INFORMATION TO USERS

This manuscript has been reproduced from the microfilm master. UMI films the text directly from the original or copy submitted. Thus, some thesis and dissertation copies are in typewriter face, while others may be from any type of computer printer.

The quality of this reproduction is dependent upon the quality of the copy submitted. Broken or indistinct print, colored or poor quality illustrations and photographs, print bleedthrough, substandard margins, and improper alignment can adversely affect reproduction.

In the unlikely event that the author did not send UMI a complete manuscript and there are missing pages, these will be noted. Also, if unauthorized copyright material had to be removed, a note will indicate the deletion.

Oversize materials (e.g., maps, drawings, charts) are reproduced by sectioning the original, beginning at the upper left-hand corner and continuing from left to right in equal sections with small overlaps.

Photographs included in the original manuscript have been reproduced xerographically in this copy. Higher quality 6" x 9" black and white photographic prints are available for any photographs or illustrations appearing in this copy for an additional charge. Contact UMI directly to order.

ProQuest Information and Learning
300 North Zeeb Road, Ann Arbor, MI 48106-1346 USA
800-521-0600

UMI®
A Multimedia Mail System With Real-time Support

Elaine Li

A Thesis

in

The Department

of

Computer Science

Presented in Partial Fulfilment of the Requirements for the Degree of Master of Computer Science at Concordia University
Montreal, Quebec, Canada

February 2001

© Elaine Li, 2001
The author has granted a non-exclusive licence allowing the National Library of Canada to reproduce, loan, distribute or sell copies of this thesis in microform, paper or electronic formats.

The author retains ownership of the copyright in this thesis. Neither the thesis nor substantial extracts from it may be printed or otherwise reproduced without the author's permission.

L’auteur a accordé une licence non exclusive permettant à la Bibliothèque nationale du Canada de reproduire, prêter, distribuer ou vendre des copies de cette thèse sous la forme de microfiche/film, de reproduction sur papier ou sur format électronique.

L’auteur conserve la propriété du droit d’auteur qui protège cette thèse. Ni la thèse ni des extraits substantiels de celle-ci ne doivent être imprimés ou autrement reproduits sans son autorisation.

0-612-59333-9
ABSTRACT

A Multimedia Mail System with Real-time Support

Elaine Li

As a multimedia mail is typically constructed over a period of time and has different mail readers, it is inevitable that its structure may change over time. In this thesis, we design a multimedia mail system with real-time scheduling support for authoring and presentation. We use the term authoring for the composition of a multimedia mail, and presentation for the reading of the mail. The major design objectives include flexible temporal and spatial specification, efficient algorithms to support incremental consistency check, presentation schedule generation, and effective editing/filter functions for both authors and readers.
Table of Contents

Chapter 1  Introduction.................................................................1

1.1  Multimedia Mail with Time and Space requirements ..................1

1.1.1  Definition of Multimedia Mail ......................................3

1.2  Mail presentation satisfying time and space requirements ..........4

1.2.1  Multimedia Mail System Vs. Traditional text-based Mail system ....4

1.2.2  Analyzing the presentation requirement of existing multimedia system ....6

1.3  Thesis Contribution ....................................................................7

1.3.1  Flexible Multimedia Mail Authoring/Presenting ....................8

1.3.2  Efficient algorithms for solving the consistent difference constraints ....8

1.3.3  Flexible editing functions for mail readers ..........................9

1.3.4  Real-time support ..................................................................9

1.4  Organization of the thesis ..........................................................10

Chapter 2  Multimedia Mail System Model .....................................11

2.1  Analyzing Authoring Models ...................................................11

2.1.1  Operational Approaches ..................................................12

2.1.2  Constraint-based Approaches .........................................16

2.2  Difference Constraint ............................................................19

2.2.1  Allen's Logic and Extensions ..........................................19

2.2.2  Difference Constraints Data Form ....................................21

Chapter 3  Multimedia Mail Authoring and Presentation ..................23

3.1  Multimedia Mail Authoring ....................................................23

3.1.1  Media Mail Authoring ...................................................23
5.3.2 SMIL Document Presentation ......................................................... 74
5.3.3 SMIL – Example ............................................................................. 76
5.3.4 Translator ...................................................................................... 78
5.3.5 Composer ....................................................................................... 80
5.4 Experimental Analysis ..................................................................... 81
5.4.1 Multimedia Mail System Authoring ............................................... 81
5.4.2 Multimedia Mail System Presentation .......................................... 83

