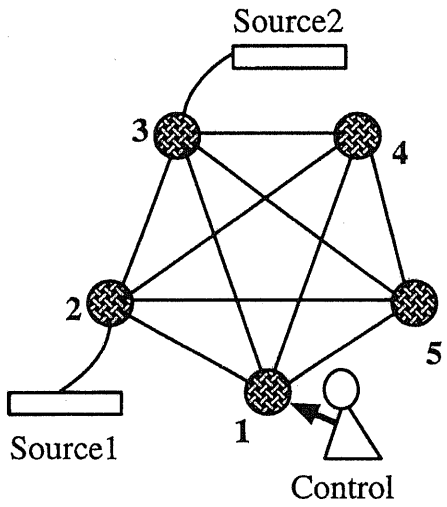


図 4.1: 想定するシステム

ここで, 操作者ノードを 1, 情報発生源ノードを i $(2 \leq i < m)$, 情報提示ノードを j $(j \geq m)$ と表記する. また, 以下の議論を進めていく上で使用する記号を表 4.1に示す.

表 4.1において, T_{ij}, T_{set} などの時間の上限及び下限はあらかじめわかっているものとする.

N	全ノードの数
T_k	各情報源でのメディア発生間隔
T_{ij}	ノードiからノードjまでの遅延 $(1 \leqq i,j \leqq N)$
T_{set}	情報源を各命令に対してセットアップするのに必要な時間（ハードディスクのシークにかかる時間、ビデオテープ装置の機械的な動作にかかる時間など）
T_c	メディア情報の圧縮伸長時間（ＬＭＳ）
T_{dci}	メディア情報の伸長時間（ＲＭＳ）$(2 \leqq i < m)$
D_{ij}	同じ時点でのノードiの再生点に対するノードjの再生点の遅れ $(1 \leqq i,j \leqq N)$
T_d	遅延変動吸収のための必要なバッファ量
F_i	各メディアバッファ中で再生点よりも後、つまり再生方向にある部分 $(1 \leqq i \leqq N)$
P_i	各メディアバッファ中で再生点よりも前、つまり逆再生方向にある部分 $(1 \leqq i \leqq N)$
W_{ri}	各ノードで再生点調整、同期回復などに必要な待ち時間 $(1 \leqq i \leqq N)$
W_{di}	各ノードで伸長が必要なメディアの情報提示の遅れを吸収するのに必要な待ち時間 $(1 \leqq i \leqq N)$
W_s	各ノードで等しい、任意の待ち時間 (Command Pending Time)

表 4.1: システム中で使用されている記号

また，メディア情報の圧縮方式としてはフレーム内圧縮方式のみを考えて以下の議論を進める．フレーム間圧縮を用いる場合については 4.3.3節で述べる．

RMS が圧縮された情報の場合，バッファ中には圧縮された情報をそのまま格納する．伸長は各ノードにおいて実際にメディアフレームが提示される際に行なわれる．また，情報源には圧縮された情報が格納されているものとする．（圧縮が必要な場合にも，圧縮にかかる時間は T_{set} に含めて考えることができる．）

まず，以下の議論を進めていく上での仮定を述べる．

仮定１： 操作命令は必ずその命令により位置が変化したフレームよりも先に到着するとする．つまり，

$$T_{1i} + T_{set} + T_{ij} \geq T_{1j} \qquad (2 \leq i < m, m \leq j \leq N) \qquad (4.1)$$

仮定 2 : 各ノードでは確実に (30 フレーム / 秒などの) 同じ速度で画面の走査が行われているものとする.

仮定 3 : 操作者ノードでの命令発行は, 各走査が完了する時点 (33 ミリ秒ごと等) においてしか発行されないとする.

仮定 4 : あるフレームはスキャンされる直前の時点においてそのフレーム情報を完全にバッファ中に持っている必要があるとする.

次に, 各ノードでの状態と操作命令を表 4.2に示す.

状態 / 命令	一時停止	再生	逆再生	早送り	巻き戻し
一時停止	×	◎	○	○	○
再生	◎	×	×	×	×
逆再生	○	×	×	×	×
早送り	○	○	○	×	○
巻き戻し	○	○	○	○	×
順コマ	◎	×	×	×	×
逆コマ	◎	×	×	×	×

表 4.2: 各状態において実行可能な命令

この表において, ◎または○で示してある部分はその状態に対して対応する命令が実行可能であることを示し, ×に対応する命令は許されないとする. また, ◎で示してある部分は後で述べる R&L 同期のセマンティクスを満たすものとする.

4.1.3 R&L 同期のセマンティクス

本節では 3.4.2節で示した R&L 同期のセマンティクスを表 4.1などで定義された記号を用いて記述する.

sync_with_one RMS と操作者からの LMS との同期. すべてのノードにおいて, 操作者ノードにおいて LMS が取り込まれたのと同じタイミングで, LMS が RMS と同期して提示されること.

式 (4.2) が満たされることが条件となる.

$$D_{i1} + T_{1i} + T_c \leq 0 \qquad (2 \leq i \leq N) \tag{4.2}$$

sync_with_all RMS と全ての参加者からの LMS との同期. すべてのノードでの再生点が各時刻において等しいこと.

式 (4.3) が満たされることが条件となる.

$$D_{ij} = 0 \qquad (1 \leq i, j \leq N) \tag{4.3}$$

sync_with_multi RMS と操作者を含む複数の参加者 (発信者ノード) からの LMS との同期.

すべての発信者ノードでの各時刻における再生点が等しく, それ以外のノードでは, 発信者ノードにおいて LMS が取り込まれたのと同じタイミングで, LMS が RMS と同期して提示されること.

式 (4.4) が満たされることが条件となる. ただし, 式 (4.4) において, N_s は発信者ノード数を示す.

$$\begin{aligned} &D_{ij} = 0 && (1 \leq i, j \leq N_s) \\ &D_{ji} + T_{ij} + T_c \leq 0 && (1 \leq i \leq N_s, N_s < j \leq N) \end{aligned} \tag{4.4}$$

sync_with_none RMS と LMS との同期は考えず, RMS 間の同期のみ保証する.

また, R&L 同期における各ユーザへの情報提示を以下のように規定する.

- 全てのノードにおいて少なくとも操作者ノードと同じだけの情報が提示される.

  (従って, 各ノードで一時停止命令を受けとった際に, 指定された位置までまだ情報提示が完了していない場合, その位置まで情報提示を行なってから停止する.)

- 再生命令に対しては各ノードで R&L 同期のための再生点調整を行なった後, LMS との同期が取れた状態で情報提示が始められる.

- 一時停止, コマ送り命令に対して各ノードでは命令到着後, 即座に反応する. コマ送り中には RMS と LMS との同期は保たれる必要はない. 命令実行後, 一定時間が経過した後, LMS との同期が回復する.

各ノードで再生命令を受け取った際の再生点調整に必要な待ち時間を式 (4.5) に示す.

$$\begin{aligned} W_{r1} &= MAX(0, D_{21}+T_{12}, D_{31}+T_{13}, \cdots, D_{N1}+T_{1N}) \qquad (\text{ノード}1) \\ W_{ri} &= W_{r1} - (D_{i1}+T_{1i}) \qquad (\text{ノード}2 \sim N) \end{aligned} \tag{4.5}$$

ただし, $MAX(x_1, x_2, \cdots, x_n)$ は $x_1, x_2, \cdots, x_n$ の最大値を取るものとする.

次に, 各ノードで一時停止, コマ送り命令実行後, LMS との同期回復に必要な待ち時間を以下に示す.

まず, ノード $i(2 \leq i \leq N)$ での同期回復のためにノード 1 で最低限必要な待ち時間 $temp_i$ は式 (4.6) のようになる.

$$\begin{aligned} temp_i &= T_{1i} \qquad (D_{i1}+T_{1i} > 0) \\ temp_i &= 0 \qquad (D_{i1}+T_{1i} \leq 0) \end{aligned} \tag{4.6}$$

従って, 各ノードで必要な待ち時間は式 (4.7) のようになる.

$$\begin{aligned} W_{r1} &= MAX(0, temp_2, temp_3, \cdots, temp_N) \qquad (\text{ノード}1) \\ W_{ri} &= W_{r1} - T_{1i} \qquad (D_{i1}+T_{1i} > 0) \\ W_{ri} &= 0 \qquad (D_{i1}+T_{1i} \leq 0) \qquad (\text{ノード}2 \sim N) \end{aligned} \tag{4.7}$$

また, 各ノードで情報提示する際に, 伸長が必要なメディアの情報提示は伸長に必要な時間だけ遅れることになる. この遅れを吸収するために, 伸長が必要でない (または伸長にかかる時間が短い) メディアの情報提示を待たせる必要がある. この待ち時間 W_{di} を式 (4.8) に示す.

$T_{dc2}, T_{dc3}, \cdots, T_{dc(m-1)}$ の最大値を T_{dcmax} とすると

$$W_{di} = T_{dcmax} - T_{dci} \qquad (2 \leq i < m) \tag{4.8}$$

R&L 同期のセマンティクスは以上述べてきた必要な待ち時間に加え, 各ノードで等しい, 任意の時間待たせた場合に満たされる.

この, 各ノードで等しい待ち時間を command pending time と呼び, W_s と表記する.

4.2 R&L 同期を満たすために必要なバッファ量の算出式

本節では R&L 同期を満たすために必要なバッファ量を求めるための計算式を導き出す.

各ノードでは各情報源ごとにバッファを持っているので, 図 4.2のように, システムは情報発生源ノードごとに分割して考えることができる.

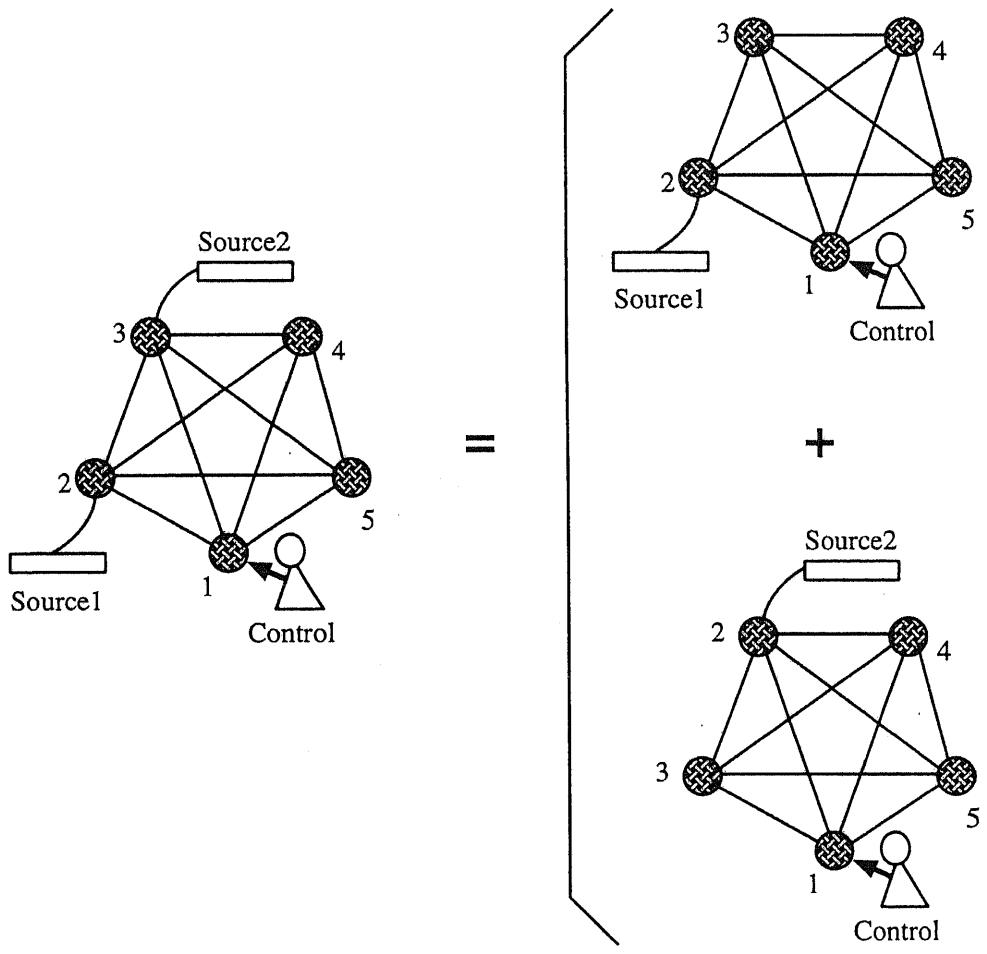


図 4.2: システムの分割

そこで, 以下, 図 4.2の分割後のような 1 つずつの操作者ノードと情報発生源ノード及び複数の情報提示ノードからなるシステムについて考えていく.

ここで, 図 4.2のように分割後のシステムで操作者ノードが 1, 情報発生源ノードが 2, 情報提示ノードが i $(3 \leq i \leq N)$ となるように番号を振り直す. また, T_{dci},W_{di} $(2 \leq$

$i < m$) は i を省き，T_{dc},W_d と表記する．さらに，各メディアバッファ中で再生点よりも後，つまり再生方向にある部分（以下，future バッファと呼ぶ．）の大きさを $F_i(1 \leq i \leq N)$，再生点よりも前，つまり逆再生方向にある部分（以下，past バッファと呼ぶ．）の大きさを $P_i(1 \leq i \leq N)$ と表記する．

以下に R&L 同期を満たすために必要なバッファ量を算出するための式を示す．

4.2.1 一時停止状態

一時停止状態はすべてのノードで再生点が同じ位置で止っている状態と規定する．ここで，仮定 3 より一時停止状態では必ず再生点はフレームの区切りに位置している．

- 再生命令

  各ノードで命令を受けてから，情報源からのメディアフレームが到着するまでの時間は以下のようになる．

$$\begin{array}{ll} T_{12} + T_{set} + T_{21} & (\text{ノード}1) \\ T_{set} & (\text{ノード}2) \\ T_{12} + T_{set} + T_{2i} - T_{1i} & (\text{ノード}3 \sim N) \end{array} \tag{4.9}$$

  各ノードの future バッファには少なくとも再生開始後，情報源からのデータが到着するまでに提示されるメディア情報が蓄えられている必要がある．仮定 2 よりこの間に提示されるメディア情報の大きさはその間の経過時間で表すことができる．

  つまり，以下の条件が満たされる必要がある．

$$\begin{array}{ll} T_{12} + T_{set} + T_{21} - W_{r1} - W_s - W_d + T_k < F_1 & (\text{ノード}1) \\ T_{set} - W_{r2} - W_s - W_d + T_k < F_2 & (\text{ノード}2) \\ T_{12} + T_{set} + T_{2i} - T_{1i} - W_{ri} - W_s - W_d + T_k < F_i & (\text{ノード}3 \sim N) \end{array} \tag{4.10}$$

  式 (4.10) 中の T_k は，情報源からのデータが到着する直前の時点で，仮定 4 より future バッファが T_k より大きくなければならないために必要となる．

  また，情報源が再生命令受信後，各ノードに一時停止中の再生点よりも L 先のデータからメディア情報を送り始めるとすると，L は式 (4.10) の左辺よりも大きいこ

とが必要である．つまり，以下の条件が満たされることが必要である．

$$\begin{aligned} L \ > \ & MAX(T_{12}+T_{set}+T_{21}-W_{r1}-W_s-W_d+T_k, T_{set}-W_{r2}-W_s-W_d \\ & +T_k, T_{12}+T_{set}+T_{23}-T_{13}-W_{r3}-W_s-W_d+T_k, \ldots, T_{12}+T_{set} \\ & +T_{2N}-T_{1N}-W_{rN}-W_s-W_d+T_k) \end{aligned} \tag{4.11}$$

この場合さらに，バッファの連続性を保つために

$$L \leq F_i \qquad (1 \leq i \leq N) \tag{4.12}$$

が満たされることが必要となる．

また，情報源からのデータが表示開始より先に到着する場合，先に到着する分のバッファも確保する必要がある．つまり，式 (4.13) の結果が正の値ならば，その大きさのバッファを L 以外に確保する必要がある．

$$\begin{aligned} & W_{r1}+W_s+W_d-T_{12}-T_{set}-T_{21} && (\text{ノード}1) \\ & W_{r2}+W_s+W_d-T_{set} && (\text{ノード}2) \\ & W_{ri}+W_s+W_d-T_{12}-T_{set}-T_{2i}+T_{1i} && (\text{ノード}3 \sim N) \end{aligned} \tag{4.13}$$

- 順コマ送り命令

順コマ送りの場合，一コマあたり T_k 時間進むとすると，一コマだけなら

$$T_k < F_1 \tag{4.14}$$

が満たされればよい．

次に，連続 M 回，T_b $(T_b > T_k)$ 時間ごとにコマ送りボタンを操作者が押す場合について考える．この場合も，各ノードで命令を受けてから，情報源からのメディアフレームが到着するまでの時間は式 (4.9) で表される．コマ送り命令は命令到着後即座に実行されるため，各ノードの future バッファには少なくとも命令到着後，情報源からのデータが到着するまでに提示されるメディア情報が蓄えられている必要がある．

つまり，式 (4.15) が満たされる必要がある．

$$\begin{aligned} & ((T_{12}+T_{set}+T_{21}-W_d)/T_b+1)T_k < F_1 && (\text{ノード}1) \\ & ((T_{set}-W_d)/T_b+1)T_k < F_2 && (\text{ノード}2) \\ & ((T_{12}+T_{set}+T_{2i}-T_{1i}-W_d)/T_b+1)T_k < F_i && (\text{ノード}3 \sim N) \end{aligned} \tag{4.15}$$

また，情報源が順コマ送り命令受信後，一時停止中の再生点よりも L 先のデータからメディア情報を送り始めるとすると，再生命令の場合と同様，以下の条件が満たされる必要がある．

$$\begin{aligned} L \ > \ & MAX(((T_{12}+T_{set}+T_{21}-W_d)/T_b+1)T_k, \\ & ((T_{set}-W_d)/T_b+1)T_k, ((T_{12}+T_{set}+T_{23} \\ & -T_{13}-W_d)/T_b+1)T_k, \ldots, ((T_{12}+T_{set} \\ & +T_{2N}-T_{1N}-W_d)/T_b+1)T_k) \end{aligned} \tag{4.16}$$

$$F_i \ \geq \ L \qquad (1 \leq i \leq N) \tag{4.17}$$

- 逆コマ送り命令

  逆コマ送り命令も順コマ送り命令の場合と同様にして，必要なバッファ量を求めることができる．

4.2.2 再生状態

再生状態はそれぞれのノードで適当な再生点を取りながら再生が行なわれている状態と規定する．

- 一時停止命令

  操作者ノードでは一時停止ボタンが押された時の再生点で無条件に一時停止する．また，操作者ノード以外のノードで一時停止命令を受けた時にそのノードの再生点が一時停止位置に達していなかった場合，一時停止位置まで再生を続け，無条件に一時停止する．逆に，一時停止命令を受けた時に一時停止位置を過ぎていた場合には，その時点で past バッファ中に一時停止位置のフレームが入っている必要がある．つまり，以下の条件が満たされる必要がある．

$$D_{i1}+T_{1i}+T_{dc}+T_k < P_i \qquad (\text{ノード}2 \sim N) \tag{4.18}$$

  ただし，情報源からのデータが表示開始よりも先に到着する場合には一時停止位置のフレームを得ることができるので式 (4.18) の条件は必要ない．

以上，まとめると，それぞれのノードで一時停止が行なえる条件は式 (4.19) のようになる．

$$\begin{array}{lll} \text{無条件} & & (\text{ノード}1) \\ D_{21}+T_{12}+T_{dc}+T_k<P_2 & (W_{r2}+W_s+W_d<T_{set}) & \\ \text{無条件} & (W_{r2}+W_s+W_d\geq T_{set}) & (\text{ノード}2) \\ D_{i1}+T_{1i}+T_{dc}+T_k<P_i & (W_{ri}+W_s+W_d<T_{12}+T_{set}+T_{2i}-T_{1i}) & \\ \text{無条件} & (W_{ri}+W_s+W_d\geq T_{12}+T_{set}+T_{2i}-T_{1i}) & \\ & & (\text{ノード}3\sim N) \end{array} \tag{4.19}$$

また，再生状態では各ノードへの情報源からのメディア情報の到着のずれを吸収するためのバッファが必要である．

情報源以外のノード $i(i=1,3,4,\ldots,N)$ では情報源ノードに較べ，メディア情報の到着が T_{2i} だけ遅れ，再生点は D_{i2} だけ進んでいる．つまり，情報源以外のノードの再生中の future バッファの大きさは F_2 を用いて以下のように表される．

$$F_i=F_2-D_{i2}-T_{2i} \qquad (i=1,3,4,\ldots,N) \tag{4.20}$$

また，各ノードの future バッファの大きさは遅延変動吸収のために必要なバッファサイズ T_d より大きいことが必要である．つまり，

$$F_i>T_d \qquad (1\leq i\leq N) \tag{4.21}$$

再生中の future バッファの大きさは式 (4.20)，(4.21) が同時に満たされるように決められる必要がある．

4.3 システム構成に関する考察

本章では R&L 同期を満たすために必要なバッファ量を求めるための計算式を使って幾つかの具体的な数値例を示し，算出結果について考察を加える．

4.3.1 算出条件

4.1.3節で述べた R&L 同期のセマンティクスを満たすために最低限必要なバッファ量を図 4.2の構成のネットワークについて求める．

ここで情報源としては動画情報の入った RAM ディスク（T_{set} = [0ミリ秒, 0ミリ秒]），動画情報の入ったハードディスク（T_{set} = [0ミリ秒, 30ミリ秒]），動画情報の入ったビデオテープ装置（T_{set} = [500ミリ秒, 1000ミリ秒]）の 3 つを想定する．ただし，[x_{min}, x_{max}] は最小値 x_{min} から最大値 x_{max} までの範囲内の値を取るものとする．