Chapter 6 Conclusion and Future Work ................................................. 84
6.1 Our Contributions ............................................................................. 84
6.2 Future Work ..................................................................................... 86

Bibliography .......................................................................................... 87
List of Figures

Temporal Structure of A Sample Multimedia Mail ...........................................2
Temporal and Spatial structure of a Sample Multimedia Mail ......................3
Temporal Structure of Example 1.1.a .............................................................4
Spatial Structure of Example 1.1.a ...............................................................4
Time-Line Model of Example 1.1.a ..............................................................13
Firefly's temporal view of Example 1.1.a .....................................................18
Presentation reshaping for different objects duration A ................................30
Presentation reshaping for different objects duration B ...............................30
Presentation reshaping for different objects duration C ...............................30
Negative Cycle in a Constraint Graph .........................................................36
G_M ..............................................................................................................39
Shortest Path Spanning Tree of G .................................................................40
Presentation Schedule before and after modifying ....................................40
Multimedia Mail Filters .................................................................................44
Possible Presentation for the sample mail ..................................................46
Original G ....................................................................................................49
Shortest Path Tree .......................................................................................50
Edge inserted into G ....................................................................................50
Edge Insertion Example T/G ........................................................................52
Shortest Path Tree -T' ................................................................................53
System Structure ..........................................................................................63
Login Screen for Multimedia Mail System ................................................64
List of Tables

Allen’s temporal relations ............................................................... 20
Example data objects ........................................................................ 24
Some multimedia constraints and corresponding specification .......... 27
Chapter 1 Introduction

Electronic mail systems are now widespread in educational establishments and businesses. Text messages can be composed and transmitted to other named users of a network. Mail systems, which allow transmission of composite multimedia messages, have also been investigated. Both the sending and receiving mail applications must have knowledge of message format. The XVmail system developed as part of the Pandora project allows a user to augment standard text-based mail with video and audio. A textual message is sent along with pointers to the related video and audio which are stored on a central service.

1.1 Multimedia Mail with Time and Space requirements

A multimedia mail usually contains a set of media objects, such as video, audio, still-pictures, texts and synthesized images that are to be presented to the user according to a set of temporal, spatial, and alteration specifications. The major difference between a multimedia mail and a traditional text mail is that the former has rich temporal and spatial properties to be satisfied in a presentation.

To create a multimedia mail, an author has to specify which objects to include in the mail and how these objects will be placed in time and space during the presentation. It is also desirable to allow a reader to modify the mail in some form for its presentation. In general, spatial specifications govern the presentation of media objects on the screen and
other output desires, and temporal specifications give the time and duration of presentation of a media object, as well as the synchronization relationships among various objects.

A Sample Multimedia Mail

An example is shown in Figure 1.1. In this example, the mail has four streams of information during the period $<t_0, t_4>$: video, audio, image and text. The spatial organization on the screen is shown Figure 1.2. The spatial structure maps each channel onto independent output devices (e.g. sub-windows, speakers).

![Diagram of four streams labeled Stream 1 to Stream 4 with time points t0, t1, t1, t2, t3, t4 on the x-axis.]

Figure 1.1: Temporal Structure of A Sample Multimedia Mail
1.1.1 Definition of Multimedia Mail

A Multimedia Mail $M$ contains a set of objects $\text{Obj}_M$, such that:

- For each object $O$ in $\text{Obj}_M$, a set of temporal constraints ($T_o$), and a set of spatial constraints ($S_o$) are associated with object $O$.

- An additional set of synchronization constraints relates the temporal occurrences of the start and end events of the presentation of the various objects.

**Example 1.1.a:** $\text{Obj}_M$ includes an audio $O_A$ and a video file $O_V$;

$T_o$: $O_A$ could be played for 2 to 5 seconds, $O_V$ could be displayed for 5 to 10 seconds;

$S_o$: $O_A$ uses the audio channel, and $O_V$ uses the video channel and occupies the whole console window space;

**Synchronization Constraints:**

$O_A$ starts 2 seconds after $O_V$ has ended;
The whole length of presentation time should not be longer than 9 seconds;

A schedule that satisfies the above constraints is shown in Figure 1.3. The space assignment is shown in Figure 1.4.