また，ネットワークとしては，低軌道，中軌道，静止軌道等を用い，衛星間中継を含む，マルチメディア時代に想定される遅延の大きな衛星通信回線を含むネットワークモデルを想定した．各遅延は可変であるとし，遅延をノード間の遅延として図 4.2の 5 角形でモデル化している．遅延時間の一例として短い方の辺の遅延 (T_{12}, T_{23}など) を [270 ミリ秒, 300 ミリ秒]，大きい方の辺の遅延 (T_{13},T_{14} など) を [432 ミリ秒, 480 ミリ秒] とした．

伸長・圧縮に伴う遅延は T_c = 0ミリ秒 とし，T_{dc} は伸長が必要な場合 33ミリ秒，必要でない場合 0ミリ秒 とした．

また，T_k = 33ミリ秒,T_b = 100ミリ秒,T_d = 33ミリ秒 とした．

各ノードで再生命令を受け取った際に再生点調整を行なうのに必要な待ち時間，及び，一時停止，コマ送り命令実行後 LMS との同期回復に必要な待ち時間を sync_with_one, sync_with_all の場合について表 4.3に示す．

	node1	node2	node3	node4	node5
sync_with_one	0ms	15ms	24ms	24ms	15ms
sync_with_all	480ms	195ms	24ms	24ms	195ms

表 4.3: 各ノードで必要な待ち時間

4.3.2 算出結果

算出結果を以下に示す．ただし，ここで示すグラフは全て横軸が W_s，縦軸が各ノードで必要なバッファの大きさとなっている．ここで，必要なバッファの大きさは T_{set}，T_{ij} $(1 \leq i, j \leq N)$ などがその取り得る値の範囲内でどのような値を取ったとしても必要十分な大きさとする．また，バッファの大きさは時間（何ミリ秒分相当か）で表す．

- 例 1

  情報源がハードディスクで source1(図 4.2)，$T_{dc} = 33$ミリ秒 の場合に sync_with_one のセマンティクスを満たすために各ノードで最低限必要なバッファの大きさを図 4.3に示す．

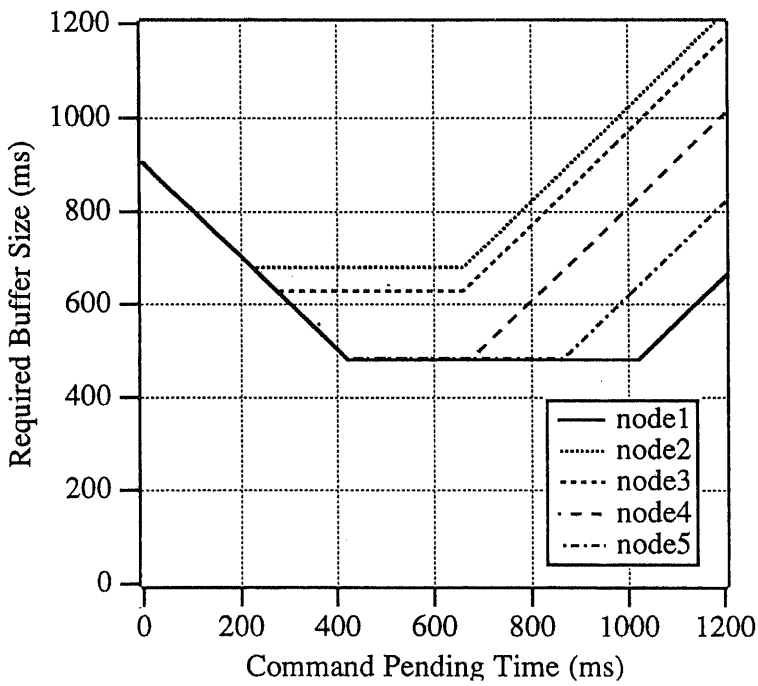


図 4.3: 必要なバッファ量（例 1）

図 4.3は以下の傾向を示す．

- W_s を増やすにつれ，必要なバッファの大きさは始めは減少するが，ある程度増やしたところでそれ以上減らなくなる。さらに，W_s を増やしていくと，必要なバッファの大きさは逆に増加し始める．

- 例 2

  図 4.4は情報源が source2 で，それ以外の条件は例 1 と同じ場合に必要なバッファの大きさを示している．

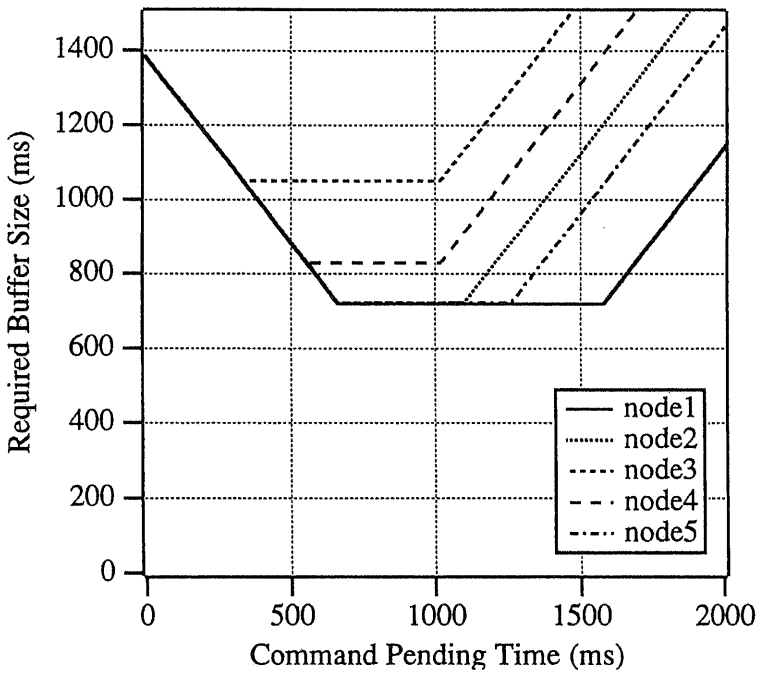


図 4.4: 必要なバッファ量（例 2）

例 2 の場合も例 1 と同様の傾向を示す．

また，図 4.3, 図 4.4は以下に示す傾向も示している．

- W_r も含めて考えると，操作者ノードに近いノードほど必要なバッファ量が小さく，情報源ノードに近いノードほど必要なバッファ量は大きい．

- 例 3

図 4.5は例 1 と同じ条件で sync_with_all のセマンティクスを満たすために必要なバッファの大きさを示している.

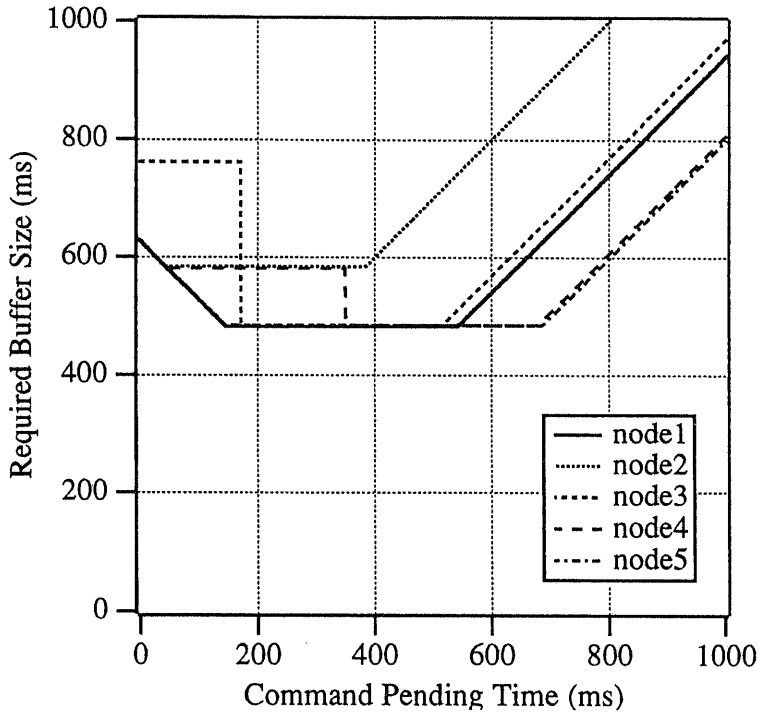


図 4.5: 必要なバッファ量（例 3）

図 4.5は以下の傾向を示す.

- ノード 3, 4 などでは W_s が小さい領域で必要なバッファの大きさが不連続な値を取る.

- 例 4

図 4.6は情報源が RAM ディスクで，それ以外の条件は例 1 と同じ場合に必要なバッファの大きさを示している．

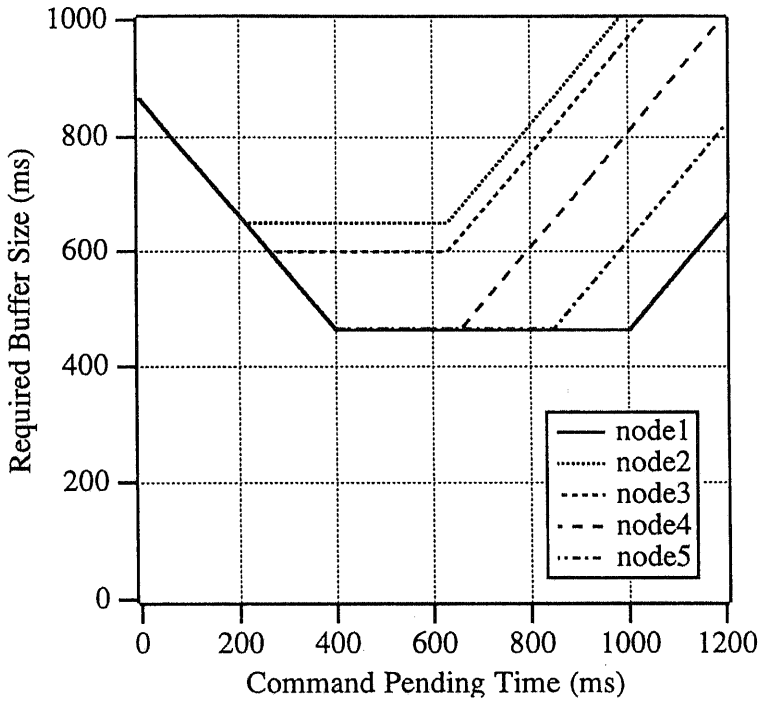


図 4.6: 必要なバッファ量（例 4）

図 4.6は以下の傾向を示す．

– RAM ディスクの場合，ハードディスクの場合よりも必要なバッファの大きさは全体としてわずかに小さい．

- 例 5

図 4.7は情報源がビデオテープ装置で，それ以外の条件は例 1 と同じ場合に必要なバッファの大きさを示している．

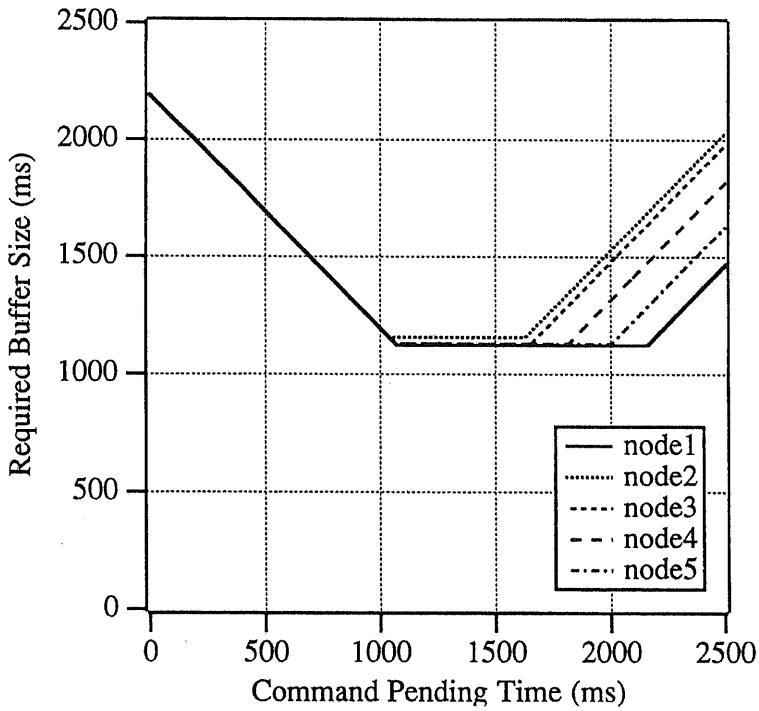


図 4.7: 必要なバッファ量（例 5）

図 4.7は以下の傾向を示す．

– ビデオテープ装置の場合，ハードディスクの場合よりもかなり大きなバッファが必要となる．

4.3.3 考察

前節で示したグラフから以下のことがわかる.

- W_s をある程度増やすことにより必要なバッファの大きさを削減できる.
  しかし, W_s をある程度以上増やすと, 情報提示より先に到着する分のバッファによりかえって必要なバッファ量が増加する.
- W_r も含めて考えると, 操作者ノードに近いノードほど必要なバッファ量が小さく, 情報源ノードに近いノードほど必要なバッファ量は大きい.
- コマ送りのために必要なバッファの大きさが各グラフにおけるバッファの大きさの最小値となっている.
- sync_with_all の場合, ノード 3, 4 などでは再生状態からの一時停止に必要なバッファのために W_s が小さい領域である程度の大きさのバッファを必要とする. また, このバッファは式 (4.19) に示したように, W_s がある一定値を超えると必要なくなる. 例 3 の場合に必要なバッファの大きさが不連続な値を取るのはこのためである.
- W_s と共に必要なバッファの大きさが減少 (増加) している部分, 及び, 例 1 の場合のノード 2,3 のように最低値を取らずに平になっている部分は一時停止状態からの再生命令に必要なバッファの大きさである.
- バッファをある程度大きく取ることで, 応答性を良くすることができる. 例えば, 例 1 や例 2 の場合, sync_with_one のセマンティクスを満たすために最低限必要なバッファを用意した場合よりもさらに大きなバッファを用意することで W_s を 500 ミリ秒程度削減することができる. つまり, 大きめのバッファを用意することで 15 フレーム分程度, 応答性を良くすることができる.

また, 例 4, 例 5 の場合も, グラフ全体の傾向は例 1 の場合と変わらない. ただし, T_{set} の値の違いにより以下に述べるような傾向を示す.

- RAM ディスクの場合, ハードディスクの場合よりも必要なバッファの大きさは全体としてわずかに小さい.
- ビデオテープ装置の場合, T_{set} が大きいため, 1000ミリ秒 ～ 2000ミリ秒 分相当のかなり大きなバッファが必要となる.

本節では，操作権を持つノードを固定して必要なバッファの大きさを求めてきたが，実際には操作権が複数のノード間で受け渡される場合も多いであろう．そのような場合には，操作権が全ての位置にある場合について必要なバッファの大きさを求め，各ノードでその最大値を求めることで必要なバッファの大きさを求めることができる．

また，MPEG[Yasu 91] のようなフレーム間圧縮を用いる場合，バッファ中の必要情報が即座に伸長，提示できるように future 方向には次の Intra フレーム又は Predicted フレームまでのフレーム情報を，past 方向には次の Intra フレームまでのフレーム情報を持っている必要がある．つまり，算出式により求まるバッファ以外に，さらに数フレーム分の情報が必要となる．

また，3.4.3節で述べた場合のようにメディアごとに発生間隔が異なる場合，発生間隔が小さいメディアでは仮定 4 を満たすために必要な 1 フレーム分の情報が小さくなるため，式 (4.10) 中の T_k などのようにそのために必要なバッファはそのメディアの発生間隔分の大きさでよい．（ただし，コマ送りなどの際の T_k は発生間隔の大きいメディアのものを用いるため変更する必要はない．）しかし，それ以外の点は変更することなく 4.2節で述べた算出式を用いて必要なバッファの大きさを算出できる．4.4節，4.6節の算出法についても同様のことが言える．

4.4 メディアごとのバッファの大きさが与えられた際の R&L 同期判定と必要な待ち時間の算出式

端末として専用端末が用いられる場合などには，予め各端末に各メディアごとのバッファ（フレームバッファなど）が設けられていることを想定することができる．

本節では，各ノードで各メディアごとのバッファ量が予め与えられている場合に R&L 同期が満たされるかの判別，及び，満たされる場合に必要な待ち時間の導出を行なうための計算式を導き出す．

R&L 同期判定，及び，必要な待ち時間の算出の手順を以下に示す (図 4.8)．

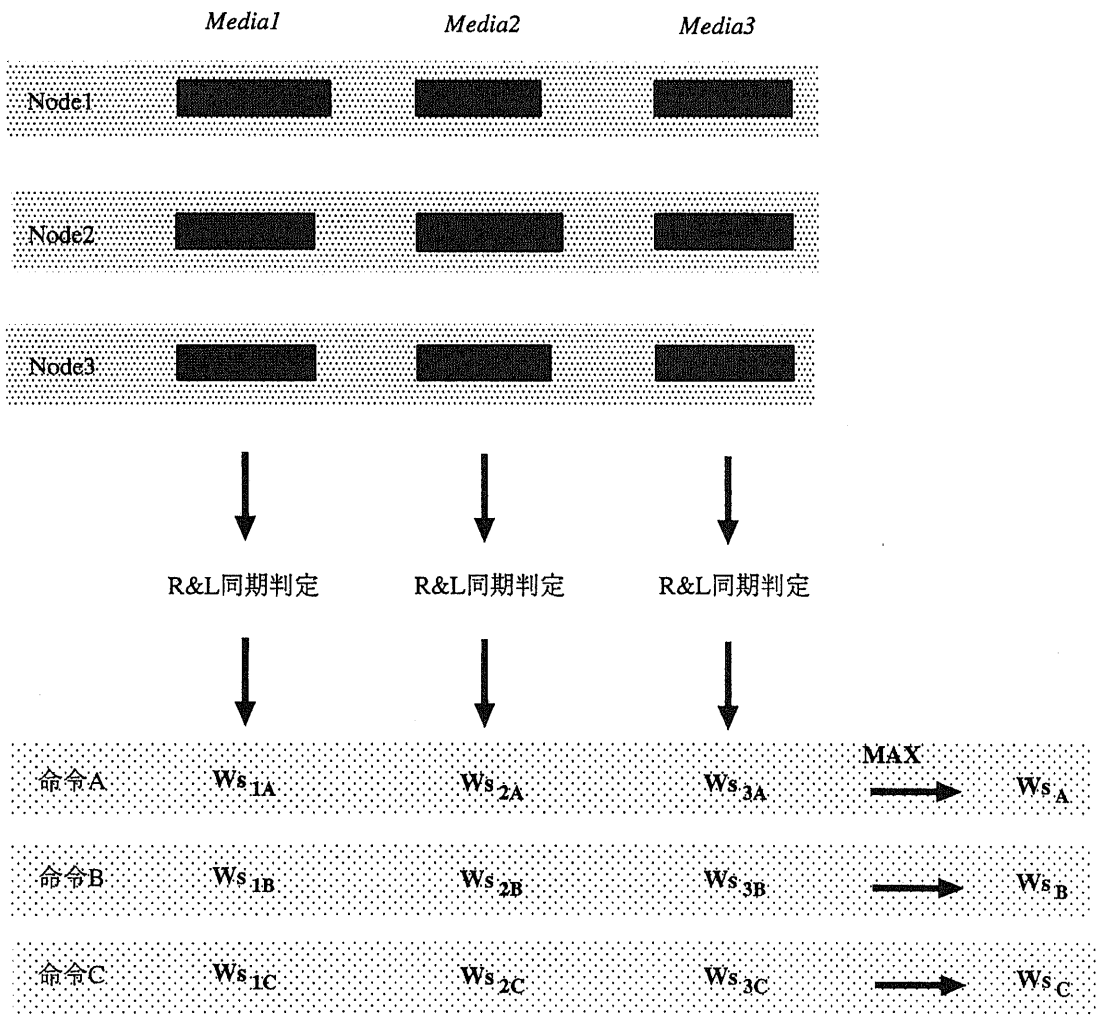


図 4.8: R&L 同期判定，及び，必要な待ち時間の算出の手順

1. 各情報源ごとに分割した各々の構成 (図 4.2) において各命令ごとに R&L 同期判定を行なう． R&L 同期の semantics が満たされる場合にはすべてのノードで等しい W_s を算出する．

2. この算出した W_s を全ての分割したシステムから持ち寄り，各命令ごとに比較し，各命令ごとの最大値をその命令に関する待ち時間とする．

3. 各情報源では最大値に達しなかった命令について，最大値との差に相当する時間の分だけ情報源からのメディア情報の送出開始を遅らせる．

以下，各情報源ごとに分割した各々の構成において各命令についての R&L 同期判定，及び， W_s の算出を行なうための式を示す．

ここで，各ノードのバッファの大きさを $FP_i(i = 1, 2, \ldots, N)$ とおく．

4.4.1 一時停止状態

- 順コマ送り命令

  一コマあたり T_k 時間進むとすると，一コマだけなら

$$T_k < F_1 \tag{4.22}$$

  が満たされればよい．

  連続 M 回， T_b $(T_b > T_k)$ 時間ごとにコマ送りボタンを操作者が押す場合，以下の式が満たされる必要がある．

$$\begin{aligned} F_i \ \geq \ & MAX(((T_{12} + T_{set} + T_{21} - W_d)/T_b + 1)T_k, \\ & ((T_{set} - W_d)/T_b + 1)T_k, ((T_{12} + T_{set} + T_{23} \\ & -T_{13} - W_d)/T_b + 1)T_k, \ldots, ((T_{12} + T_{set} \\ & +T_{2N} - T_{1N} - W_d)/T_b + 1)T_k) \qquad (1 \leq i \leq N) \end{aligned} \tag{4.23}$$

  この条件が満たされなければ R&L 同期のセマンティクスを満たすことはできない．

  また，この条件が満たされれば

$$W_s = 0 \tag{4.24}$$

となる.