![Temporal Structure of Example 1.1.a](image)

Figure 1.3: Temporal Structure of Example 1.1.a

![Spatial Structure of Example 1.1.a](image)

Figure 1.4: Spatial structure of Example 1.1.a

1.2 Mail presentation satisfying time and space requirements

1.2.1 Multimedia Mail System Vs. Traditional text-Based Mail System

There are a few differences between multimedia information and traditional text information.
• The semantics of the information in different media are different: both video and audio have temporal dimensions that do not exist in traditional forms of text data. A multimedia mail system, which uses video and audio along with traditional data forms, must provide a means for representing the temporal aspects of data necessary for synchronizing data objects with different temporal requirements. Audio, video, and image data must be managed in such a way that a user can easily query and/or browse the information space as efficiently as possible.

• Due to the larger sizes and streaming characteristics of video and audio data, a multimedia mail system requires more efficient and intelligent information storage and delivery mechanisms, and flexible scheduling to achieve quality presentation.

• Transferring video and audio over a network is different from transferring text: video and audio have higher bandwidth and more stringent delay requirements; they are also more robust in the sense that they may tolerate some fraction of information being lost during the communication. Hence, a multimedia system must use new techniques for benefiting from the underlying network structure. Allowing different users distributed over a network to do collaborative work or receive mail requires creating and sharing of multimedia objects and mail documents.
Hence, most of the techniques developed for text-based environments are inadequate for us.

1.2.2 Analyzing the presentation requirement of existing multimedia system

Multimedia authoring systems will become more attractive to both naive users and experts when they are enhanced with easy-to-use and yet powerful advanced features, such as graphical user interfaces, application independent structures and portability among platforms. Most existing authoring systems still rely on the use of traditional databases, which are well suited for traditional data such as text or numbers, for both their internal data representation/storage and for the database capabilities they provide to the applications. Unfortunately, this makes it difficult for a multimedia authoring system to benefit from the properties of multimedia objects in increasing the storage and retrieval efficiency. The characteristics of multimedia, such as video, are inherently different from those of the traditional data, and they warrant a different approach. In particular, video data has additional attributes including:

1. a temporal extent which cannot be efficiently represented in the relational model,
2. a set of associated events and sub-objects that need to be extracted and indexed for content-based retrieval,
3. quality of service constraints governing the allowable delays in the presentation system, such as for disk access and network communication, and
4. metadata, such as streaming of video.
Some existing authoring systems provide access to open database connectivity (ODBC) technology. This enables a user to attach his database management system to the authoring system. In theory, such systems could be integrated with more advanced multimedia database systems using database specific drivers, thus solving the problem of storage for multimedia. However, there is no uniform query language to retrieve data in different forms. Also, the schedulability of the retrieved multimedia objects in a presentation leads to another issue. A rigid presentation schedule decided at authoring time obviously has many shortcomings. When the reader system cannot accommodate the schedule because of system delays incurred at run time, either poor quality of service will result or the event will have to be abandoned. We address this problem by leaving scheduling (re-scheduling) to be handled at runtime.

The current advances in the network area attract people to use distributed information style. However, the communication characteristics that are applicable to the traditional networks are very different from the characteristics of those that will carry multimedia information. These characteristics need to be fully explored, and efficient protocols need to be developed for providing fast and high quality multimedia communication. Such protocols must address the heterogeneous capabilities and needs of different users, and they need to shape the multimedia object to fit in to the available resources.

1.3 Thesis Contribution

In this thesis, we design a multimedia mail system with real-time scheduling support for authoring and presentation. The major design objectives include flexible
temporal and spatial specification, efficient algorithms to support the incremental consistency checking, presentation schedule generation, and effective editing/filter functions for both authors and readers. We use the term authoring for the composition of a multimedia mail, and presentation for the reading of the mail.

1.3.1 Flexible Multimedia Mail Authoring/Presenting

We use a difference constraint temporal specification for multimedia mail. This approach allows us specify temporal ranges to be used in event scheduling. Specifically, the start time and the duration of an object can be decided independently, given a set of associated temporal constraints.

Both spatial and temporal specifications can be uniformly written in the form of difference constraints. A constraint solver for difference constraints can be used to find a feasible solution for a given set of constraints. Hence both time specification and space allocation can be dealt with uniformly in a single run of the constraint solver. If no such schedule exists, then the requirements given by the user (author/reader) are inconsistent, and the user can be alerted.

1.3.2 Efficient algorithms for solving the consistent difference constraints

Solving a set of difference constraints is equivalent to finding the shortest paths in the graph associated with these constraints [5]. In the constraint graph, inconsistent constraints are captured by the existence of negative cycles. Hence detection of
inconsistency is solvable by detecting the presence of negative cycles in an equivalent constraint graph.

1.3.3 Flexible editing functions for mail readers

A reader may wish to skip viewing certain parts of a mail. In our system, a user is allowed to modify media objects dynamically. The result of the editing operation may invalidate an existing presentation schedule. In such a case, the consistency of constraints must be re-checked and any inconsistency that has been introduced should be removed. We propose a set of standard filter functions made available to a reader for the purpose of altering the presentation of a mail.

1.3.4 Real-time Support

There are two elements of real-time support in the proposed mail system. The first element concerns with the fast interactive editing and dynamic rescheduling of the resulting mail. The mail system is equipped with an incremental re-scheduling algorithm with which a fast response time can be achieved. The second element is embedded in the presentation platform that makes use of the Synchronized Multimedia Integration Language (SMIL) to ensure real-time synchrony in the actual presentation. The SMIL standard enables a user to specify the what, where and when of various media objects in a presentation using a simple and clear markup language.
1.4 Organization of the thesis

Chapter 2 introduces the underlying concepts of the multimedia mail system. This includes the system model and the difference constraints. Chapters 3 and 4 focus on the actual algorithms used in authoring, scheduling and dynamic modification. Chapter 5 describes the system architecture of a prototype multimedia mail system implemented with SMIL. Chapter 6 includes the summary, followed by the descriptions of some key algorithms and the proofs of some results in the appendix.
Chapter 2 Multimedia Mail System Model

Due to the temporal dimension of multimedia mail, building an integrated authoring tool is a challenging task. It is not possible to specify a flexible dynamic behavior and to see its effects immediately. So we must distinguish between the specification phase of the temporal scenario and its presentation phase. An authoring system for multimedia mail must handle these two phases interchangeably so that a mail user can easily move from the specification phase to the presentation phase and vice versa during the course of mail refinement.

2.1 Analyzing Authoring Models

There are two major approaches in authoring: operational or constraint-based. They differ in how close the multimedia mail document description is to the actual presentation. They are briefly reviewed in the following:

- Operational approaches find their roots in finite state machines, which are that is used to capture the temporal scenario of the multimedia document. The author specifies a scenario in a script language or an operational structure. Therefore the presentation resembles closely the operational structure described.
- Constraint-based approaches set the specification outside this operational scheme. They are based on constraint programming and are characterized by a
schedule-generation (formatting) phase that computes the times of occurrence of various presentation events. This formatting phase can be seen as a compilation of a declarative specification into an operational structure that directly maps into a presentation. Thus, an author specifies what scenarios are allowed without specifying the details of a single scenario.

2.1.1 Operational Approaches

We review briefly three variants of the operational approach:

(i) absolute timeline,
(ii) script language, and
(iii) graphical structure.

Temporal axis - Timeline Model

A most intuitive way to specify temporal behaviors is to place events on a single timeline. In doing so, the author specifies the start time and the exact duration the presentation of each object. The corresponding state machine only includes time as input. It does not allow the variability of time and hence the approach leads to rigid schedules.

In such a timeline model, the presentation schedule of all objects is completely specified. Due to its simplicity, this model forms the basis of many multimedia-authoring systems, such as MacroMind Director [51]. Timeline model is suitable for those applications that do not require any flexibility in the temporal specifications and do not
need to be modified once created. We show a sample temporal scenario for a multimedia
mail in the timeline model in Figure 2.1. Rigidity is its main drawback: users must
specify exactly when an event should occur, and the presentation of individual objects is
not allowed to shrink or stretch. It is incapable of handling variations in system delays,
such as communication delay between server and client.