- 逆コマ送り

  逆コマ送りの場合も順コマ送りの場合と同様に同期条件を求めることができる.

- 再生命令

  コマ送り命令, 逆コマ送り命令で必要な future バッファ, past バッファの大きさをそれぞれ FB_i, PB_i $(1 \leq i \leq N)$ とする.

  情報源からは以下に示す式で求まる L 先のデータから各ノードに情報が送られる.

$$L = MIN(FP_1 - PB_1, FP_2 - PB_2, \ldots, FP_N - PB_N) \tag{4.25}$$

この場合, W_s は以下の式により求まる.

$$\begin{aligned} W_s \;=\; & MAX(0, T_{12} + T_{set} + T_{21} - W_{r1} - W_d + T_k - L, \\ & T_{set} - W_{r2} - W_d + T_k - L, T_{12} + T_{set} + T_{23} - T_{13} \\ & -W_{r3} - W_d + T_k - L, \ldots, T_{12} + T_{set} + T_{2N} - T_{1N} \\ & -W_{rN} - W_d + T_k - L) \end{aligned} \tag{4.26}$$

また, 情報源からのデータが表示開始より先に到着する場合, 先に到着する分のバッファが確保できなければ R&L 同期の semantics を満たすことはできない. つまり, 式 (4.27) の結果が正の値ならば, その大きさのバッファが L 以外に確保できる必要がある.

$$\begin{aligned} & W_{r1} + W_s + W_d - T_{12} - T_{set} - T_{21} && (\text{ノード}1) \\ & W_{r2} + W_s + W_d - T_{set} && (\text{ノード}2) \\ & W_{ri} + W_s + W_d - T_{12} - T_{set} - T_{2i} + T_{1i} && (\text{ノード}3 \sim N) \end{aligned} \tag{4.27}$$

4.4.2 再生状態

- 一時停止命令

  まず再生中の各ノードへの情報源からのメディア情報の到着のずれ及びネットワークの遅延変動を吸収するために最低限必要な future バッファの大きさ FN_i $(i = 1, 2, \ldots, N)$ を求める (式 (4.20), (4.21)).

ここで，式 (4.28) となるノードが一つでも存在すれば R&L 同期の semantics を満たすことはできない．

$$FP_i - FN_i < 0 \qquad (1 \leq i \leq N) \tag{4.28}$$

次に，R&L 同期の semantics を満たす場合の W_s を求める．

$$D_{i1} + T_{1i} + T_{dc} + T_k < FP_i - FN_i \qquad (1 \leq i \leq N) \tag{4.29}$$

ならば，

$$TW_i = 0 \tag{4.30}$$

それ以外ならば

$$\begin{aligned} TW_1 &= 0 && (\text{ノード}1) \\ TW_2 &= T_{set} - W_{r2} - W_d && (\text{ノード}2) \\ TW_i &= T_{12} + T_{set} + T_{2i} - T_{1i} - W_{ri} - W_d && (\text{ノード}3 \sim N) \end{aligned} \tag{4.31}$$

として (ただし $TW_i < 0$ ならば $TW_i = 0$)，

$$W_s = MAX(TW_1, TW_2, \ldots, TW_N) \tag{4.32}$$

となる．

4.5 システム構成に関する考察

本節では R&L 同期判定，及び，必要な待ち時間の算出を行なうための計算式を使って幾つかの具体的な数値例を示し，算出結果について考察を加える．

4.5.1 算出条件

算出条件は 4.3.1節の場合と同じ条件とする．

4.5.2 算出結果

算出結果を以下に示す．

まず，各メディアバッファの大きさが全てのノードで等しい場合のバッファの大きさと W_s の関係を示す．（ここでは各メディアバッファが等しい場合についての結果のみ示すが，それ以外の場合にも 4.4節で述べた算出式を用いて同様に算出できる．）

次に，求めたグラフを用いて実際に各メディアバッファの大きさが与えられた時の同期判定及び W_s の算出を行なう．

バッファの大きさと W_s の関係

ここで示すグラフはすべて横軸がバッファの大きさ，縦軸が必要な W_s となっている．

また，必要な W_s は T_{set}，$T_{ij}(1 \leq i,j \leq N)$ などがその取り得る値の範囲内でどのような値を取ったとしても必要十分な大きさとする．

- 例 1

  情報源がハードディスクで source1, $T_{dc} = 33$ミリ秒 の場合に sync_with_one のセマンティクスを満たすために各ノードで最低限必要な W_s を図 4.9に示す.

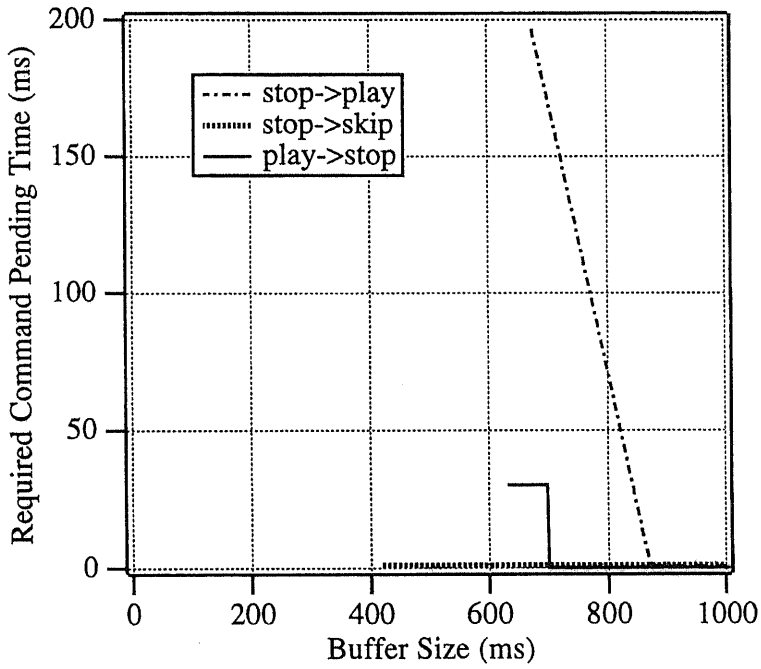


図 4.9: 必要な W_s （例 1）

図 4.9は以下の傾向を示す.

- *stop* → *skip* では常に必要な W_s が 0 となる.
- *play* → *stop* では W_s が不連続な値を取る.
- *stop* → *play* では W_s が線形に減少する.
- *stop* → *play* が同期判定及び W_s の算出を行なう際に最も厳しい条件となる.

- 例 2

図 4.10は情報源が source2 で，それ以外の条件は例 1 と同じ場合に必要な W_s を示している．

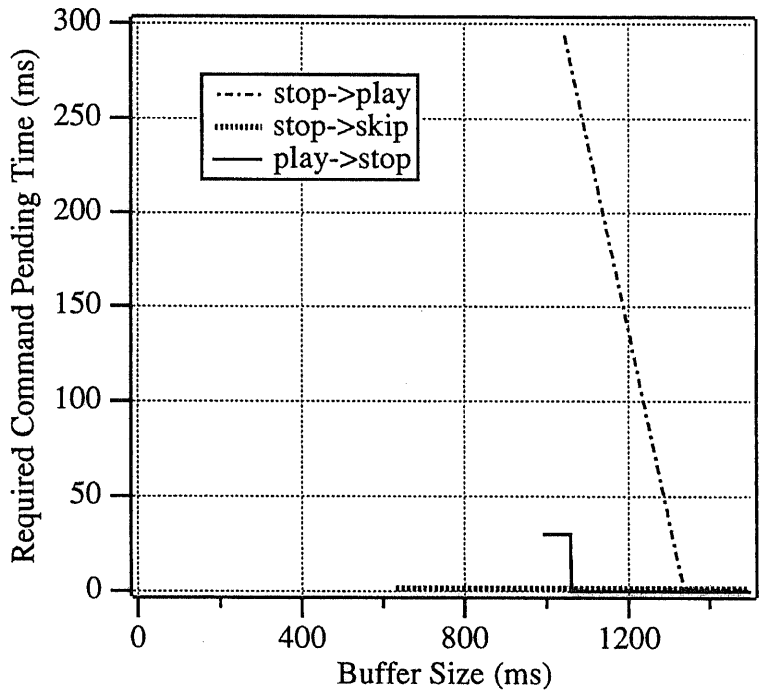


図 4.10: 必要な W_s （例 2）

図 4.10は例 1 の場合と同様な傾向を示す．

- 例 3

  図 4.11は例 1 と同じ条件で sync_with_all のセマンティクスを満たすために必要な W_s を示している.

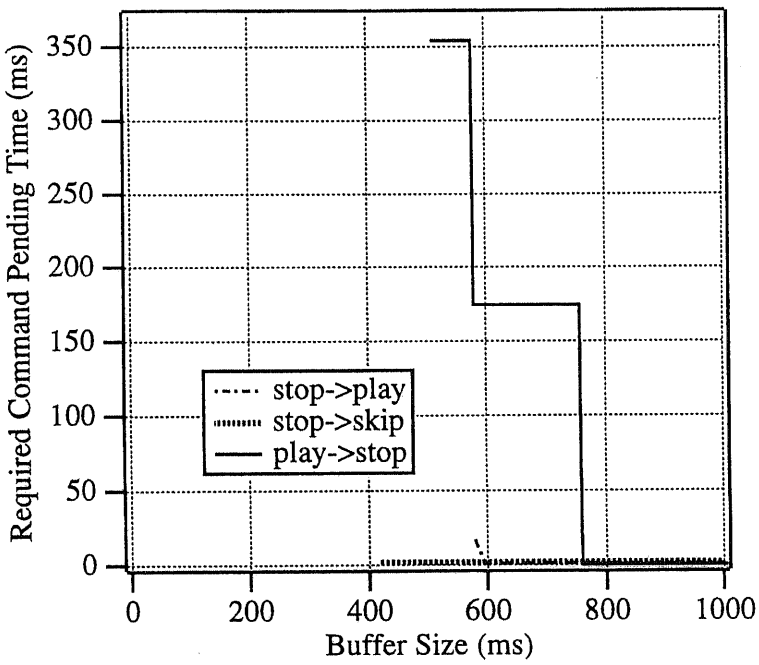


図 4.11: 必要な W_s （例 3 ）

図 4.11は以下の傾向を示す.

- *play* → *stop* が 2 段階になっている.
- *stop* → *play* に必要な W_s は sync_with_one の場合に比べて小さい.

- 例 4

図 4.12は情報源が RAM ディスクでそれ以外の条件は例 1 と同じ場合に必要な W_s を示している.

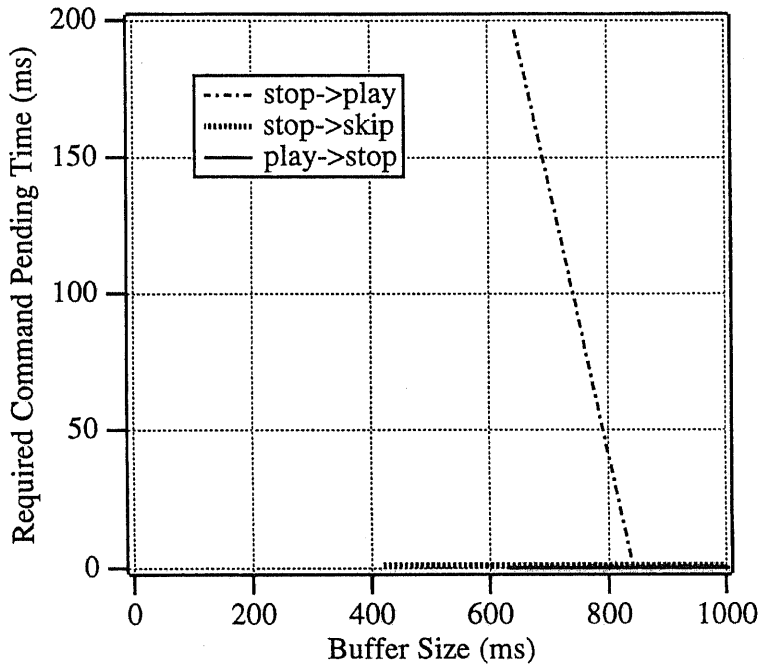


図 4.12: 必要な W_s （例 4 ）

図 4.12は以下の傾向を示す.

- *play* → *stop* でも W_s が常に 0 となる.

- 例 5

図 4.13は情報源がビデオテープ装置で，それ以外の条件は例 1 と同じ場合に必要な W_s を示している．

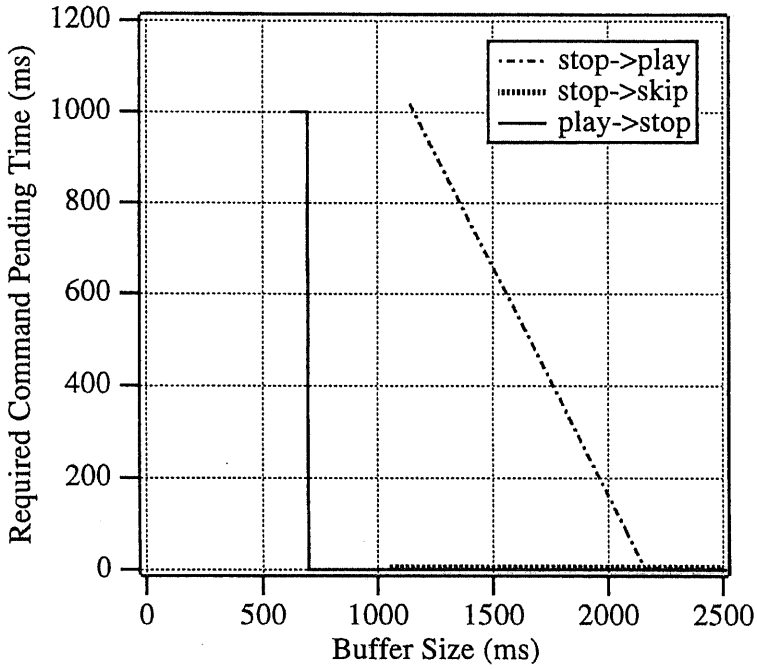


図 4.13: 必要な W_s （例 5）

図 4.13は以下の傾向を示す．

– 例 1 の場合に比べ大きなバッファがないと同期のセマンティクスを満たせない．また，その際に必要な W_s も全体的にかなり大きくなる．

R&L 同期判定と必要な W_s の算出

前節で求めたグラフを利用して実際に R&L 同期判定と必要な W_s の算出を試みる.

- 例 1 において各ノードの source1 に対するメディアバッファの大きさが $678ms$ よりも少なければ, sync_with_one のセマンティクスを満たすことはできない.
- 例 1, 例 2 において各ノードの source1, source2 に対するメディアバッファの大きさがそれぞれ $800ms$, $1200ms$ の場合, sync_with_one のセマンティクスを満たすことができる.

  停止状態から再生を行なう際に必要な W_s は source1 に対して $74ms$, source2 に対して $141ms$ であり, システム全体の $stop \rightarrow play$ 命令に対する W_s は $141ms$ となる. また, この場合 source1 からのメディア送出を $67ms$ 遅らせる. 他の命令に対する W_s は $0ms$ である.

4.5.3 考察

前節で示したグラフなどから以下のことがわかる.

- sync_with_one の場合, $stop \rightarrow play$ が同期判定及び W_s の算出を行なう際に最も厳しい条件となる.
- sync_with_all の場合, $play \rightarrow stop$ の影響が大きい.
- 例 3 において, $play \rightarrow stop$ が 2 段階になっているのは式 (4.29) の条件が満たされるバッファの大きさ, 及び, 式 (4.31) の値が各ノードで異なるためである.

本節では, 操作権を持つノードを固定して必要な R&L 同期判定, 及び, W_s の算出をおこなってきた. 操作権が複数のノード間で受け渡される場合には, 操作権が操作者になりうる全てのノードにある場合について R&L 同期判定を行ない, 全ての場合に R&L 同期が満たされることを確かめる必要がある. また, R&L 同期が満たされる場合, 操作権の位置ごとに W_s の算出, 情報源からの送出待ち時間の算出などを行なう必要がある.

また, MPEG[Yasu 91] のようなフレーム間圧縮を用いる場合, future 方向には次の Intra フレーム又は Predicted フレームまでのフレーム情報を, past 方向には次の Intra フレームまでのフレーム情報を余分に持っている必要があるため, そのフレーム情報分の

大きさ（数フレーム分）を与えられたバッファの大きさから予め差し引いて R&L 同期判定，及び，W_s の算出を行なう必要がある．

4.6 メディア全体のバッファの大きさが与えられた際の R&L 同期判定と必要な待ち時間の算出式

ワークステーションの主記憶のように，全てのメディアが全体として一つのバッファを共有する場合も想定できる．

このように，各ノードのメディアバッファの合計が与えられた場合には以下のようにグラフを利用することで各メディアへのバッファの割り当てを決めてから W_s を算出することができる．ただし，ここで各メディア i $(1 \leq i \leq N_m)$ ごとに，[単位時間分のバッファ] 当たりに必要な [メモリ量としてのバッファ] の大きさを K_i とする．

1. 各メディア i について，W_s が与えられた際に各ノード j $(1 \leq j \leq N)$ に必要なバッファ量を求める式から W_s とバッファ量の関係 (式 (4.33)) を示すグラフを求める（4.2節と同様）．

$$buff = f_{ij}(W_s) \tag{4.33}$$

2. 各ノードの W_s とバッファ量の関係 (式 (4.34)) を表すグラフを求める．

$$Buff = g_j(W_s) = \sum_{i=0}^{N_m} K_i \times f_{ij}(W_s) \tag{4.34}$$

3. 求めたグラフと各ノードのバッファ量の合計 AB_j により各ノードでの R&L 同期判定，及び，必要な W_s の算出を行なう (図 4.14)．この W_s に対して求まる各メディアについてのバッファの大きさを各メディアに対するバッファの割り当てとする．

4. 4.4節と同様の方法により各メディアごとに割り当てられたバッファから各命令ごとの W_s を改めて算出する．

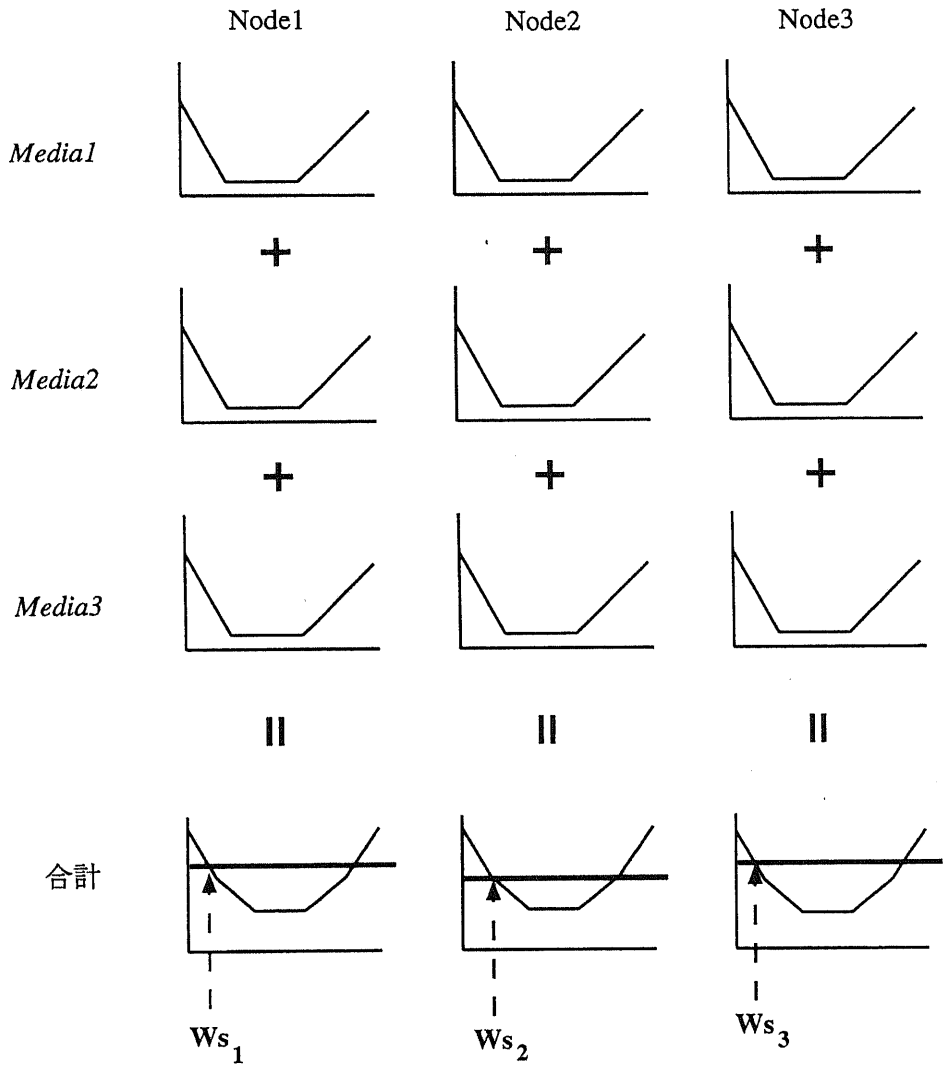


図 4.14: 各ノードでの R&L 同期判定，及び，必要な W_s の算出

4.7 システム構成に関する考察

本節ではメディア全体のバッファの大きさが与えられた際の R&L 同期充足判定と必要な待ち時間の算出を行なうための計算式を使って幾つかの具体的な数値例を示し，算出結果について考察を加える．

4.7.1 算出条件

算出条件は 4.3.1節の場合と同じ条件とする．ただし，source1，source2 に対する K_i はどちらも $1Mbytes/$秒 とする．

4.7.2 算出結果

算出結果を以下に示す．

まず，W_s と各ノードのバッファの大きさの関係を示す．

次に，各ノードのバッファ全体の大きさが与えられた際の R&L 同期判定及びシステム設計を行なう．

W_s と各ノードのバッファの大きさの関係

本節では W_s と各ノードのバッファの大きさの関係を示す．ただし，ここで示すグラフは全て横軸が W_s，縦軸が各ノードで必要なバッファの大きさとなっている．ここで，必要なバッファの大きさは T_{set}，$T_{ij}(1 \leq i,j \leq N)$ などがその取り得る値の範囲内でどのような値を取ったとしても必要十分な大きさとする．また，バッファの大きさはバイト単位で表す．

- 例 1

  図 4.15は情報源がハードディスクで $T_{dc} = 33$ミリ秒 の場合に sync_with_one のセマンティクスを満たすために各ノードで最低限必要なバッファの大きさを示している.

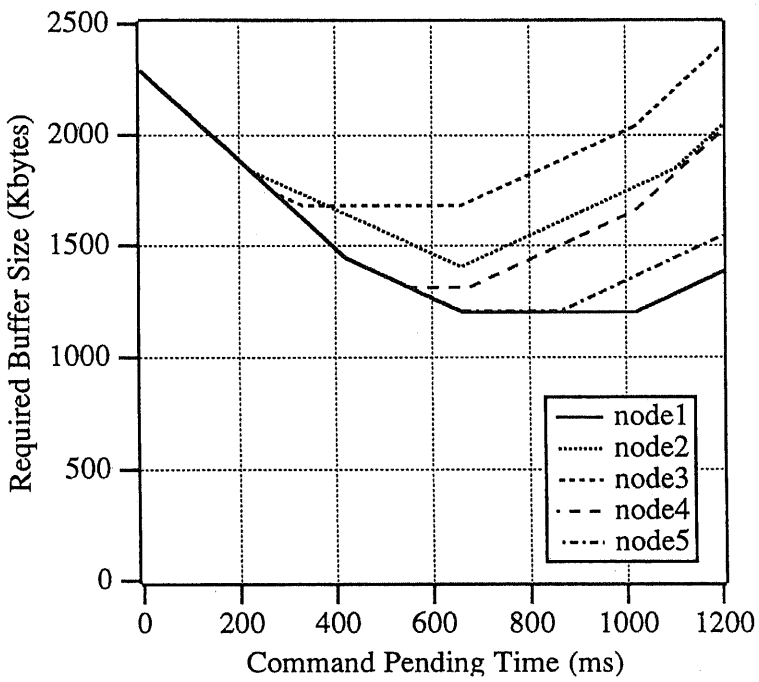


図 4.15: 必要なバッファ量（例 1）

図 4.15は以下の傾向を示す.

- W_s を増やすに従い必要なバッファの大きさは一旦は減少する. しかし, W_s をある程度以上増やすとかえって必要なバッファ量が増加する.

- 例 2

図 4.16は例 1 と同じ条件で sync_with_all のセマンティクスを満たすために場合に必要なバッファの大きさを示している.

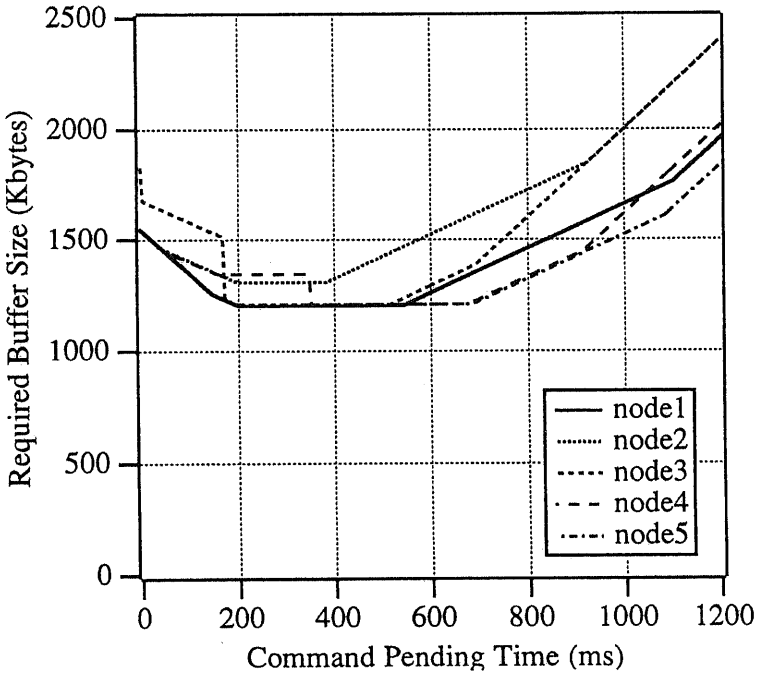


図 4.16: 必要なバッファ量（例 2）

図 4.16は以下の傾向を示す.

– ノード 3，4 などでは W_s が小さい領域で必要なバッファの大きさが不連続な値を取る.

R&L 同期判定とシステム設計

前節で求めたグラフを利用して各ノードのバッファ全体の大きさが与えられた際の同期判定とシステム設計を行なう.

例 1 の場合 sync_with_one の場合の同期判定とシステム設計を行なう.

- ノード 1 では 1201.2Kbytes 以上，ノード 2 では 1398.4Kbytes 以上，ノード 3 では 1674.0Kbytes 以上，ノード 4 では 1306.8Kbytes 以上，ノード 5 では

1201.2Kbytes 以上の大きさのメモリが必要となる．各ノードにそれ未満の大きさのメモリしか余っていなければ，sync_with_once のセマンティクスを満たすことはできない．

- 各ノードに 1800Kbytes ずつの free のメモリがある場合に各ノードで各メディアに必要なバッファの大きさを表 4.4に示す．

	node1	node2	node3	node4	node5
Media1 (from Source1)	643.2 Kbytes	678.0 Kbytes	643.2 Kbytes	643.2 Kbytes	643.2 Kbytes
Media2 (from Source2)	1122.0 Kbytes	1122.0 Kbytes	1122.0 Kbytes	1122.0 Kbytes	1122.0 Kbytes

表 4.4: 各メディアバッファの大きさ

また，この際の各命令に対する W_s，及び，各情報源からの送出待ち時間を表 4.5に示す．

命令 / Ws	stop ->play	stop ->skip	play ->stop
Ws of Media1 (送出待ち時間)	231ms (0ms)	0ms (0ms)	30ms (0ms)
Ws of Media2 (送出待ち時間)	219ms (12ms)	0ms (0ms)	0ms (30ms)
Ws	231ms	0ms	30ms

表 4.5: W_s

- 各ノードにそれぞれ [node1, node2, node3, node4, node5] = [1400Kbytes, 1500Kbytes, 1800Kbytes, 1700Kbytes, 1600Kbytes] の free のメモリがある場合に各ノードで各メディアに必要なバッファの大きさを表 4.6に示す.

	node1	node2	node3	node4	node5
Media1 (from Source1)	481.8 Kbytes	678.0 Kbytes	627.0 Kbytes	481.8 Kbytes	481.8 Kbytes
Media2 (from Source2)	822.0 Kbytes	822.0 Kbytes	1047.0 Kbytes	825.0 Kbytes	822.0 Kbytes

表 4.6: 各メディアバッファの大きさ

また, この際の各命令に対する W_s, 及び, 各情報源からの送出待ち時間を表 4.7に示す.

命令 / Ws	stop ->play	stop ->skip	play ->stop
Ws of Media1 (送出待ち時間)	392ms (127ms)	0ms (0ms)	30ms (0ms)
Ws of Media2 (送出待ち時間)	519ms (0ms)	0ms (0ms)	30ms (0ms)
Ws	519ms	0ms	30ms

表 4.7: W_s

例 2 の場合 sync_with_all の場合の同期判定とシステム設計を行なう.

- ノード 2 では 1301.4Kbytes 以上，それ以外のノードでは 1201.2Kbytes 以上の大きさのメモリが必要となる．各ノードにそれ未満の大きさのメモリしか余っていなければ，sync_with_all のセマンティクスを満たすことはできない.
- 各ノードに 1600Kbytes ずつの free のメモリがある場合に各ノードで各メディアに必要なバッファの大きさを表 4.8に示す.

	node1	node2	node3	node4	node5
Media1 (from Source1)	551.2 Kbytes	582.0 Kbytes	759.0 Kbytes	579.0 Kbytes	551.2 Kbytes
Media2 (from Source2)	841.0 Kbytes	841.0 Kbytes	841.0 Kbytes	841.0 Kbytes	841.0 Kbytes

表 4.8: 各メディアバッファの大きさ

また，この際の各命令に対する W_s，及び，各情報源からの送出待ち時間を表 4.9に示す.

命令 / Ws	stop ->play	stop ->skip	play ->stop
Ws of Media1 (送出待ち時間)	47ms (0ms)	0ms (0ms)	354ms (0ms)
Ws of Media2 (送出待ち時間)	35ms (12ms)	0ms (0ms)	6ms (348ms)
Ws	47ms	0ms	354ms

表 4.9: W_s

- 各ノードに 1400Kbytes ずつの free のメモリがある場合に各ノードで各メディアに必要なバッファの大きさを表 4.10に示す.

	node1	node2	node3	node4	node5
Media1 (from Source1)	481.8 Kbytes	582.0 Kbytes	481.8 Kbytes	579.0 Kbytes	481.8 Kbytes
Media2 (from Source2)	743.7 Kbytes	743.7 Kbytes	743.7 Kbytes	759.0 Kbytes	743.7 Kbytes

表 4.10: 各メディアバッファの大きさ

また，この際の各命令に対する W_s，及び，各情報源からの送出待ち時間を表 4.11に示す.

命令 / Ws	stop ->play	stop ->skip	play ->stop
Ws of Media1 (送出待ち時間)	47ms (0ms)	0ms (0ms)	354ms (0ms)
Ws of Media2 (送出待ち時間)	35ms (12ms)	0ms (0ms)	6ms (348ms)
Ws	47ms	0ms	354ms

表 4.11: W_s

4.7.3 考察

図 4.15，図 4.16で示したグラフは 4.3節で示したグラフと同様に，以下のような傾向を示す．

- W_s をある程度増やすことにより必要なバッファの大きさを削減できる．しかし，W_s をある程度以上増やすとかえって必要なバッファ量が増加する．
- W_r も含めて考えると，操作者ノード (node1) に近いノードほど必要なバッファ量が小さく，情報源ノード (node2,node3) に近いノードほど必要なバッファ量は大きい．

また，R&L 同期判定とシステム設計を行なった結果から以下のようなことがわかる．

- 各ノードにおいて最低限必要なバッファの大きさ以上のメモリが余っていなければ，R&L 同期を満たすことはできない．
- 最低 1 つのノードが与えられたメモリを使い切るが，それ以外のノードでは与えられたメモリ全ては必要ない．ただし，sync_with_all の場合には必要なバッファの大きさが不連続な値を取るため，表 4.10の場合のように全てのノードで与えられたメモリを使い切らないといったことも起こり得る．

本節では，操作権を持つノードを固定して必要な R&L 同期判定，及び，W_s の算出をおこなってきた．操作権が複数のノード間で受け渡される場合には，操作権が操作者になりうる全てのノードにある場合について R&L 同期判定を行ない，全ての場合に R&L 同期が満たされることを確かめる必要がある．また，R&L 同期が満たされる場合にも，バッファの分割の仕方によっては，あるノードにおいてそれぞれのメディアごとに必要なバッファの大きさの最大値の合計がそのノードのバッファの大きさを上回ってしまう可能性もある．このような場合には，そのノードでのあるメディアバッファの大きさを最大値としている操作位置での W_s を調整することでそのメディアバッファの大きさを小さくする必要がある．R&L 同期も満たされ，各ノードでのバッファの分割の調整も完了した場合，さらに操作権の位置ごとに W_s の算出，情報源からの送出待ち時間の算出などを行なう必要がある．

また，MPEG[Yasu 91] のようなフレーム間圧縮を用いる場合，future 方向には次の Intra フレーム又は Predicted フレームまでのフレーム情報を，past 方向には次の Intra フ

レームまでのフレーム情報を余分に持っている必要があるため，そのフレーム情報分の大きさ（数フレーム分）を与えられたバッファの大きさから予め差し引いて R&L 同期判定，及び，W_s の算出を行なう必要がある．MPEG を用いた RMS が複数ある場合には，その分の（2 倍，3 倍した）大きさを各ノードのバッファの大きさから差し引いてやる必要がある．

第 5 章

LAN 環境における負荷変動吸収機構の試作と評価

本章ではLAN環境において試作したシステムにおけるメディア間同期機構，及び，その評価を示す．本同期機構はR&L同期を可能とすると共に，負荷変動に柔軟に対処しつつ情報提示の際のメディア品質を維持する仕組みも備える．

5.1 試作システムにおける同期方式

5.1.1 試作システムの概要

本節では Ethernet(10Mbps) で結ばれた複数台の UNIX ワークステーション上に実装した試作システムの概要について述べる.

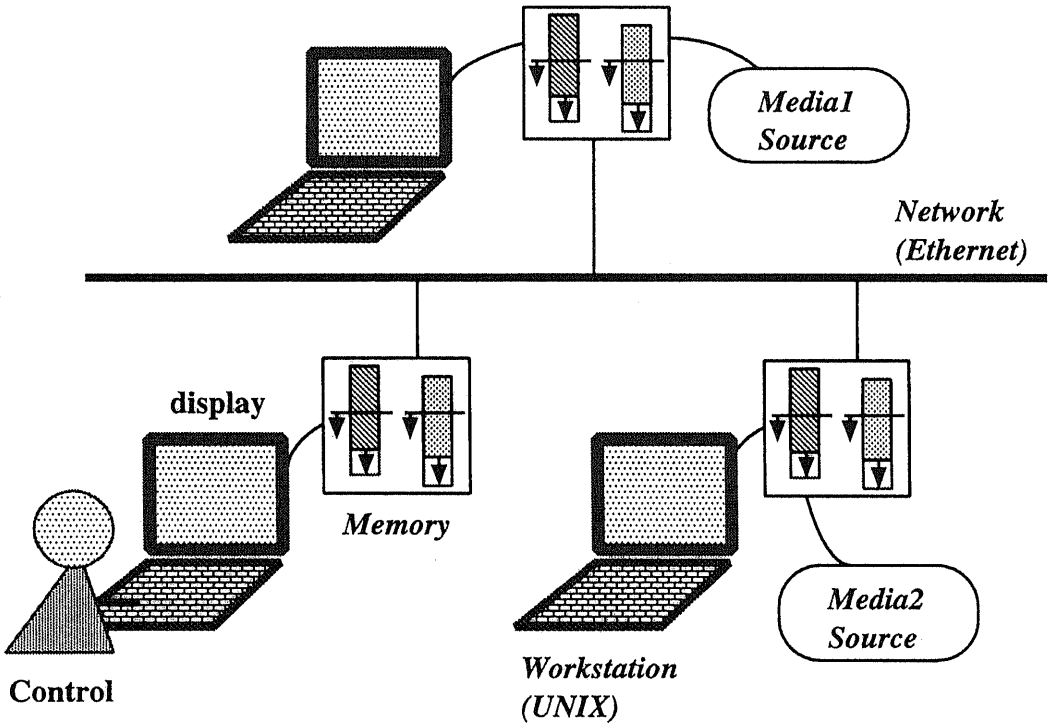


図 5.1: 試作システム

試作システムではシステム中に一つ存在し, 初期化などの際に用いられる管理プロセスと各参加ノードごとに起動される動作プロセスの 2 種類のプロセスを用いる. また, これらのプロセス間に各メディア情報及び制御情報ごとに connection を張ることで情報をやりとりする. 情報送出, 表示などは各動作プロセス中で定期的にスケジュールされる procedure を用いて行なう.

情報源としてはハードディスク中に格納された動画像 (1 フレーム 21,600 bytes), テキスト情報 (1 つにつき 20 bytes), 及び音声 (8000 bytes/sec) の 3 つを用い, 操作命令としては再生, 一時停止, 逆再生, 早送り, 巻き戻しなどの命令をサポートしている.

LMS としては音声情報のみを用いる. 操作者ノードを含む複数のノードが発信者ノー

ドとなり，発信者ノードから各ノードに音声情報が送られる．

発信者ノードでは，話したい時のみ（talk ボタンを押すことで）音声デバイスを open する（つまり，read 用の音声 queue に情報を取り込む．）．

5.1.2 試作システムにおける R&L 同期

試作システムでは QOS を指定することでネットワーク，端末などの資源を予め確保することができないため，ネットワークの伝送遅延を予測できず，各端末における他のプロセスの影響も排除できない．

このような環境では，第 4章におけるような厳密な R&L 同期を取ることはできない．そこで，試作システムにおいてはある程度の同期ずれを許容しながらもできるだけ 3.4.2節で述べた R&L 同期のセマンティクスに近付けることを目標とする．

また，試作システムではネットワークの伝送遅延を予測できないため各ノードではバッファ中に提示できるメディア情報が無くなった時点でメディア情報の到着を待つ．

ただし，実装においても「全ての参加者ノードで少なくとも操作者ノードで提示されるメディア情報と同じだけのメディア情報が提示される．」という条件は満たされるものとする．

5.1.3 同期方式の概要

本節では試作システムにおける同期方式について述べる．

本システムのように QOS を指定することで資源を確保することができないネットワークと端末により構成されるシステムの場合，他のプロセスによる影響を排除できない．従って，このような環境でメディア間の同期を取るには各ノードでの負荷変動，遅延時間変動などを吸収するための何らかの機構が必要となる．

このような環境におけるメディア間同期の手法 [Fuji 93, Ander 91] としては，基準となる時間軸に全てのメディアの進行を合わせる方法，基準となるメディアの進行に他のメディアの進行を合わせる方法などが考えられる．

R&L 同期を実現するためにはノード間での再生点の位置関係を合わせる必要があるが，時間軸に合わせる方法や基準となるメディアに合わせる方法では再生点を意図的に動かすことができず，負荷変動，遅延時間変動などに応じて変化するバッファ上で適切な再生点を取りながら情報提示を行なうができない．

そこで，本システムでは各ノードのバッファ中に理想的な再生点を定め，この理想的

な再生点に各メディアの再生点を近付けることで各ノードにおけるメディア間の同期を取る．また，この理想的な再生点を，R&L 同期のセマンティクスを満たすようにノード間で調整することで R&L 同期を実現する (図 5.2).

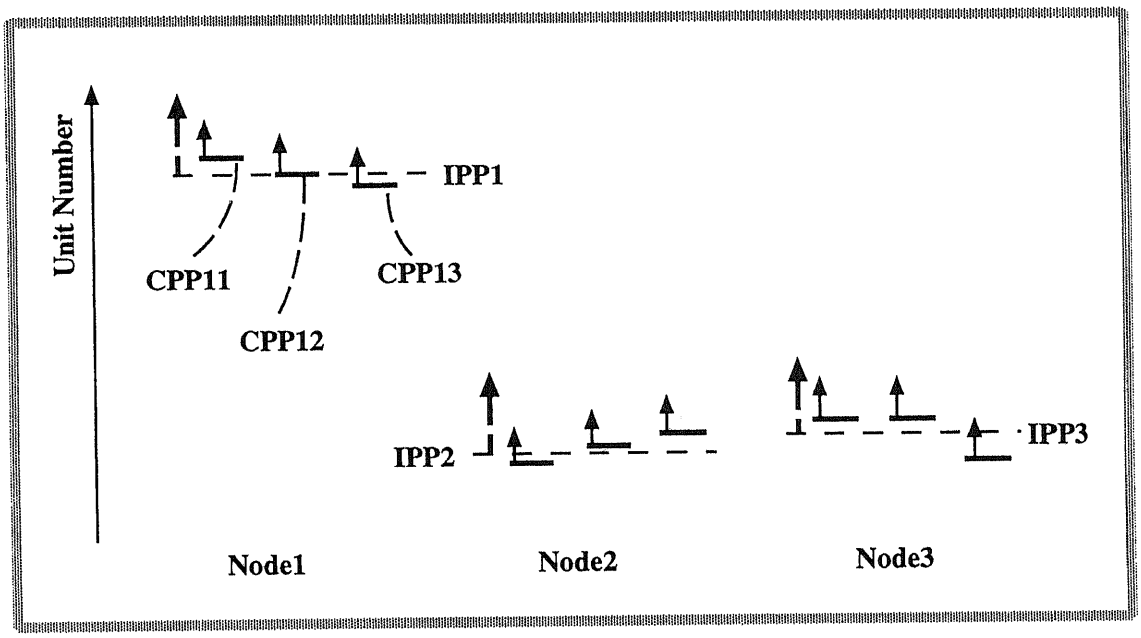


図 5.2: A media synchronization mechanism

本同期手法では再生間隔の基準となる時間 (T_{int}) は一定とする．そして，情報提示の際には各メディアの再生間隔ごとに適切なフレームを選択，提示することでバッファ上の再生位置を意図的に変化させる．

また，本システムでは情報源には予め同期付けられた情報が格納されていることを想定しているため，Acme[Ander 91] のように各フレームが送出される際に基準となるメディアに合わせてフレーム番号を付加する方法では予め意図していたような同期が取れない可能性がある．

そこで，本システムでは，各メディアフレームの送出を基準となる時間軸に合わせることとする．

以上述べてきたように，本システムでは各情報源からのメディア送出は基準となる時間 (T_{int}) に同期させることで，各情報提示ノードでの情報提示は理想的な再生点に合わせることでメディア間同期を維持する．そして，この理想的な再生点をノード間で調整することで R&L 同期を実現する．

以下，第5.2節では，試作中のシステムにおいて付加変動を吸収しながらR&R同期を維持する手法について述べる．第5.3節でR&R同期方式の評価について述べる．第5.4節では，R&L同期を実現するための手法について述べる．最後に第5.5節でR&L同期方式の性能評価について述べる．

5.2 R&R 同期方式

本節では試作システムにおける R&R 同期方式について述べる.