![Timeline Diagram](image)

Figure 2.1. The basic Time-line model graphically describes how
media within a presentation are arranged over time

**Script Languages - MHEG**

Another approach is based on the programming paradigm Lingo[41],
IconAut[42], MHEG[43]. This approach is very expressive. MHEG is a standard for the
coded representation of composite multimedia; components can be related by
synchronization or hyperlinks. It is targeted for interactive multimedia applications.
MHEG involves objects, which are active, autonomous and reusable. Data is represented
as MHEG objects, which are coded with the aim of direct presentation. Internal details
are masked and applications have to deal with only high-level functions.
In MHEG, a temporal behavior can be associated with a composite object by means of a link object that consists of a set of "event & conditions --actions" statements. Hence it is quite expressive. However, several shortcomings exist. First of all, an author is assumed to have programming skill. Second, it is clumsy to describe a temporal placement in a script language. In some languages such as MHEG, making it difficult for an author to visualize the composite scenario.

Figure 2.2 is an MHEG description of the same example shown in Figure 2.1;

Start audio "AUDIO.au";
Wait(10);
Start video "VIDEO1.mpeg";
Wait(20);
Map text "TEXT.doc"
Wait (30);
Unmap text "TEXT";
When end(TEXT)
  Map Image "IMAGE.pic";
Wait(20);
Unmap Image "IMAGE";
When end(VIDEO1);
Wait(10);
Start video "VIDEO2.mpeg";

Figure 2.2 Script Language Example
Graphical Structure - Petri net Model

Graphical structures such as trees or PetriNet provide more global and explicit structures for a user. When time is incorporated in such a structure in the form of delays, it becomes more difficult to analyze.

Petri-net is a well-known parallel computing model. In the context of multimedia authoring, it has been used in OCPN [44] and HTSPN [45] to model temporal scenarios. Objects are modeled by places and temporal information is either associated with places or transitions. In [20], the Petri net model accommodates temporal inexactness, such as when a delay is unknown at authoring time. In that case, it captures temporal precedence, without knowing exact timing separation between presentation events. Little and Ghafoor[44] propose an interval based model using Object Composition Petri Nets (OCPN). It is a variation of the traditional timed Petri net. Two levels of protocol for synchronization of time dependent data are used: Network Synchronization Protocol (NSP), which maintains individual connections, and Application Synchronization Protocol (ASP), which provides support for multimedia documents with OCPN-based temporal specifications [25]. They assume that only one object can be transferred over a link at a time.

Raghavan, Prabhakaran, and Tripathi use probabilistic context free grammars to model the operational behavior of multimedia presentations [27], and to generate a traffic source model based on buffer availability. They show how a probabilistic attribute
context free grammar can capture the operational semantics of OCPN-based temporal specifications and network inputs. However, unlike the time model, the graphical nature of the Petri net model can become complex and difficult to analyze when the document becomes relatively large. In addition, translating a scenario into a graph structure involving places and arcs is not easily acquired skill for an average user.

2.1.2 Constraint-based Approach

The key point in this approach is to use a set of relations to specify differences in the time of occurrences of presentation events. The presentation of an object has an elastic duration that is specified by the author explicitly or implicitly depending on the nature of the object. The elastic time model [6] first introduced in GlueTemporal and Firefly is a pioneering example.

A temporal formatter is used to check the consistency of a given set of temporal relations, and if possible to derive a schedule that satisfies these relations. Because of the use of constraints, a solution space can be searched for an objectively good schedule. Thirteen generic relations are defined in Allen's logic [8]. Modification of the specification is easy as the author can simply add or modify a relation, leaving the issue of consistency and scheduling to the formatter. It is particularly suitable to a dynamic usage environment where the mail can be incrementally modified and viewed. The cost of these advantages is the necessity to provide a schedule generator that chooses a feasible scenario and expresses it in an executable form. A remaining issue is to handle
objects with uncertain parameters. In such a case, temporal consistency check may have to be done on-the-fly, when the parameters become known at runtime. Isis, Firefly and Madeus are three systems that belong to this category. We will use Friefly as an illustration example below.