5.2.1 情報源ノードでの調整

video,text の場合

video,text の場合, 情報送出を行なわない情報送出区間を設けることでメディア送出を基準となる時間軸に合わせる.

1. n_{skip} の算出

   式 (5.1) により n_{skip} を算出する. ここで, T_{int} は情報送出 procedure のスケジュール間隔, T_{sch} は情報送出 procedure が実際にスケジュールされるのにかかった時間であり, 前回送出したフレームより n_{skip}+1 フレーム先のフレームを送出することで情報送出を基準となる時間軸に合わせることができる.

$$n_{skip} = [(T_{sch} - T_{int}/2)/T_{int}] \tag{5.1}$$

   また, n_{skip} が大きいことは負荷が大きいことを示しており, 負荷を下げるために何らかの工夫をすることが望ましい. 本実装では n_{skip} + 1 先のフレームを送出した後, 引き続く $[n_{skip}/2]$ 回のスケジューリングの際には情報送出を行なわない.

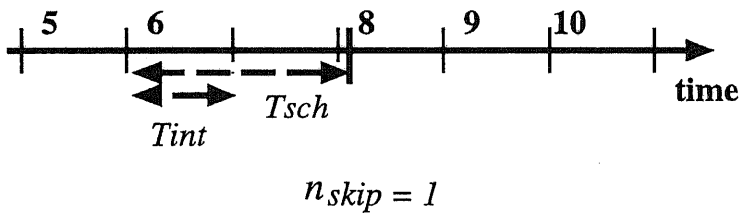


図 5.3: 情報送出の一例

図 5.3に情報送出の一例を示す. 図 5.3において 5 unit 目, 6 unit 目の media unit を送出した後, $T_{sch} = 70ms$ 後 ($T_{int} = 33ms$) に情報送出 procedure がスケジュールされたとすると, $n_{skip} = 1$ となる. 従って, 7 unit 目の media unit が skip され, 8 unit 目の media unit が送出される.

情報源から各ノードに送出するデータパケットのデータへの付加情報は unit 番号と l_skip, r_skip(それぞれ $n_{skip}, [n_{skip}/2]$ のいずれかに対応) となる.

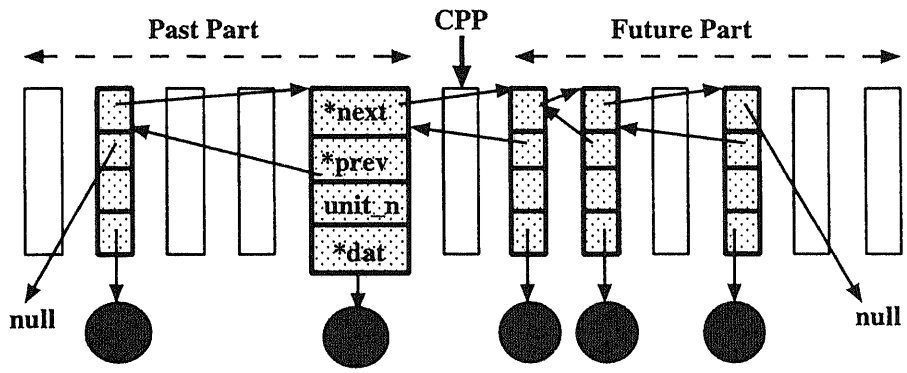


図 5.4: メディアバッファ

各情報提示ノードでは送られてきたデータを unit 番号 u_num と l_skip, r_skip を参考にして各ノードのメディアバッファに格納する. メディアバッファの構造を図 5.4に示す. 図 5.4において *next は次の実フレームへのポインタ, *prev は前の実フレームへのポインタを示している. また, CPP は現在の再生点であり, バッファ上において CPP よりも unit 番号が大きい部分が Future Part, CPP よりも unit 番号が小さい部分が Past Part となる. *dat は実データを指すポインタである. 実データへのポインタが無いフレーム (dummy フレーム) は情報送出ノードにおいて skip されたフレームである. また, audio に関しては図 5.4で示した情報以外に有音, 無音を示す付加情報が各フレームごとに存在する.

2. n_{skip} の平滑化

一時的な負荷の増大により n_{skip} が大きな値を取ることは情報提示の際の連続性を損なうため, 望ましくない. 本実装では情報送出の実時間からのずれを許し, n_{skip} の上限 U_SKIP と下限 L_SKIP を適応的に変化させることで n_{skip} の平滑化を図りつつ負荷変動に対応する (図 5.5).

以下に, 平滑化の手法を示す.

- $n_{skip} > U\_SKIP$ の場合, n_{skip} を U_SKIP に設定し実時間からの情報送出の遅れを示す変数 $delay$ に $(n_{skip} - U\_SKIP)$ を加える.

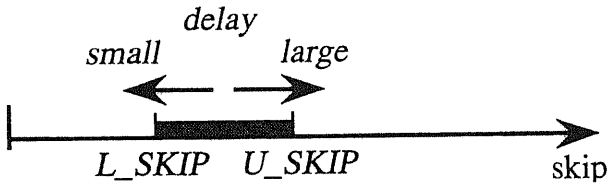


図 5.5: Normalization of n_{skip}

- $n_{skip} < L\_SKIP$ で，$delay > 0$ の場合，n_{skip} を L_SKIP に設定し，$delay$ から $(L\_SKIP - n_{skip})$ を引く．
- $delay$ の大きさに応じて U_SKIP，L_SKIP を調整する．

3. 高速再生，巻き戻しなどへの対応

   高速再生などでは，通常再生よりも N 倍の速度で情報を送出する必要がある．本システムでは，n_{skip} を N 倍し，さらに $N-1$ だけ増やすことで送出速度を N 倍にしている．

audio の場合

audio の場合，情報送出を行なわない情報送出区間を設けることは望ましくない．そこで，各情報送出区間において複数のメディアフレームを送れることとし，基本的に $skip$ が起こらないようにしている．

また，無音区間のメディア送出を行なわないことで送出情報量を減らすようにしている．

1. M 倍送出

   audio の 1 フレーム分の情報量は video などと比べてかなり小さいため，各情報送出区間において最大 M フレーム分の情報送出を許すこととする．つまり，式 (5.1) において n_{skip} の値が 0 にならなかった場合にもその分のメディアフレームを送出することで skip が起こらないようにできる．この M の値を早送りの際などの倍速パラメータ N よりも大きく取っておけば，早送りの際などにもすべてのメディア情報を送出できるため質の良い音声を提示できる．

2. 無音区間の利用

音声情報源では，無音部を送出する必要がないため，有音部のみ送出することで送出情報量を減らすことができる．

ただし，無音部において全く情報を送らないと各情報提示ノードでは次の有音部のフレームが送られるまでその無音部分が無音なのか，単にまだ送られてこないだけなのか判断できないため，無音部においても（データの入っていない）フレーム番号のみからなる無音パケットを送ることとしている．

また，有音部と無音部を両方含むパケットも存在し，このようなパケットには有音部のフレーム番号以外に無音部の長さなどの情報も付加する．

また，音声情報を格納するメディアバッファには図 5.4の情報以外に，各フレームが有音か無音かを示す情報が必要となる．

以上述べてきた方法により，情報源からの情報送出はほぼ基準となる時間軸に追随する．

5.2.2 情報提示ノードでの調整

video,text の場合

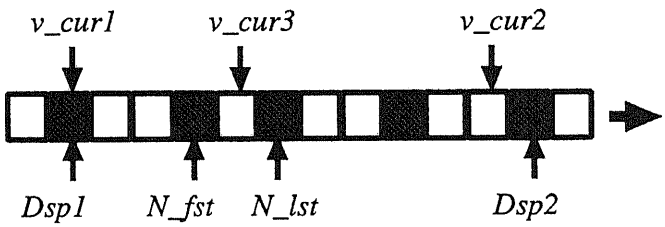
式 (5.1) と同様の式により，情報提示 procedure がスケジュールされる度に n_{skip} を求める．n_{skip} が 0 より大きい場合，$(n_{skip}+1)$ 回再生点の更新を行なったフレームを提示する．また，引き続く $[n_{skip}/2]$ 回の情報提示 procedure スケジュールの際には情報の提示は行わない．(つまり，$[n_{skip}/2]$ 回同じ情報が提示され続ける．ただし，各スケジュールの際に再生点の更新は行う．)

そして，再生点の更新の度に，理想的な再生点との位置関係により *skip* または *pause* を行なうことで理想的な再生点に追随する．

また，メディアバッファ中に *v_current* のフレームが存在しない場合，メディアバッファ中のどの実フレームを選択するかの方針を以下に示す．

1. *v_current* に最も近い実フレーム *nearest_f* を選択．
2. *v_current* が 2 つの実フレーム *n_first*,*n_last* のちょうど真ん中にある場合，再生中とすると $[v\_prev < n\_first]$ ならば *n_first* を $[v\_prev \geq n\_first]$ ならば *n_last* を選択．（逆再生中も同様．）

このような方法を用いることで情報提示ノードでは最適な unit を負荷に応じて提示できる.



In the case of v_cur3

$$v\_prev < N\_fst \rightarrow Dsp3 = N\_fst$$

$$v\_prev \geqq N\_lst \rightarrow Dsp3 = N\_lst$$

図 5.6: 実際に提示するフレーム

図 5.6において, *v_cur* が *v_cur*1 の位置にある場合 *Dsp*1 が, *v_cur*2 の位置にある場合最も近い位置にある実フレームである *Dsp*2 が提示される. また, *v_cur* が *v_cur*3 の位置にある場合, 前回提示されたフレーム *v_prev* により *N_fst*, *N_lst* のどちらかのフレームが提示されることとなる.

また N 倍速に関しては, n_{skip} 及び再生点の更新位置を N 倍先にすることで実現している.

audio の場合

音声の場合, video や text で用いた方法では *skip* の度に音が途切れてしまう.

そこで, 音声の場合には, 以下のようにして, 音声デバイスの queue を常にある程度満たしておき, *skip* の際にも queue が無くなって音が途切れることがないようにする.

- 音声提示 procedure のスケジュールの度に queue 中の音声データの大きさを示す変数 *v_queue* をスケジュール間隔に比例した大きさだけ減らす. そして, *v_queue* の大きさがある固定長 *s_queue* になるまで前回のスケジュールの際に最後に挿入したフレーム *lv_q* の次のフレームから順次 queue を満たしていく.

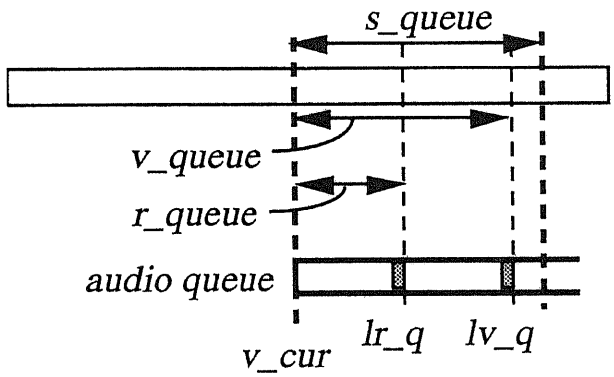


図 5.7: audio queue

- *lv_q* と *v_queue* により現在提示されているフレームを予測し *v_current* とする．この *v_current* と理想的な再生点との位置関係により *skip*, *pause* を行なうかどうか決める．ただし，*skip*, *pause* は即座には行なわず，*lv_q* 以降 *SEARCH* フレーム分のフレームについて無音部を検査し，できるだけ無音部において（有音部において行なわれているはずの）*skip*,*pause* を行なうようにする．つまり，音楽などのように有音部が続く場合以外は，無音部において *skip*,*pause* を行なうことで有音部の音声品質を同期よりも優先させる．

  skip,*pause* は *lv_q* の次またはそれ以降に挿入されるフレームに適用されるため，*s_queue* が大きいほど理想的な再生点への収束は遅れる．（付加変動による音の途切れを防ぐため，*s_queue* はある程度以上小さくできない．）

- 音声データは 1 秒当たり 8000 バイトずつ再生されることになっているが，その再生速度のわずかなずれも，長時間再生を続ける場合，かなり大きな同期ずれを引き起こす可能性がある．そこで本システムでは，無音区間が続く場合実際には queue にデータを挿入せず，有音部が挿入される際にすでに挿入されているはずの無音部のデータを挿入してから有音部のデータを挿入することとし，無音部で queue が空になった際に同期をとり直すことができるようにしている．

- 本システムでは逆再生，高速逆再生の場合には，音声は提示しない．高速再生の場合には，queue に挿入するフレームを N ごとにすることで N 倍速を実現する．また，早送り，巻き戻しの際には音声は通常の速度で再生方向に提示される．これは，

N 倍速で動く仮想的な再生点と，実際の再生点の位置関係により，実際の再生点を次々と有音部の開始位置に飛ばしていくことで実現している (図 5.8).

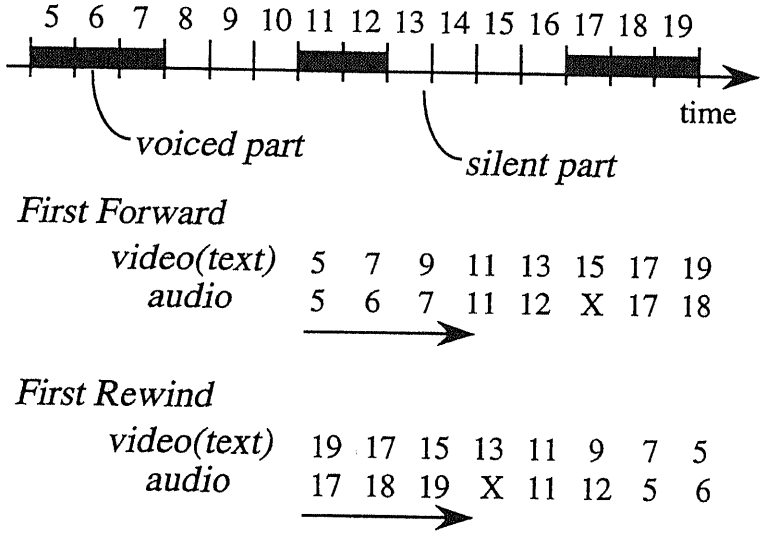


図 5.8: First Forward and First Rewind

図 5.8は N が 2 の場合の早送り，巻き戻し時に提示されるフレームを示している．早送りの場合，video(text) が 5,7,9 フレーム目を提示している際には N 倍速で動く仮想的な再生点はまだ次の有音部に達していないため，audio に関しては通常の再生速度で 5,6,7 フレーム目が提示されている．11 フレーム目において N 倍速で動く仮想的な再生点が次の有音部に到達し，audio の実際の再生点は 11 フレーム目に跳ぶ．図から明らかなように，video(text) が 15 フレーム目を提示している時を除いて audio はその有音部中のフレームを提示しており，図 5.8の場合には 19 フレーム目以外の全ての有音部を提示している．

巻き戻しの場合，video(text) が 19,17,15 フレーム目を提示している際には N 倍速で動く再生点は次の有音部に達していないため，audio に関しては通常の再生速度で順方向に 17,18,19 フレーム目を提示している．有音部を提示する方向以外は，巻き戻しに関しても早送りの場合と同様のことが言える．

5.2.3 情報提示ノードでの再生点のずれへの対応

本システムでは，”バッファ中の理想的な位置に各メディアの再生点を近付ける”ことで各ノードにおけるメディア間の同期を取る．

理想的な再生点

理想的な再生点はの決め方については，5.4節で述べる．

再生点の調整

実際の再生点 (F_{real}) と理想的な再生点 (F_{ideal}) の位置関係 (図 5.9) に応じて *skip*,*pause* を行なうことで F_{real} を F_{ideal} に近付ける．
本システムでは F_{real} が F_{ideal} から遠いほど *skip* や *pause* の頻度を大きくしている．

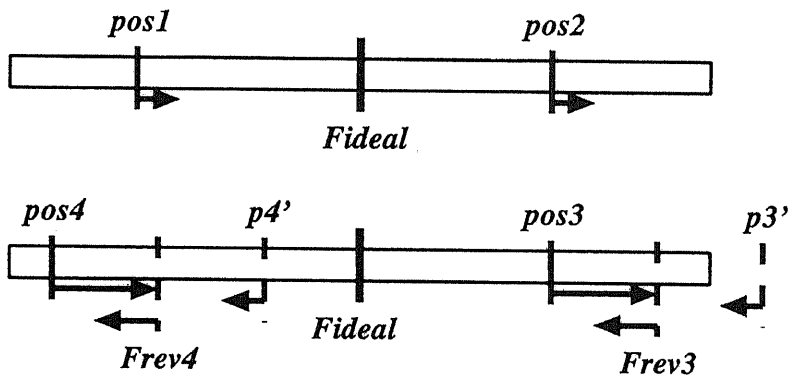


図 5.9: 再生点の位置関係

- F_{real} が *pos*1 のような F_{ideal} よりも後の位置にある場合，*skip* により F_{ideal} に近づく．
- F_{real} が *pos*2 のような F_{ideal} よりも先の位置にある場合，*pause* により F_{ideal} に近づく．

また実装では $distance = |F_{real} - F_{ideal}|$，*skip* や *pause* を行うと 0 になり，行わない間 increment され続けるカウンタを *n_count* としたときに以下の式を満たした時に *skip* もしくは *pause* を行うものとした．

$$n\_count > Const/distance \quad (Const = 10.0) \tag{5.2}$$

つまり，F_{real} が F_{ideal} から遠いほど *skip* や *pause* の頻度が大きくなる．

図 5.9の $pos3$ や $pos4$ のように，逆再生開始点 F_{rev} まできたときに反転する命令が reserve されている場合，式 (5.2) をそのまま適用することはできない．

- F_{real} が $pos3$ にあることは F_{real} が $p3'$ において逆再生されていることと等価である．$pos4$ についても同様．従って，$v\_dist = F_{real} - F_{ideal} + 2 \times (F_{rev} - F_{real})$ とした時，v_dist が正ならば $skip$，負ならば $pause$ によって F_{ideal} に近づく．

又，この際 $distance = |v\_dist|$ として式 (5.2) を適用できる．

5.3 R&R 同期方式の評価

本節では，試作システムにおける R&R 同期方式の評価について述べる．

5.3.1 評価環境

Ethernet で接続された 3 台の UNIX workstation を用いて R&R 同期機構の評価を行なった．3 台のマシーンの役割を以下に示す．

SS10 (Sparc station 10) RMS 情報源 (video,text)，情報提示の 2 つの役割を兼ねる．

IPX (Sparc station IPX) 情報源 (audio)，操作者，情報提示の 3 つの役割を兼ねる．

SS1 (Sparc station 1) 情報提示のみを行なう．

バッファの大きさは video， text に対しては 20 フレーム分， audio に対しては 20+20 フレーム分とし，$T_{int} = 200ms$ とした．初期状態としては各メディアバッファが 0 フレーム目からのメディア情報で半分だけ満たされている状態とした．

また，本測定において， 5.4節で述べる R&L 同期機構は sync_with_all のセマンティクスを満たすようにした．

5.3.2 評価結果

IPX から再生命令を発行した際に，各ノードで各フレームが提示された時間を以下の 2 つの場合について測定した．

No-Load 他のプロセスが一切走っていない場合

Load SS1 において 100 フレーム目から 150 フレーム目のメディア情報が提示されている際に負荷を加えた場合．

No-Load の場合

No-Load の場合の SS1 での測定結果を図 5.10に示す.

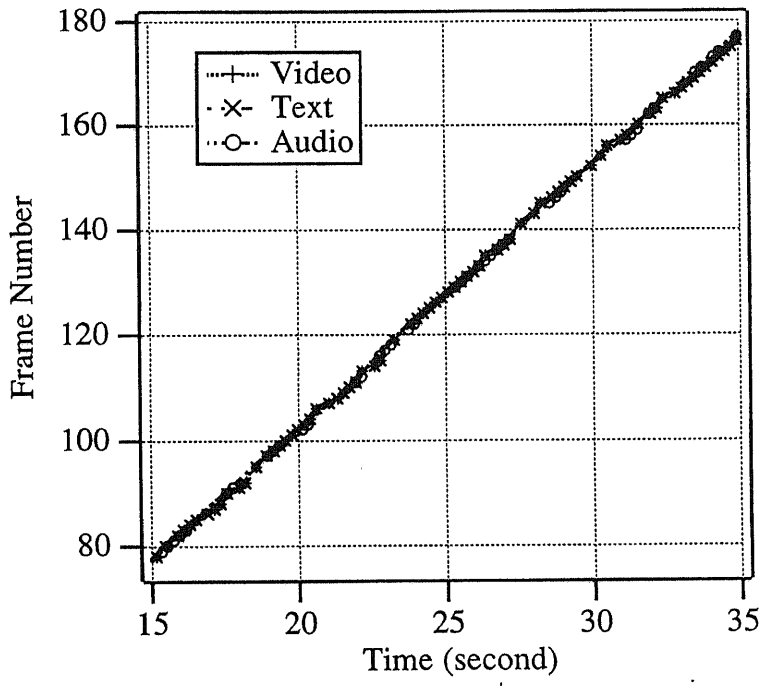


図 5.10: Presentation time at SS1

図 5.10は以下の傾向を示す.

- 3 つのメディアはよく同期して提示されている.
- audio の情報提示は skip されていないが, video, text に関しては若干跳ばされている.

No-Load の場合の IPX での測定結果を図 5.11に示す.

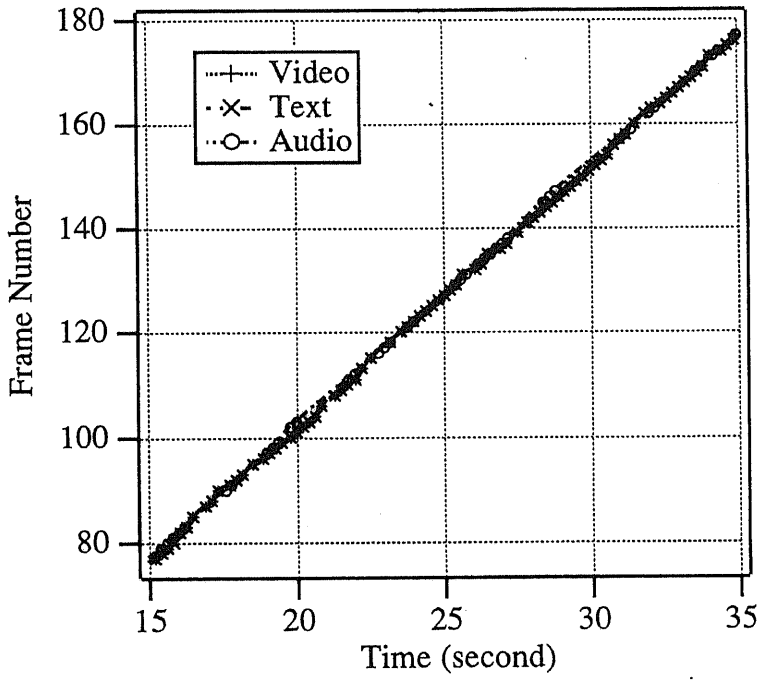


図 5.11: Presentation time at IPX

図 5.11は以下の傾向を示す.

- 3 つのメディアはよく同期して提示されている.
- 3 つのメディアともほとんど跳ばされていない.

No-Load の場合の SS10 での測定結果を図 5.12に示す.

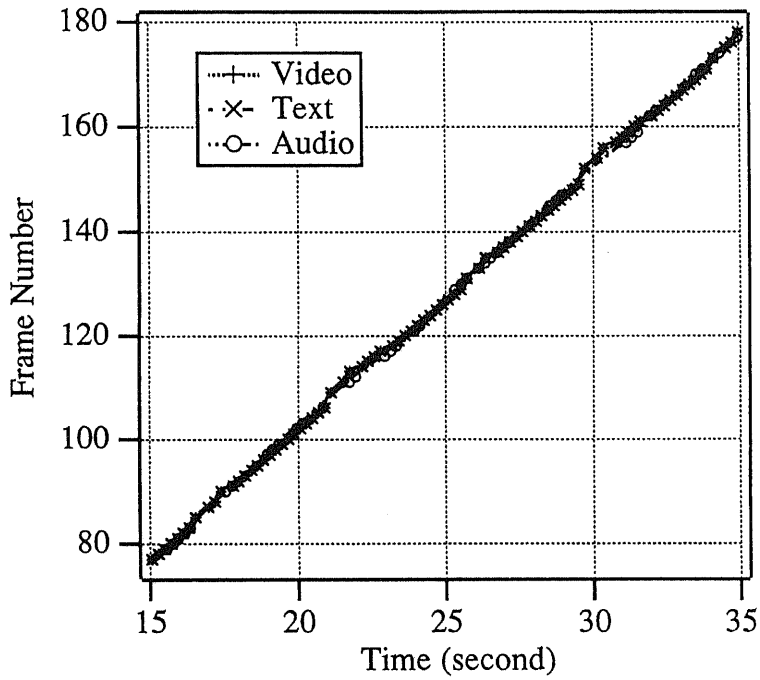


図 5.12: Presentation time at SS10

図 5.12は以下の傾向を示す.

- 3 つのメディアはよく同期して提示されている.
- 3 つのメディアともほとんど跳ばされていない.

Load の場合

Load の場合の SS1 での測定結果を図 5.13に示す.

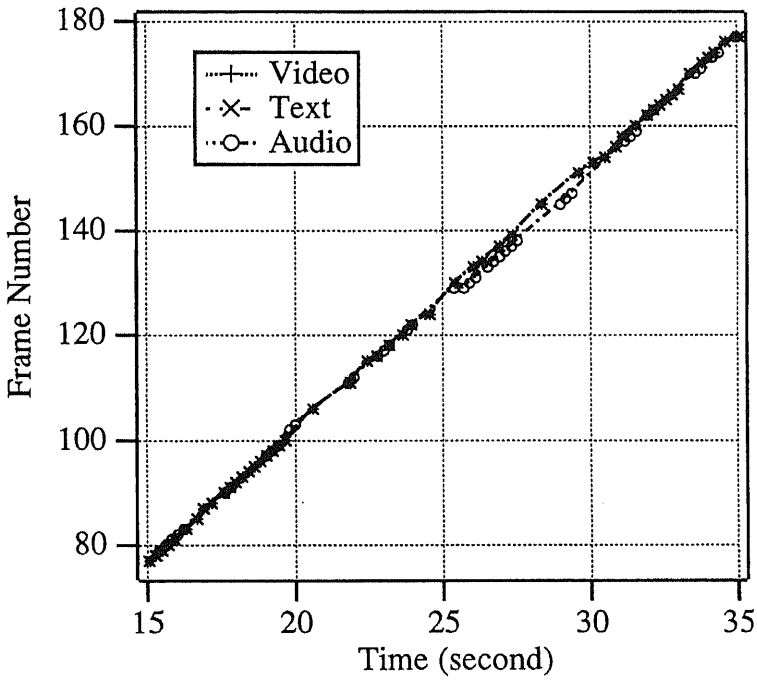


図 5.13: Presentation time at SS1 (variable load)

図 5.13は以下の傾向を示す.

- 3つのメディアは各ノードでほぼ同期して提示されている.
- 負荷が加えられている間 (100 フレーム目から 150 フレーム目のメディア情報が提示されている間) は video, text の情報提示がかなり skip されている. 音声の提示は skip されていない.

Load の場合の IPX での測定結果を図 5.14に示す.

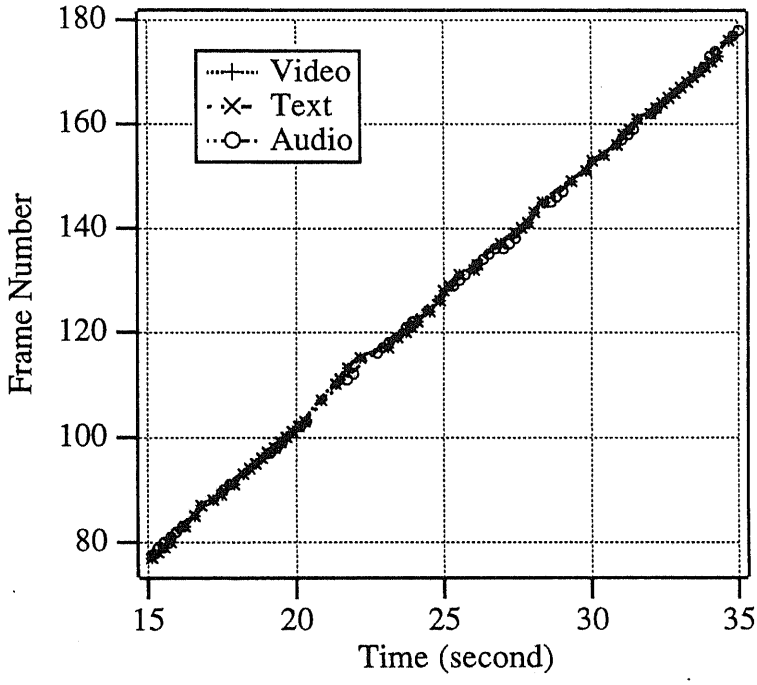


図 5.14: Presentation time at IPX (variable load)

図 5.14は以下の傾向を示す.

- 3つのメディアは各ノードで同期して提示されている.
- 負荷が加えられている間 (100 フレーム目から 150 フレーム目のメディア情報が提示されている間) も (SS1 に比べて) それほど skip されていない.

Load の場合の SS10 での測定結果を図 5.15に示す.

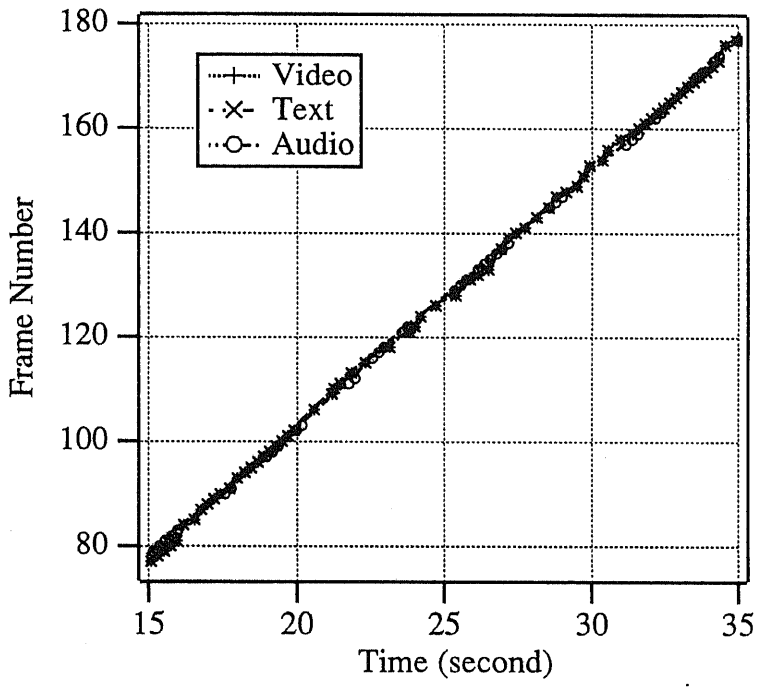


図 5.15: Presentation time at SS10 (variable load)

図 5.14は以下の傾向を示す.

- 3 つのメディアは各ノードでよく同期して提示されている.
- 負荷が加えられている間 (100 フレーム目から 150 フレーム目のメディア情報が提示されている間) も (SS1 に比べて) それほど skip されていない.

5.3.3 考察

図 5.10, 図 5.11, 図 5.12から以下のことがわかる.

- No-Load の場合には 3 つのメディアはよく同期して提示されている.
- audio の情報提示は skip されていない.
- video, text に関しては若干跳ばされている. SS1, IPX, SS10 の順で跳ばされた unit 数が多い. つまり, CPU の余裕度に応じて提示された unit 数が多い.

また, 図 5.13, 図 5.14, 図 5.15 から以下のことがわかる.

- 3 つのメディアは各ノードでほぼ同期して提示されている.
- 負荷が加えられている間 (100 フレーム目から 150 フレーム目のメディア情報が提示されている間) は video, text の情報提示がかなり skip されている. 音声の提示は skip されていない.
- 負荷が加えられている間, SS1 ではかなり情報提示が skip され, 提示されるメディアの品質も悪くなっているが, IPX, SS10 ではそれほど skip は起こっていない. つまり, SS1 に加えられた負荷は他のノードにはそれほど影響を及ぼしていない.

今後, ネットワーク, 端末などにおいても QOS を申告することで資源を確保できるようになると考えられるが, そのような環境においても本節で述べた同期機構を用いることで QOS の申告に柔軟性を持たせることができ (必要な CPU 資源の上限と下限を申告するなど.), システム中の資源を有効に活用することができる.

5.4 R&L 同期方式

本節では試作システムにおける R&L 同期方式について述べる.

5.4.1 ノード間での再生点調整

R&L 同期の実現のためにはノード間での再生点の調整が必要となる.

例えば, 3.4.2節で述べた sync_with_one を実現するためには図 3.5のように操作者ノードの再生点が他ノードの再生点よりも幾らか進んでいる必要がある.

また, sync_with_multi や sync_with_all では幾つかのノードの再生点は操作者の再生点と一致している必要がある.

R&L 同期のいずれかのセマンティクスを満たすためには, システム中の各ノードの再生点は以下に示す 2 つのタイプのどちらかに属さなければならない.

s_type グローバルな時間軸上で再生点が操作者ノードの再生点と完全に一致するノード.

d_type グローバルな時間軸上で再生点が操作者ノードの再生点よりも幾分遅れるノード.

ここで, 各ノードの時計のずれ程度の同期ずれは許容するものとし, グローバルな時間軸を各ノードの時間とする.

また, 操作命令への付加情報として, 命令開始位置以外にその命令が出された時刻も付加するものとする.

操作命令に対する提示開始タイミング

一時停止状態で操作者ノードから再生開始命令が発せられた場合, (操作者ノードを含む) 各ノードでは以下のように対応する (5.16).

s_type のノード [再生命令が操作者ノードで発せられた時間] +[一定の待ち時間 (Tw)] 後に再生を開始する. 再生開始命令受信時に既にこの時間を過ぎていた場合, 即座に再生を開始する.

d_type のノード [再生命令を受け取った時間] +[一定の待ち時間 (Tw)]+[audio queue が吐き出されるのにかかる時間 (Tq)] 後に再生を開始する.

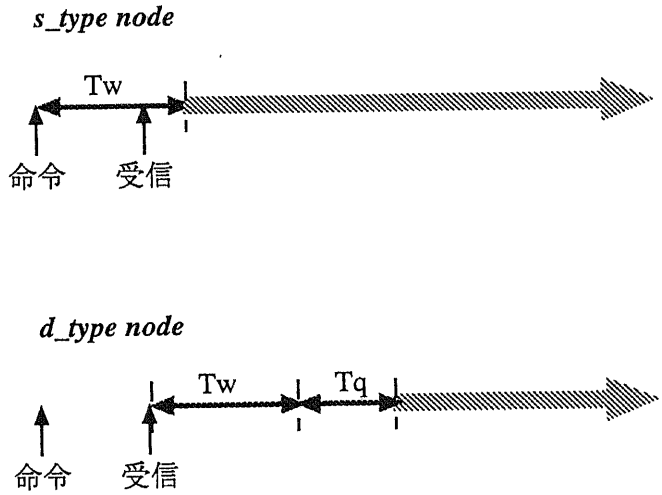


図 5.16: 情報提示開始タイミング

ここで，Tw は sync_with_one の場合 0，それ以外の場合命令の伝送遅延時間程度の値とする．

また，5.2節で述べたように負荷変動の影響を避けるため audio queue は一定に保たれる必要がある．従って，d_type のノードでは Tq だけ余分に再生点を遅らせる必要がある．

一時停止命令に関してはこのような待ち時間を考える必要はないが，それ以外の命令に関しては同様にして各ノードでの情報提示開始タイミングを調整することで R&L 同期が満たされるようにする．

sync_with_multi の場合の情報提示開始タイミングの一例を図 5.17に示す．

図 5.17において Node1(操作者ノード)，Node2，Node3 が s_type のノード，Node4 が d_type のノードとする．

時刻 T0 において Node1 において命令が出されるとすると，Tw 後の時刻 T2 が s_type のノードに対する理想的な再生開始時間となる．Node3 への命令到着時間を T1 とすると，まだ T2 に達していないので Node3 は Node1 と同時に T2 から情報提示を開始できる．Node2 への命令到着時間を T3 とすると，既に T2 を過ぎているため即座に情報提示を開始する．このように，Tw が小さいと s_type のノードが同時に提示開始を行なうことが難しくなるが，Tw が大きいくなると待ち時間が大きくなるため望ましくない．Node4 への命令到着時間を T4 とすると，Node4 ではさらに Tw+Tq 時間だけ待った後

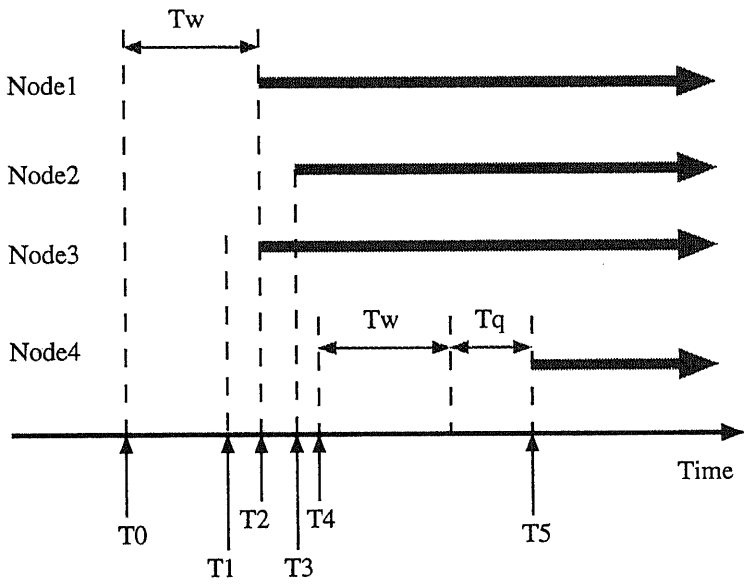


図 5.17: sync_with_multi における提示開始タイミング

で時刻 T5 から情報提示を開始する.

理想的な再生点

理想的な再生点の決定

理想的な再生点 (F_{IPP}) は 5.4.1節で述べた理想的な命令開始時間 (T_{Istart}) からの経過時間 (T_{elapse}) と，命令開始フレーム番号 (F_{Istart}) から式 (5.3) により算出する (図 5.18).

$$F_{IPP} = F_{Istart} + T_{elapse} \times P_{rate} + F_{offset} \tag{5.3}$$

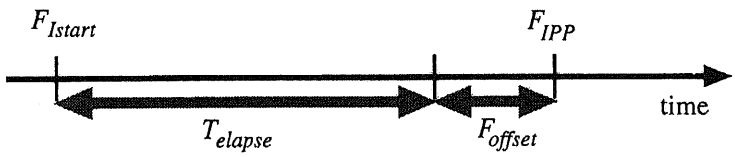


図 5.18: 理想的な再生点

ここで，P_{rate} は情報提示速度（33frame/sec 等，逆再生の際には負の値）を示す．ま

た，F_{offset} は 5.4.1節で述べる情報提示中の再生点調整による理想的な再生点の変化分を示す．

例えば，F_{Istart} が 22th unit，T_{elapse} が 2 秒，P_{rate} が 30units/ 秒，F_{offset} が 0 unit の場合，F_{IPP} は 82th unit となる．

情報提示中の調整

5.4.1節で述べたような，情報提示開始時の調整だけではネットワークの遅延変動などの様々な不確定要素に対して柔軟に対応できない．本節では，このような情報提示中の変動に対処する方法を示す．

ネットワークの遅延変動などにより，以下のような問題が生じる．

1. ネットワークの負荷が大きくなると，d_type のノードで LMS が RMS よりも遅れて到着するようになる．

2. 様々な変動により，各ノードでの再生点がバッファ上で適切な位置を取れなくなる．

1 に関しては，各 d_type ノードにおいて LMS を受信した際に，F_{offset} を調整することで対処することとする．

2 に関しては，グローバルな調整が必要となる．そこで，システム中に一つ存在する管理プロセスが情報提示中に各ノードの状態情報を収集し，理想的な再生点を動的に変化させることで情報提示中の再生点の調整を行なうこととする．

例えば，図 5.19では，全体的に各ノードの再生点がバッファの Tail に近い位置にある．この場合，ノード間の再生点の位置関係を保ったまま各ノードの再生点をバッファの先頭に近付けることで past バッファをある程度確保することが望ましい．

このようなシステムワイドな再生点調整を以下のように実装した．

1. 各ノードは定期的に各ノードの状態情報を管理プロセス（初期化プロセスと共用）に送る．ここで，各ノードの状態情報としては，各バッファの先頭フレーム番号，各バッファの大きさ，理想的な再生点，現在の動作モード（再生中等）及び，状態情報送出時刻とする．また，一時停止中には状態情報は送らない．

2. 管理プロセスでは，全てのノードからの同一時刻での状態情報が揃った時点で，理想的な再生点を全体として前後にどれだけ動かすか決める．ここで，IPP の調整は各ノードで利用可能な future バッファの大きさの minimum と past バッファの minimum とが等しくなる位置に IPP が来るように行なった．

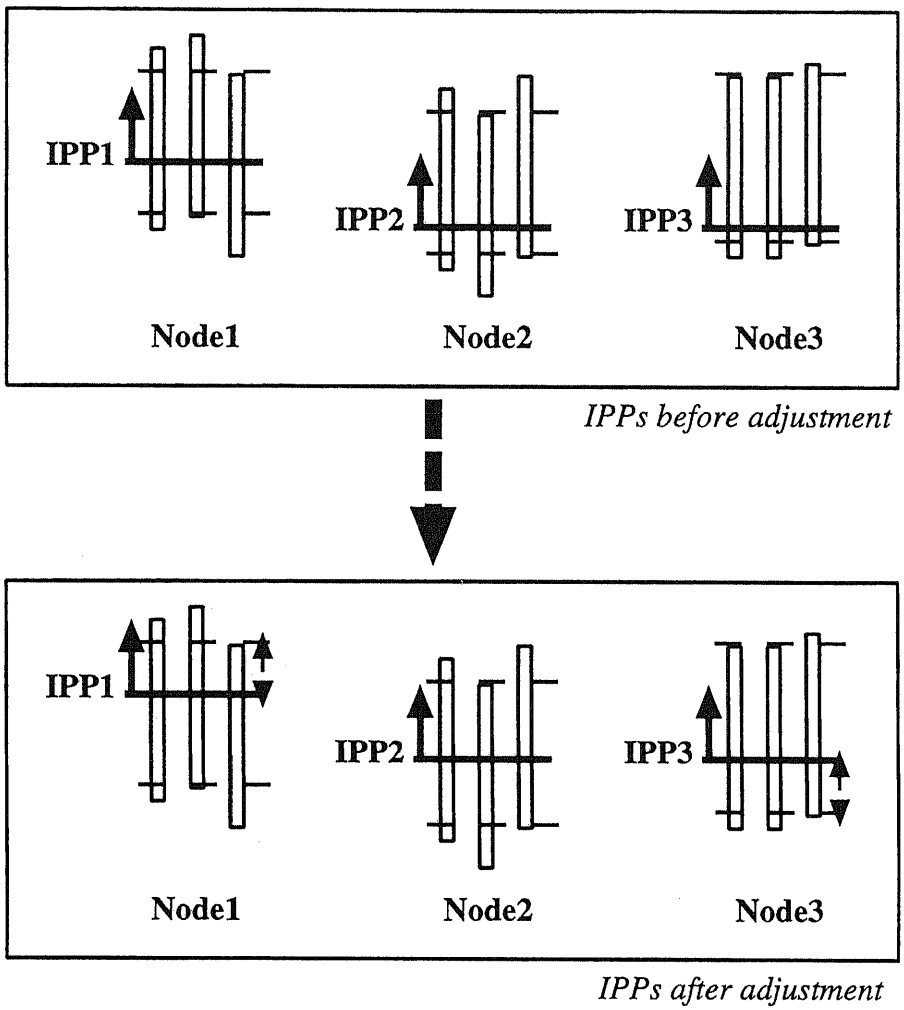


図 5.19: Adjustment of IPPs during Presentation

3. 管理プロセスは IPP の調整値を各ノードに送る.