FireFly System

Buchanan and Zellweger use constraints to specify temporal relationships among multimedia objects within a document in their system called FireFly [5]. Their work is based on the use of difference constraints for the specification of temporal information (we use the same in our system). It is a powerful system that supports and models synchronous and asynchronous behaviors. In Firefly, temporal composition is expressed in terms of relations between event instances (beginning and ending of objects and user interactions). This is the first work in the constraint-based area that has considered the unpredictable nature of some multimedia objects. In order to provide a static formatting process to handle this kind of objects, the Firefly scheduler partitions the temporal scenario at compile-time by grouping connected components. The simplex algorithm is used to find the optimal solution for each partition independently. An event-driven scheduler dynamically handles the integration of the partitions. Unfortunately, its time performance is not good enough in an interactive and incremental context as the compilation time is long. Hence real time scheduling is not attainable. This is the key drawback of Firefly system. Other weaknesses of the Firefly include lack of abstraction and the apparent exponential time required in schedule generation.
Figure 2.3 Firefly's temporal view of the example in Figure 2.1

To sum up,

- Constraint-based approaches seem to be more appropriate for building powerful authoring tools. They offer the same expressiveness as operational techniques. An author need not specify the durations of all the objects involved in his document. Instead, a temporal formatter can take care of such details, provided it is fast enough. Firefly employs a linear programming solver to perform this task. Unpredictable objects, partially handled by Firefly and Madeus, lead to some difficulties in the schedule integration.

- Constraint-based approaches also allow greater flexibility in actual presentations. This is a significant advantage whenever the presentation system must deal with network delays that can affect the timing of object delivery and hence subsequent presentations. A global supervision of timing
allows runtime adjustments of the schedule without violating the original temporal constraints. This is a key point of this thesis.

2.2 Difference Constraint

As mentioned earlier, a major difference between multimedia objects and text objects is the additional temporal semantics in multimedia objects. Hence, much research is directed to the understanding of the temporal behavior of multimedia data and their inter-relationships.

2.2.1 Allen’s Logic and Extensions

Significant advances have been made in the area of temporal specification of multimedia presentations. James F. Allen proposed a formalism based on intervals [8]. In his formalism, Allen differentiates between properties, processes, and events, and describes each of them using time intervals. He defines thirteen qualitative temporal relationships (before, meets, overlaps, during, starts, finishes, equal, etc) that can hold between two intervals, and he shows how these relationships can be used in reasoning about time. More specifically, he provides an algorithm which, given a set of relationships among the intervals in a database, can infer the relationships among all the intervals in it. Allen’s logic is based on a technique called constraint propagation, which basically computes all consequences of a new relation using the transitive closure of temporal relations. Allen proves that his algorithm requires $O(N^2)$ (N stands for the
number of events) time and space, and he also shows how these requirements can be reduced using reference intervals.

<table>
<thead>
<tr>
<th>Multimedia Constraint</th>
<th>Specification</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 a before b</td>
<td>$et(a) - st(b) \leq \varepsilon$</td>
</tr>
</tbody>
</table>
| 2 a equal b           | $st(b) - st(a) \leq 0 \quad st(a) - st(b) \leq 0$  
                          | $et(a) - et(b) \leq 0 \quad et(b) - et(a) \leq 0$ |
| 3 a meets b           | $et(a) - st(b) \leq 0 \quad st(a) - et(b) \leq 0$ |
| 4 a overlaps b        | $st(a) - st(b) \leq \varepsilon \quad st(b) - et(a) \leq \varepsilon$  
                          | $et(a) - et(b) \leq \varepsilon$ |
| 5 a during b          | $st(b) - st(a) \leq \varepsilon \quad et(a) - et(b) \leq \varepsilon$ |
| 6 a starts b          | $st(a) - st(b) \leq 0 \quad st(b) - st(a) \leq 0$  
                          | $et(a) - et(b) \leq \varepsilon$ |
| 7 a finishes b        | $st(b) - st(a) \leq \varepsilon \quad et(a) - et(b) \leq 0$  
                          | $et(b) - et(a) \leq 0$ |

Table 2.1: Allen's temporal relations ($\varepsilon$ is a very small negative number)

However, Allen's work is incomplete for multimedia applications because it lacks the quantitative aspect of time. Users of the multimedia applications are not only interested in the qualitative relationships between two events, but they are also interested in the quantitative relationships between them.
2.2.2 Difference Constraints Data Form

Those spatial, temporal constraints can all be uniformly described within a small class of the language of real valued linear constraints. This class of constraints is referred as difference constraints. Generalized linear constraints have the form

\[ a_1x_1 + a_2x_2 + \ldots + a_nx_n \leq b \]

where \( a_1, \ldots, a_n, b \) are rational numbers (positive and negative), and \( x_1, \ldots, x_n \) range over the real numbers (positive and negative). However, difference constraints have the form

\[ x_1 - x_2 \leq b. \]

Thus, difference constraints are a special case of linear constraints where, there are only two variables (i.e. \( n=2 \) in Equation 1), and one of the variables has coefficient of 1 while the other has coefficient of \(-1\).