4. 各ノードでは送られてきた調整値を F_{offset} に加える.

5.4.2　Live メディア情報

LMS としては，教育システムなどで先生が教材について説明する声，会議システムなどで資料について説明する人の声など，人の声を用いることが多いであろう.

そこで，本実装でも LMS として音声情報を用いることとする.

Sun の audio device は read，write 用に一つずつしか open できないため，各ノード

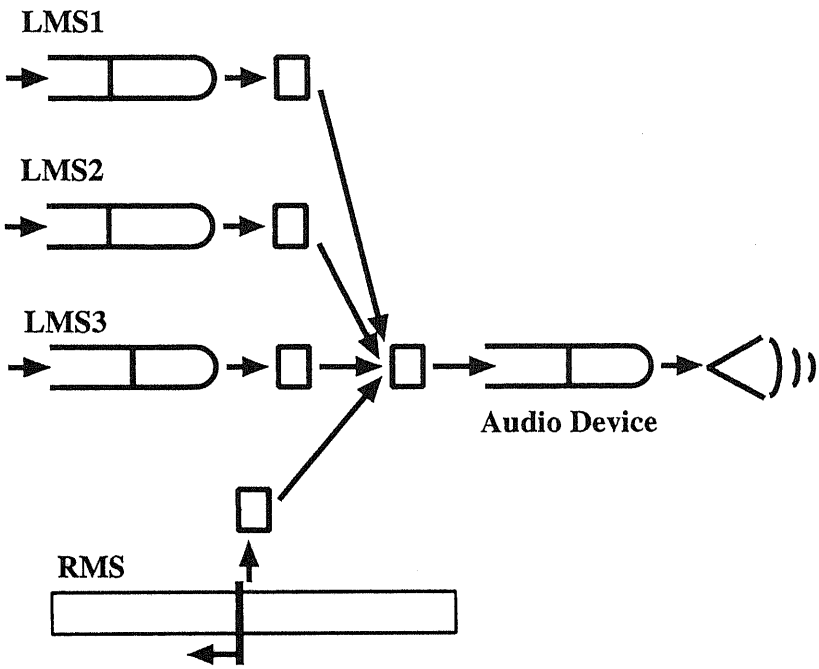


図 5.20: 音声情報の提示

では図 5.20のように音声情報源，発信者ノードなどから来る複数の音声情報を一つにまとめて audio device に write する必要がある．

発信者ノードからの情報送出

送信者ノードでは talk ボタンが押されると定期的に" 音声情報を audio device から read し他ノードに送出する procedure" が起動される．ただし，この procedure は図 5.21のように必ずしも定期的に起動されるとは限らず，従って一度に audio device から read するデータの大きさも変動する．

audio device から read したデータは有音，無音の判定を行ない，無音の場合，情報送出は行なわない．

有音の場合には図 5.21のように read したデータが有音部の先頭（図中の N）か，途中（図中の C）か，末尾（図中の T）か判別され，そのことを示す情報も送出フレームに付加する．また，有音部の先頭の場合，その時点で提示されているメディアフレーム番号から有音部の先頭が read された瞬間に提示されていたメディアフレーム番号を推定し，そのフレーム番号と，これまでに受信し発行された命令総数，及び，現在の状態（一時

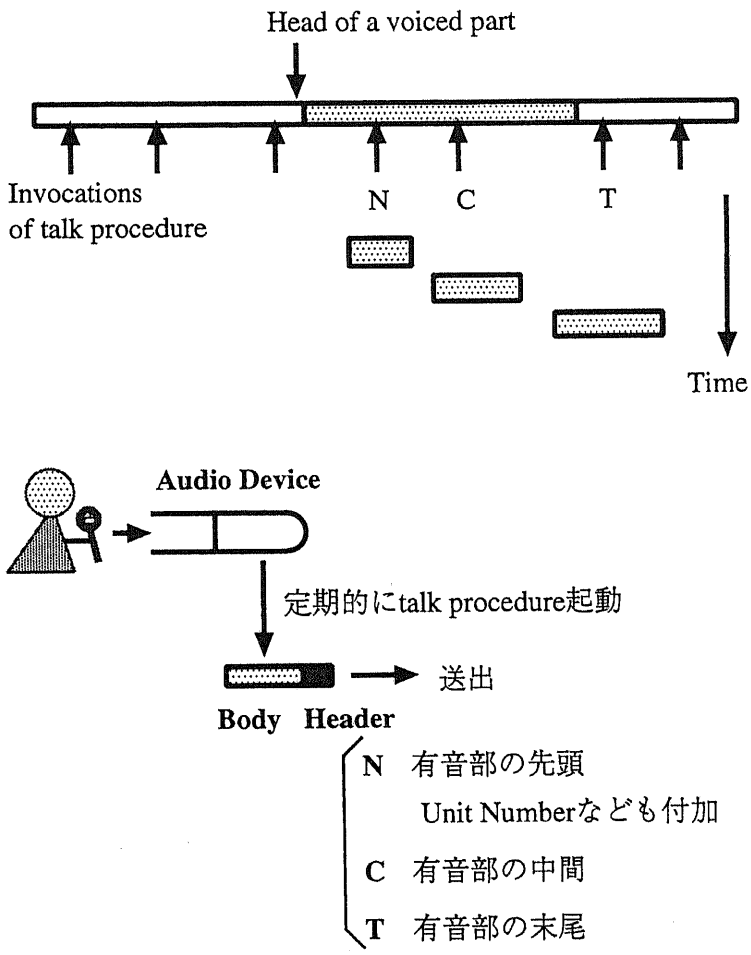


図 5.21: 発信者ノードからの送出フレーム

停止中など）も送出フレームに付加する．

図 5.22に LMS の送出，及び，提示の一例を示す．図 5.22では，まず，Node1 において 9 unit 目が提示されている時に（参加者が話すことで）LMS がシステム中に取り込まれる．この場合，9 unit 目を示す付加情報とともに取り込まれたメディア情報が各ノードにマルチキャストされる．Node2 で 6 unit 目が提示されている時にこの LMS フレームが到着したとすると，この LMS フレームは 9 unit 目が提示されるまで待ってから提示される．また，Node3 では 10 unit 目が提示されている時にこの LMS フレームが到着したとすると，すでに 9 unit 目は提示された後なので，到着した LMS フレームは即座に提示される．

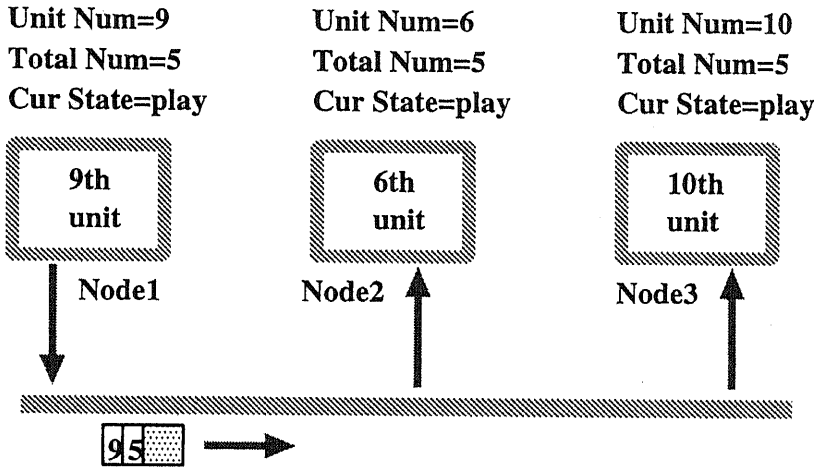


図 5.22: Delivery and presentation of a LMS

各情報受信ノードでの情報提示

各情報受信ノードでは発信者ノードから次々と送られてくる音声情報を発信者ごとにバッファに保持する．これらの音声情報は図 5.23のように，各有音部単位 (図中の Voiced Part) でまとめられ，それらの有音部は受信した順序で保持される．

図 5.23において， Frame Num は有音部の先頭に対応するフレーム番号を， Com Count, Com Mode はその有音部（の先頭）が送信者ノードで read された時点で送信者ノードにおいてそれまでに発行された命令総数，ノードの状態をそれぞれ示している．これらの情報は送信者ノードからの送出フレームの付加情報として取得できる．

Present Flag はその有音部が提示可能状態になったことを示すフラグである．このフラグは各受信ノードにおける再生点が，送信者ノードにおいてこの有音部が read された時点における再生点に到達した際に ON になる．この判定は [Frame Num, Com Count, Com Mode] と各受信ノードでの再生点，命令発行総数，状態とを比較することで行なう．

sync_with_all などの s_type のノードでは，有音部を受信した時点で受信ノードにおける再生点は既に送信者ノードでの read の時点での再生点を過ぎているので，受信と同時に Present Flag も ON になる．

定期的に起動される音声提示 procedure は， Present Frag が ON になっている有音部を，先頭から順番に audio device に write していく．

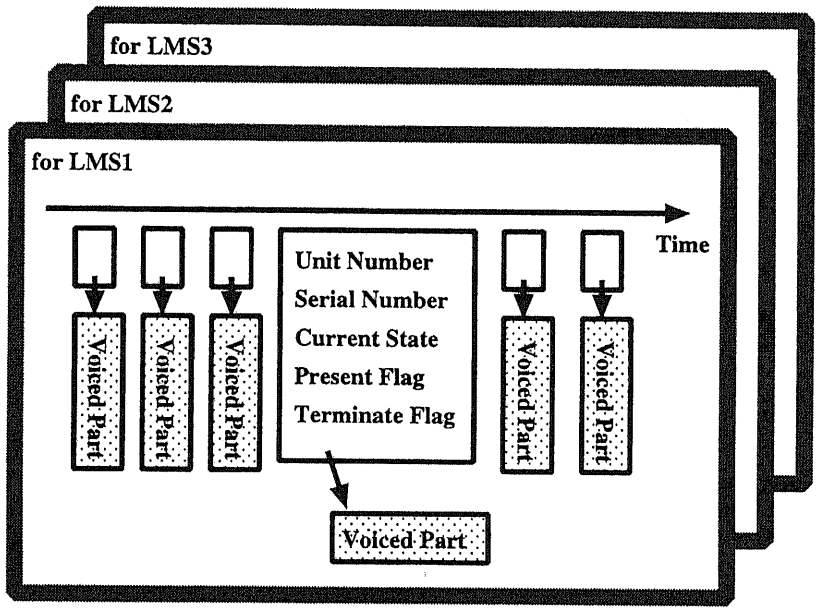


図 5.23: LMS 受信用 queue

audio device に write された有音部はバッファから解放される.

Terminate Flag はこの有音部に関する情報を受信している際に OFF, 受信が完了した時点で ON になるフラグである. 音声提示 procedure はこのフラグが OFF である有音部に関しては, Present Flag が ON の際にも, 音声提示の連続性が保証できる程度の情報が受信されていなければこの有音部を audio device に write しない.

5.5 R&L 同期方式の評価

本節では，試作システムにおける R&L 同期方式の評価について述べる．

5.5.1 評価環境

Ethernet で接続された 3 台の UNIX workstation を用いて R&L 同期機構の評価を行なった．3 台のマシーンの役割を以下に示す．

SS10 (Sparc station 10) RMS 情報源 (video, text), LMS 発信 (voice), 情報提示の 3 つの役割を兼ねる．

IPX (Sparc station IPX) 情報源 (audio), LMS 発信 (voice), 操作者, 情報提示の 4 つの役割を兼ねる．

SS1 (Sparc station 1) 情報提示のみを行なう．

バッファの大きさは video, text に対しては 20 フレーム分, audio に対しては 20+20 フレーム分とし，$T_{int} = 100ms$ とした．初期状態としては各メディアバッファが 0 フレーム目からのメディア情報で半分だけ満たされている状態とした．

また，本測定の際には他のプロセスは走っていなかった．

5.5.2 評価結果

RMS の提示タイミングと LMS の提示タイミングを以下のように測定した．

- RMS の提示タイミング

  IPX において再生命令が出され，各ノードで 0 フレーム目からの再生が行なわれた際に各ノードにおいて実際に各フレームが提示された時間を測定した．ただし，音声フレームの情報提示時間はそのフレームが音声 Queue に挿入された時間と，その時点での Queue の長さにより算出した値を用いている．

- LMS の提示タイミング

  各情報提示ノードにおいて RMS に対する LMS の情報提示タイミングの遅れを測定した．RMS に対する LMS の提示遅れは各有音部の先頭が audio device に write される際に測定した．

sync_with_one

図 5.24に sync_with_one の場合の RMS の提示タイミングを示す.

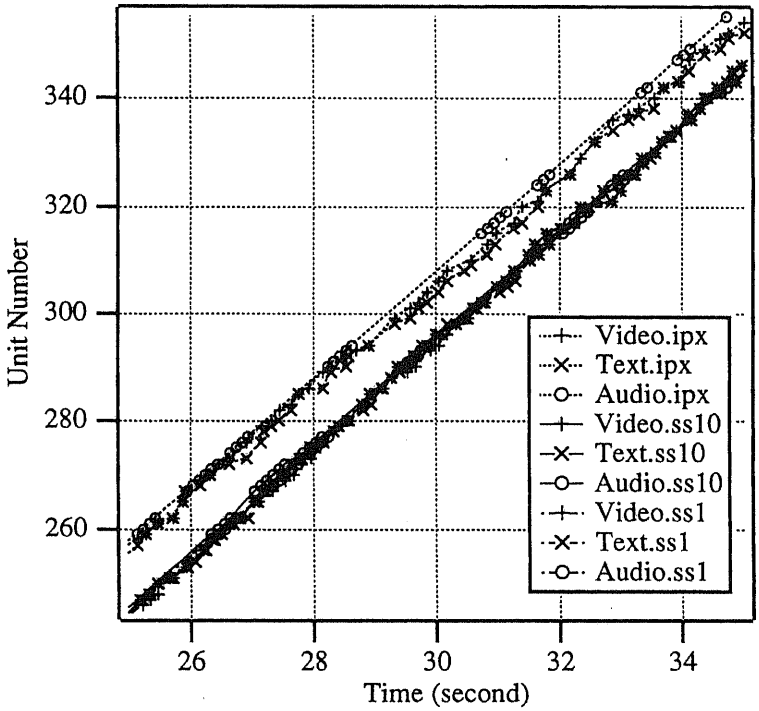


図 5.24: sync_with_one の場合の情報提示タイミング

表 5.1に LMS の提示タイミングを示す. ただし, 表 5.1中の値は有音部の先頭が提示される際の LMS の遅れの平均, 及び, 最大値 (括弧中の値) を示している.

presentation nodes / LMSs	SS10	IPX	SS1
a LMS generated at IPX	1.36 units (5 units)	0 units (0 units)	1.36 units (4 units)

表 5.1: sync_with_one の場合の各ノードでの LMS の情報提示タイミングの遅れ

sync_with_multi

図 5.25に sync_with_multi の場合の RMS の提示タイミングを示す.

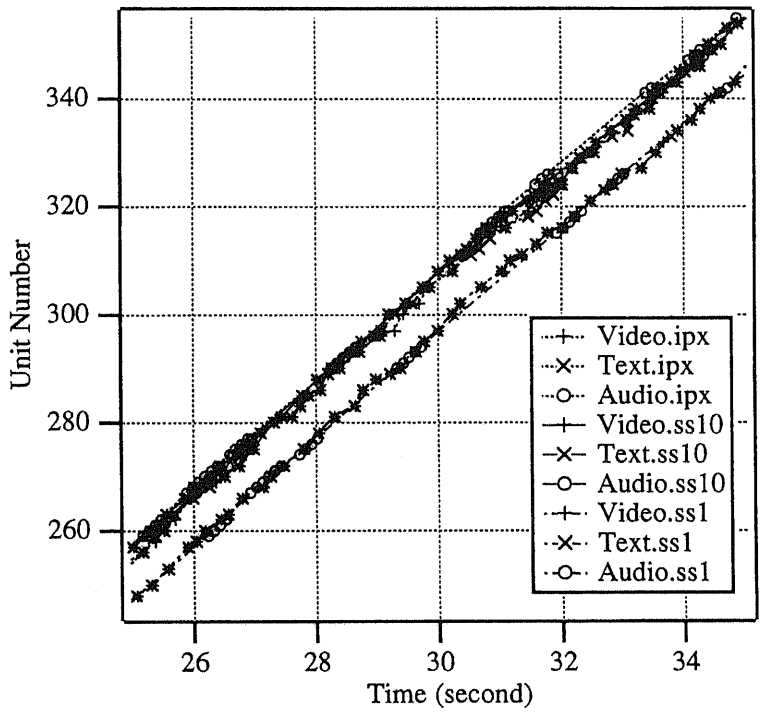


図 5.25: sync_with_multi の場合の情報提示タイミング

表 5.2に LMS の提示タイミングを示す.

presentation nodes / LMSs	SS10	IPX	SS1
a LMS generated at SS10	0 units (0 units)	10.95 units (15 units)	1.95 units (5 units)
a LMS generated at IPX	10.14 units (13 units)	0 units (0 units)	0.43 units (2 units)

表 5.2: sync_with_multi の場合の各ノードでの LMS の情報提示タイミングの遅れ

sync_with_all

図 5.26に sync_with_all の場合の RMS の提示タイミングを示す.