Due to the fact that difference constraints have a very tightly restricted syntactic form, it turns out that analysis of such constraints is much easier. As space and time constraints can all be described as difference constraints, this means that a constraint solver for difference constraints may be used to handle space and time constraints within a single unified implementation.

Note that by using difference constraints, not only can we specify Allen’s 13 temporal relationships, but also specify more complex quantitative relationships that cannot be expressed in Allen’s framework. For instance, in Allen’s approach, it is possible to state
that event A occurs before event B. However, it is not possible to say, for instance, that
event A must precede event B by at most 10 seconds and at least 5 seconds. In our system
we can easily specify such temporal separation between two events.

As we mentioned earlier, Buchanan and Zellweger, Kim and Song also used
difference constraints to describe temporal properties of multimedia documents. A
difference between our approach and theirs lies in the fact that they assume that a given
set of constraints is consistent. In our case, we address detection of inconsistency and
potential interactions with the user to resolve inconsistencies dynamically.
Chapter 3 Multimedia Mail Authoring and Presentation

3.1 Multimedia Mail Authoring

The multimedia mail system environment includes a set of users (U), a set of information sources (S), a set of media-objects (ObjM), and a network (N) that connects the users with each other and the sources.

3.1.1 Media object definition

A media object, O in ObjM has four major attributes:

- Name (O) is the name of the object. For simplicity, we will use O as Name(O).
- Type (O) is the type of media object. Each object must be declared as a static, quasi-static or temporal object. These types are described below.
- DispType(O) is the type of display format of object O.
- Attr(O) is a list of special attributes of the object (e.g., size, quality/resolution and length).

Object Type: Every multimedia object has one and only one associated type:

- Static type: static objects like text objects usually consist of a single atomic component. A gif file is an example of a static object.
- Quasi-static type: some objects consist of multiple pages (like postscript files). Each page can be considered an atomic sub-object. However, the display length of each sub-object may vary, at run-time. If $O$ is a quasi-static object, we use the notation $\text{len}(O)$ to denote the number of atomic sub-objects in $O$.

- Temporal type: Some objects contain a predetermined number of atomic components, and a predetermined frequency for the display of these components. Audio objects and fixed-rate video objects are such objects.

The follow table gives three data objects: for the .mpg the presentation time of the video is 2.5 minutes, for the .bmp file the image does not have a pre-set duration, for the .txt file each page must be presented for at least half a minute.

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Size</th>
<th>Users</th>
<th>Frag</th>
<th>Dur</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mpg</td>
<td>a.mpg</td>
<td>180 MB</td>
<td>...</td>
<td>18000 (frames)</td>
<td>${\text{st}(O) - \text{et}(O) \leq -2.5,$ $\text{et}(O) - \text{st}(O) \leq 2.5}$</td>
</tr>
<tr>
<td>Bmp</td>
<td>b.bmp</td>
<td>100KB</td>
<td>...</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>Txt</td>
<td>c.txt</td>
<td>20KB</td>
<td>...</td>
<td>236 pages</td>
<td>${\text{st}<em>{\text{frag}}(1,O) - \text{et}</em>{\text{frag}}(O) \leq -0.5,$ $\text{...},$ $\text{st}<em>{\text{frag}}(236,O) - \text{et}</em>{\text{frag}}(236,O) \leq -0.5}$</td>
</tr>
</tbody>
</table>

Table 3.1: example data objects
3.1.2 Temporal Constraints

In a multimedia mail \( M \), there is a set of temporal constraints \( T(O) \) that involve object \( O \). As we know, constraints are constructed from variables. In the case of our multimedia mail, each multimedia object \( O \) in \( \text{Obj}_M \) in the mail is associated with the following temporal variables.