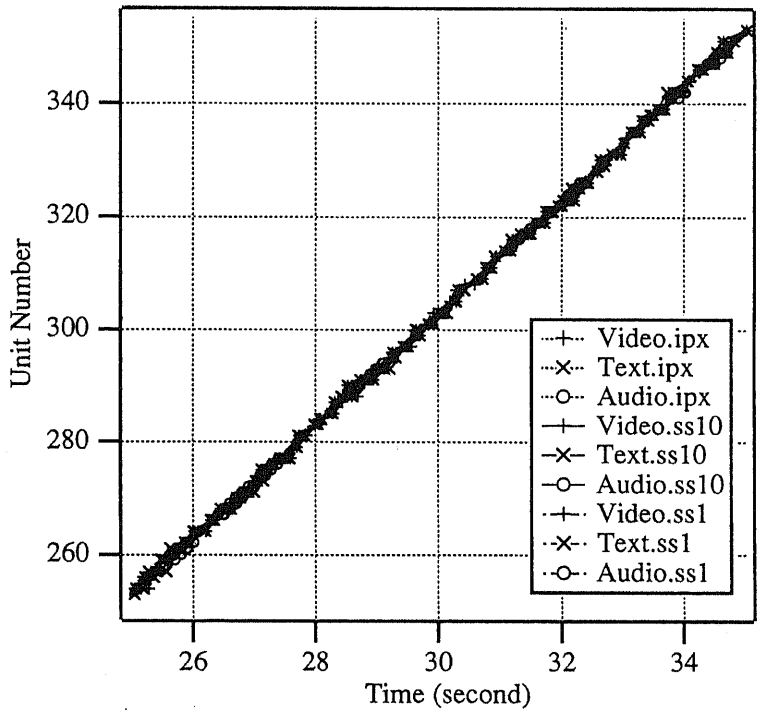


図 5.26: sync_with_all の場合の情報提示タイミング

表 5.3に LMS の提示タイミングを示す.

presentation nodes / LMSs	SS10	IPX	SS1
a LMS generated at SS10	0 units (0 units)	9.75 units (11 units)	10.75 units (12 units)
a LMS generated at IPX	12.38 units (14 units)	0 units (0 units)	11.25 units (14 units)

表 5.3: sync_with_all の場合の各ノードでの LMS の情報提示タイミングの遅れ

5.5.3　考察

図 5.24，図 5.25，図 5.26から，以下に示すことがわかる．

- 各ノードにおいて 3 つのメディアはほぼ同期して提示されている．
- d_type ノードでは s_type ノードに比べて CPP が遅れている．
- s_type ノードでは CPP はほぼ等しくなっている．

以上の結果から，5.4.1節，5.4.1節等で示した R&L 同期実現のための再生点調整はきちんと動作していることがわかる．

表 5.1，表 5.2，表 5.3から以下に示すことがわかる．

- d_type ノードでは LMS の提示遅れは小さい (5 unit 以内)．
- s_type ノードでは LMS の提示遅れは大きい (平均 10 unit 程度)．

以上の結果から，LMS は R&L 同期のセマンティクスをほぼ満たしながら提示されていることがわかる．

また，3.4.3節で述べた場合のようにメディアごとに発生間隔が異なる場合，LMS 中の送出フレームへの付加情報，IPP などはすべて，最も発生間隔が大きいメディアの同期時間軸に従うものとする．各ノードでの再生点調整の際には，発生間隔が小さいメディアの CPP を最も発生間隔が大きいメディアの同期時間軸上での値に変換し，その値を F_{real} として式 (5.2) 中の distance の値を求める必要がある．つまり，図 3.8中の Media2, Media3 のように再生間隔が最も大きいメディア（この場合 Media1）の 1 フレームの間に N_g $(N_g > 0)$ フレームが存在する場合，F_{CPP} から F_{real} への変換は式 (5.4) により行なわれる．

$$F_{real} = 1 + (F_{CPP} - 1)/N_g \tag{5.4}$$

それ以外はフレーム発生間隔がすべてのメディアで等しい場合と同様にしてメディア間の同期を取ることができる．

本実装においては LMS として人の声を用いた．遠隔会議システムなどでは人の声以外にも，参加者の顔等といった動画情報を LMS として用いることも考えられる．そのような場合にも LMS 中の各フレームが取り込まれた際に提示されていた RMS 中のメディア情報の unit 番号をその LMS 中のフレームに付加することで本論文で示した同期方式をそのまま適用できる．

第 6 章

分散マルチメディアシステムの実現に向けて

本章では第3章，第4章，及び，第5章で示した検討を踏まえ，分散マルチメディアシステムの実現に向けてさらに必要な諸技術について述べる．

本章では第3章，第4章，第5章で示した検討を踏まえ，分散マルチメディアシステムの実現に向けてさらに必要な諸技術について考える．

第2章で示したように，分散マルチメディアシステムにおいては従来の高速化技術だけではなく，ユーザへの情報提示の際のメディア品質を考慮することが重要となる（メディア間同期もマルチメディア情報をユーザへ提示する際の QOS の尺度のひとつであると考えることができ，個々のメディアの QOS との trade-off を考えながら全体としてのメディア品質を上げるようにすることが必要となる．）．以下，ユーザに対してより品質の高いマルチメディアメディア情報を提示するという視点から分散マルチメディアシステムの実現に向けてさらに必要な諸技術を示す．

- ユーザへの QOS の尺度の確立

  マルチメディアシステムにおいてはユーザに対してより高品質の情報を提示することが最も重要である．しかし，ユーザにとってのメディア品質は万人に取って同じとは言い難く，また，メディア情報をどのような目的で利用するかによっても違ってくる．従って，すべてのメディア情報を，これは grade 3 のメディア情報であるなどといったように，統一の尺度でランク付けすることはなかなか困難であろう．

  しかし，マルチメディアシステムにとってより品質の高い情報を提示することが最重要であるからには，どのようなマルチメディア情報が品質が高いと言えるのかということをわかっている必要があろう．逆に言えば，より品質の高い情報をユーザに対して提示できるようにシステムを構成していくことが重要であると考えられる．

  ユーザへの QOS の尺度の決め方としては，例えば，ニュース，音楽，ショッピングなどといった用途ごとに QOS の尺度を決めておき，それぞれの用途ごとの尺度に対して若干，補正の余地を残しておくことで各ユーザの主観に応じた微調整が行なえるようにするなどといった方法が考えられる．

- QOS 保証を実現するための基盤技術の確立

  図 2.11で示したように，ユーザからの QOS 要求を実現するためには OS，ネットワークプロトコルなどで資源を確保する必要がある．

  UNIX などの non-preemptive な OS では各タスクの実行に要する時間を予測することはできず，QOS 保証を実現することはできない．また，従来の TCP/IP など

のネットワークプロトコルではネットワーク資源を確保することでネットワークのQOS を保証するなどといったことはできない.

そこで，このようなQOS 保証の基盤を確立するために2章で述べたようにコンピュータ資源へのQOS 要求，ネットワーク資源へのQOS 要求などに対して資源を確保し，要求されたQOS を保証するための研究が現在様々な研究機関で活発に行なわれている．今のところ，これらの研究はリアルタイムOS を拡張したマルチメディアOS によるコンピュータ資源の確保，FDDI の同期転送モードを利用したネットワーク資源の確保といったものが主であるが，将来的にはATM 技術を用いた広帯域ネットワークと高速な端末で構成されるシステムにおいてend-to-end のQOS 保証を実現するための基盤技術が確立されていくであろう.

- 資源に応じた品質のメディア情報を提示するための技術の確立

  将来，上述のQOS 保証実現のための基盤技術が整った際にも，資源に応じた品質のメディア情報を提示できる機構が備わっていることは重要である.

  これは以下の理由による.

  - 新たに遠隔会議などのサービスを実行しようとした際に要求した資源が確保できないかもしれない．このような際の対応としては，新たなサービスの実行を拒絶する，新たなサービスや既に実行中のサービスのQOS を下げることで新たなサービスの実行を可能にするなどといったことが考えられる．後者の場合，各サービスのQOS をできるだけ下げずに新たなサービスのための資源を提供することが重要となる.
  - あるQOS レベルを保証するのに必要な資源は常に全て使用されているとは限らない．むしろ，ある程度余裕を持って資源を確保しておかないとそのQOS レベルを保証することは難しいであろう．与えられた資源をより有効利用するためにはQOS 要求に（最小値と最大値を指定するなどといった方法で）幅を持たせ，より柔軟に資源割り当てを行なえることが望ましい．このような場合にも資源に応じてより良い品質のメディア情報を提示できる技術が重要となる.

DMSIC でも，直観的に判断できる範囲でユーザからのQOS を考慮しており，また，様々なパラメータを調整することでCPU などの資源が足りなくなった時にど

こからメディア品質を落していくか制御できる枠組は提供している．つまり，ユーザがパラメータをチューニングすることでCPUが足りなくなった際に個々のユーザが望むように全体としてのメディア品質が劣化していくようにすることができる．

以上，ユーザにより良い品質のマルチメディア情報を提示するといった視点から分散マルチメディアシステムの実現に向けてさらに必要な諸技術について述べてきた．分散マルチメディアシステムはコンピュータ，通信，メディア統合技術などといったかなり広範囲の様々な技術を基礎として成り立っており，今後さらにそれらの構成要素を分散マルチメディアシステムにとって最適なものとしていくことが必要であろう．また，上述した以外にも，マルチキャスト通信など分散マルチメディアシステムを構成するための様々な技術を確立していくことが必要となるであろう．

第 7 章

結論

本論文では，分散マルチメディアシステムにおける同期方式について述べた．

第 2章ではまず，研究の背景となる知識として分散マルチメディアシステムを実現する上での諸技術について述べた．また，マルチメディアにおけるメディア間同期に関する従来の研究を概観し，その研究動向をまとめた．

第 3章では，遠隔会議システムにおける諸技術を概観し，その中での同期の位置付けを示した．また，遠隔会議システムでの同期の問題点を示し，本研究の目的を明確化した．さらに，本研究で対象とするシステムの構成を示し，蓄積型情報源から取り出されるメディア情報 (RMS) と参加者の声などの生のメディア情報 (LMS) との間の情報提示の際のタイミングに基づいた同期 (R&L 同期) のセマンティクスを定めた．

第 4章では，予めネットワーク，端末などの資源を確保することで QOS が保証できるような環境において本システムを設計する際の設計条件について検討した．

本検討ではまず，以下の 3 つについてその算出手法を示した．

1. 命令に対する待ち時間が与えられた際に必要なバッファ量の算出法
2. 各ノードで各メディアごとのバッファ量が与えられた際の同期判定及び必要な待ち時間の算出手法
3. 各ノードでメディア全体のバッファ量が与えられた際の同期判定及び必要な待ち時間の算出手法

算出手法 1 は新たにシステムを設計する際に，必要なバッファ量を見積もる場合などに用いることができる．この手法を用いてバッファ量の算出を行なった結果，再生などの命令に対する応答時間の設定値をある程度まで大きくすることで必要なバッファ量を削減できることを確認した．

算出手法 2 は端末として専用端末が用いられる場合などのように予め各端末に各メディアごとのバッファ（フレームバッファなど）が設けられていることを想定できる場合に R&L 同期判定及び必要な待ち時間の算出を行なうのに用いることができる．

算出手法 3 はワークステーションの主記憶のように，全てのメディアが全体として一つのバッファを共有するような場合に R&L 同期判定及び必要な待ち時間の算出を行なうのに用いることができる．

第 5章では，LAN 環境において構築したシステムにおいて R&L 同期を実現する機構を示した．

本同期機構においては各ノードのバッファ中に理想的な再生点 (IPP) を定め，各メディアの再生点 (CPP) を IPP に近付けることで各ノードにおけるメディア間の同期を取る．また，IPP をノード間で調整することで R&L 同期を実現する．

本同期機構の特徴を以下に示す．

- R&L 同期のセマンティクスを満たしながら各メディア情報が提示される．
- 各ノードの負荷変動に柔軟に対応しつつ情報提示の際のメディア間の同期を保つことができる．

また，3 台の UNIX ワークステーションを用いた評価により本同期機構の有効性を示した．

第 6章では，第 3章，第 4章，第 5章で示した検討を踏まえ，ユーザにより良い品質のマルチメディア情報を提示するといった視点から分散マルチメディアシステムの実現に向けてさらに必要な諸技術を示した．

以上，分散マルチメディアシステムにおける同期方式について論じた．R&L 同期は RMS が予め存在する情報であるために予めバッファリングしておくことが可能であるのに対し，LMS は生の情報であるためにできるだけ早く提示されることが望ましいといった RMS と LMS の性質の違いに着目した同期である．将来，メモリコストの低下により端末においてフレームバッファなどの大容量メモリが備わり，画像圧縮技術などの進歩により各メディアの情報量がさらに少なくなれば，各端末においてより長い時間分のメディア情報をバッファリングすることができるようになると考えられ，R&L 同期はますます実現し易くなるであろう．

コンピュータの高速化，ネットワークの広帯域化など分散マルチメディアシステムを実現するための基盤は整いつつあり，本論文で示した同期方式が今後現れるであろう様々な分散マルチメディアシステムの構築において役立つことを期待したい．

謝辞

学位論文を書き終えるにあたり，大学院の５年間にわたり御指導賜わり，本研究の立ち上げを始めとする節目節目において数々の貴重な御助言により研究を支えて頂きました齊藤忠夫教授，ならびに相田仁助教授に心から感謝の意を表します．

また，日頃の研究生活において大変お世話になりました齊藤・相田研究室の職員の富山忠宏助手，千葉新吾技術官，元大学院生および現大学院生の付宏女史，サーダン・ゾーカイ氏，黒岩実氏，ウイ・エン・ティアム氏，ピシット・タンティロ ーッチャナキッチャカーン氏，オヌル・アルティンタシュ氏，スポット・ティアラウ ット氏，小口正人氏，川口洋二氏，ボラブット・プライワン氏，川合史朗氏，長澤育範氏，増岡義政氏，チュア・ヤオチアン氏，金炳錫氏，玉木剛氏，藤田博文氏，寺西祐人氏，柳寅太氏，青木輝勝氏，小島章裕氏，高橋淳一氏，箕浦真氏，ハーネ・リャム・ミン氏，日高宗一郎氏，林博之氏，森野博章氏ならびに卒論生の皆様，研究生の皆様，秘書の篠崎恭子女史，芦川みさき女史，大江寿美子女史に感謝致します．

1994 年 12 月 20 日

大野 隆一

参考文献

[ISO 93] ISO/IEC CD 13522-1, Coded representation of multimedia and hypermedia information objects (MHEG) (1993).

[Kretz 92] Kretz F., and Colaitis F.: "Standardizing Hypermedia Information Objects", IEEE Commun. Mag., **30**, 5, pp.60-70(1992).

[NHK 94] NHK 放送技術研究所編: "ディジタル放送技術辞典", 丸善 (1994).

[Yasu 91] 安田浩編著: "マルチメディア符号化の国際標準", 丸善 (1991).

[Hara 91] 原島博監修: "画像情報圧縮", オーム社 (1991).

[Gall 91] Gall D.L., "MPEG : A Video Compression Standard for Multimedia Applications", Communications of the ACM, April 1991

[Asa 91] 浅見他: "マルチメディア OS 生まれる", 日経エレクトロニクス,no.534, 日経 BP 社 (1991).

[Apple 92] "QuickTime Starter Kit User's Guide", Apple Computer,Inc (1992).

[Lecho 89] Lehoczky, J. P., Sha, L. and Ding, Y.: "The Rate-Monotonic Scheduling Algorithm: Exact Characterization and Average Case Behavior", In Proceedings of 10th IEEE Real-Time Systems Symposium (Dec. 1989).

[Toku 89] Tokuda, H. and Mercer, C. W.: "ARTS: A Distributed Real-Time Kernel", ACM Operating Systems Review, Vol. 23, No. 3 (July 1989).

[Toku 92] Tokuda, H., Tobe, Y., Chou, S. T. C. and Moura, J. M. F.: "Continuous Media Communication with Dynamic QOS Control Using ARTS with an FDDI Network", In Proceedings of ACM SIGCOMM '92 (Aug. 1992).

[Ara 93] Arakawa, H., Katcher, D. I., Strosnider, J. K. and Tokuda, H.: "Modeling and Validation of Real-Time Mach Scheduler", In Proceedings of ACM SIGMETRICS '93 (1993).

[Ander 90] Anderson, D. P., Tzou, S., Wahbe, R., Govindan, R., and Andrews, M.: "Support for continuous media in the DASH System", In Proceedings of the 10th International Conference on Distributed Computing systems, pp.54-61(1990).

[Ander 93] Anderson, D. P.: "Metascheduling for Continuous Media", ACM Transactions on Computer Systems, **11**, 3, pp.226-252(1993).

[Merc 90] Mercer, C. W., Ishikawa, Y., and Tokuda, H.: "Distributed Hardstone: A distributed real-time benchmark suite", In Proceedings of the 10th International Conference on Distributed Computing systems, pp.70-77(1990).

[Bach 86] Back, M. J.: "The Design of the UNIX Operating System", Prentice-Hall (1986). (翻訳: 坂本他:"UNIX カーネルの設計", bit 別冊.).

[Leff 89] Leffler, S. J., McKusick, M. K., Karels, M. J., and Quarterman, J. S.: "The Design and Implementation of the 4.3BSD UNIX Operating System", Addison-Wesley (1989). (翻訳: 中村他:"UNIX4.3BSD の設計と実装", 丸善.).

[Toku 93] 徳田英幸, 西田竹志他: "マルチメディア・ネットワーク", 日経エレクトロニクス増刊号,no.583, 日経 BP 社 (1993).

[Tomi 92] 富永英義他,"B-ISDN 入門", オーム社 (1992).

[Koi 93] 濃沼健夫, 赤池武志 "マルチメディア通信と ATM ネットワーク", 情報処理学会マルチメディア通信と分散処理研究会,61-8,1993.

[Aki 91] 秋山他,"広帯域 ISDN 特集 ", 電子情報通信学会誌, 1991-11.

[Mura 90] 村上伸一, 酒井善則共著: "マルチメディア通信システム", 昭晃堂 (1990).

[Ander 91] Anderson D.P., and Homsy G.: "A Continuous Media I/O Server and Its Synchronization Mechanism", IEEE COMPUTER, **24**, 10, pp.51-57(1991).

[Fuji 93] 藤川和利, 下條真司, 松浦敏雄, 西尾章治郎, 宮原秀夫: "分散型ハイパメディアシステム Harmony における情報間同期機構の実現", 信学論 *(D-I),J76-D-I,9,* (1993-9)

[Lit 91] Little T.D.C.: "Multimedia Synchronization Protocols for Broadband Integrated Services", IEEE J. Sel. Areas Commun., **9**, 9, pp.1368-1382(1991).

[Rama 93] Ramanathan S., and Rangan P.V.: "Adaptive Feedback Techniques for Synchronized Multimedia Retrieval over Integrated Networks", IEEE/ACM Trans. Networking., **1**, 2, pp.246-260(1993).

[Klara 92] Klara, Jonathan, "An Integrated Multimedia Architecture for High-Speed Networks", *Multimedia '92.*

[Stein 90] Steinmetz R.: "Synchronization Properties in Multimedia Systems", IEEE J. Sel. Areas Commun., **8**, 3, pp.401-412(1990).

[Li 92a] Li L., Karmouch A., and Georganas N.D., "Synchronization in Real Time Multimedia Data Delivery", in Procceedings of the IEEE ICC '92., pp.587-591(1992).

[Li 92b] Li L., Karmouch A., and Georganas N.D., "Real-time Synchronization Control in Multimedia Distributed Systems", *in Procceedings of the IEEE Multimedia '92 Workshop.,* April 1992.

[Naka 91] 中島, 三浦, 矢部,"ATM 網を用いた TV 会議システムにおけるタイミングについての一検討", 信学技報, OS91-11.

[Rang 93] Rangan P.V., Vin H. M., and Ramanathan S.: "Communication Architectures and Algorithms for Media Mixing in Multimedia Conferences", IEEE/ACM Trans. Networking., **1**, 1, pp.20-30(1993).

[Ahuja 92] Ahuja S.R., Ensor J.R., and Horn D.N.: "Coordination and Control of Multimedia Conferencing", IEEE Commun. Mag., **30**, 5, pp.38-43(1992).

[Clark 92] Clark W.J.: "Multipoint Multimedia Conferencing", IEEE Commun. Mag., **30**, 5, pp.44-50(1992).

[Ahuja 88] Ahuja S.R., and Ensor J.R.: "The Rapport multimedia conferencing system", Proc. ACM Int. Conf. on Office Information Systems, pp1-8(1988).

[Sakata 92] 阪田史郎著: "グループウェアの実現技術", ソフト・リサーチ・センター (1992).

[Matu 91] 松下温著編: "グループウェア入門", オーム社 (1991).

[Ellis 91] Ellis C.A., Gibbs S.J., and Rein G.L., "Groupware : some issues and experiences", Communications of the ACM, January 1991

[Ellis 89] Ellis C.A., and Gibbs S.J., "Concurrency Control in Groupware Systems", Proceedings of the Conference on the Management of Data, May 1989

[Ohm 92] Ohmori T., Maeno K., Sakata S., Fukuoka H., and Watabe K., "Distributed Cooperative Control for Sharing Applications Based on Multiparty and Multimedia Desktop Conferencing System: MERMAID", Proceedings of the 12th International Conference on Distributed Computing Systems, June 1992

[Naka 94] 中村章, 清山信正, 池沢龍, 都木徹, 宮坂栄一: "リアルタイム話速変換型受聴システム", 日本音響学会誌,*50-7*,(1994)

[Mills 93] Mills D.L., "Simple Network Time Protocol (Version 3) specification, implementation and analysis", *DARPA Network Working Group Report RFC-1305.*, March 1993.

発表文献

学会誌論文

[1] 大野, 相田, 齊藤: "マルチメディア遠隔提示システムの同期条件の検討", 電子情報通信学会誌 (B-I), 採録済.

[2] Ohno R., Aida H., and Saito T., "A Media Synchronization Mechanism for a Distributed Multimedia System with Interactive Control", submitted to IEICE Transactions on Communications (EB-I).

国際学会

[3] Ohno R., Aida H., and Saito T., "An Implementation of Media Synchronization Mechanism for a Distributed Multimedia System with Interactive Control", In Proc. of IPSJ/IEEE 9th International Conference on Information Networking.

[4] Ohno R., Aida H., and Saito T., "Implementation of a Media Synchronization Mechanism for a Remote Conferencing System", In Proc. of 1995 Pacific Workshop on Distributed Multimedia Systems.

国内大会

[5] 大野, 相田, 齊藤: "リアルタイム分散エディタにおけるバッファ間の一貫性制御方式", 第 45 回情報処理学会全国大会,1U-6,1992.

[6] 大野, 相田, 齊藤: "マルチメディアの会話型操作についての一検討", 1993 年電子情報通信学会春季全国大会,D-217,1993.

[7] 大野, 相田, 齊藤: “会話型操作を伴う分散マルチメディアシステムの実装”, 1993 年電子情報通信学会秋季全国大会,D-151,1993.

[8] 大野, 相田, 齊藤: “会話型操作を伴う分散マルチメディアにおける情報間同期機構の実現”, 第 48 回情報処理学会全国大会,6C-1,1994.

[9] 大野, 相田, 齊藤: “会話型操作を伴う分散マルチメディアにおける情報間同期機構の評価”, 第 49 回情報処理学会全国大会,1D-5,1994.

[10] 大野, 相田, 齊藤: “分散マルチメディアシステム DMSIC におけるメディア間同期機構”, 1995 年電子情報通信学会春季全国大会,1995 (発表予定).

[11] 江濱, 大野, 相田, 齊藤: “シナリオ再生のための分散マルチメディアシステム”, 1995 年電子情報通信学会春季全国大会,1995 (発表予定).

研究会

[12] 大野, 寺西, 相田, 齊藤: “マルチメディアの会話型操作における同期方式の検討”, 情報処理学会マルチメディア通信と分散処理ワークショップ, 1993-3.

[13] 矢野, 寺西, 大野, 相田, 齊藤: “会話型操作を伴うマルチメディアシステムにおける同期方式”, 情報処理学会マルチメディア通信と分散処理研究会,65-11,1994.

[14] 大野, 相田, 齊藤: “会話型操作を伴う分散マルチメディアシステムにおける同期機構”, 情報処理学会マルチメディア通信と分散処理研究会,66-21,1994.

[15] 大野, 相田, 齊藤: “マルチメディア遠隔提示システムの同期条件の検討”, 情報処理学会マルチメディア通信と分散処理研究会,66-22,1994.

[16] 大野, 相田, 齊藤: “分散マルチメディアシステム DMSIC における同期機構の実現と評価”, 情報処理学会マルチメディア通信と分散処理研究会,69-8,1995.

[17] 大野, 相田, 齊藤: “マルチメディア遠隔提示システムの設計に関する一検討”, 情報処理学会マルチメディア通信と分散処理研究会,69-9,1995.