- \( \text{et}(O) \): denotes the time of the end event of object \( O \)
- \( \text{st}(O) \): denotes the time of the start event of object \( O \)
- \( \text{st}_i(O) \): denotes the time of the start event of the \( i^{th} \) component of object \( O \)
  
  (For quasi-static or temporal object)
- \( \text{et}_i(O) \): denotes the end event of the \( i^{th} \) component of object \( O \)
  
  (For quasi-static or temporal object)

Each constraint is of the form \( T_1 - T_2 \leq d_{21} \)

where

- \( T_1, T_2 \in \{ \text{st}(o_j), \text{et}(o_j), \text{et}_2(o_j), \ldots, \text{st}_{\text{len}(o_j)}(o_j), \text{et}_{\text{len}(o_j)}(o_j), \text{et}(o_j) \} \cup \{ \text{st}_p, \text{et}_p \} \)

  (\( j = 1, \ldots, k; \) len stands for the total number of components of ObjectO)
- \( \text{st}_p, \text{et}_p \) denote the start and end of the presentation respectively
- \( d_{21} \) is a constant

**Example 3.1.2.a:** There are three objects in the sample mail; \( O_v \) is a video clip, \( O_t \) is a text, and \( O_a \) is an audio clip.

The temporal constraints \( T(O) \) are specified by:

- Within an object: (Intra-object constraint):
\[ \text{st}(O_v) - \text{et}(O_v) \leq -5 \quad // \text{the video will play for at least 5 time unit s} \]

- **Between objects:** (Inter-object constraint)

  \[ \text{et}(O_v) - \text{et}(O_a) = -5 \quad // \text{the video starts 5 time units ahead of the end of the text} \]

  \[ \text{st}(O_v) - \text{st}(O_a) = 0 \quad // \text{the video must start at the same time as the audio} \]

We can quantify Allen's relations and transform them into the difference equation model. The following example in Table 3.2 is an illustration.
<table>
<thead>
<tr>
<th></th>
<th>Multimedia Constraint</th>
<th>Specification</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>a should start when b starts</td>
<td>st(a) - st(b) ≤ 0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>st(b) - st(a) ≤ 0</td>
</tr>
<tr>
<td>2</td>
<td>a should start 2 seconds after b ends</td>
<td>et(b) - st(a) ≤ 2</td>
</tr>
<tr>
<td></td>
<td></td>
<td>st(a) - et(b) ≤ -2</td>
</tr>
<tr>
<td>3</td>
<td>a should start 2 seconds before the end of the presentation</td>
<td>et_p - st(a) ≤ 2</td>
</tr>
<tr>
<td></td>
<td></td>
<td>st(a) - et_p ≤ -2</td>
</tr>
<tr>
<td>4</td>
<td>a should start within 3 seconds after the start of the $\gamma^{th}$ frame of object b</td>
<td>et(a) - st_7(b) ≤ 3</td>
</tr>
<tr>
<td></td>
<td></td>
<td>st_7(b) - st(a) ≤ 0</td>
</tr>
<tr>
<td>5</td>
<td>a should end within 2 seconds of the start of the $\gamma^{th}$ frame of the object b</td>
<td>et(a) - st_7(b) ≤ 2</td>
</tr>
<tr>
<td></td>
<td></td>
<td>st_7(b) - st(a) ≤ 2</td>
</tr>
<tr>
<td>6</td>
<td>a should be presented for 7 seconds</td>
<td>et(a) - st(a) ≤ 7</td>
</tr>
<tr>
<td></td>
<td></td>
<td>st(a) - et(a) ≤ 0</td>
</tr>
<tr>
<td>7</td>
<td>The 2$^{nd}$ frame of a should start when the 5$^{th}$ frame of b ends</td>
<td>st_2(a) - et_5(a) ≤ 0</td>
</tr>
<tr>
<td></td>
<td></td>
<td>et_7(a) - st_2(a) ≤ 0</td>
</tr>
</tbody>
</table>

Table 3.2: some multimedia constraints and the corresponding specifications
Implicit Temporal Constraints

Some implicit constraints exist by default. They arise as a consequence of the semantics of the start and end of an object. The end event of an object must occur after its start. For example, we have the following set of implicit constraints:

\[ st(O) - et(O) \leq 0 \quad st_p - st(O) \leq 0 \quad et(O) - st_p \leq 0 \]

For each temporal or quasi-static multimedia object \( a \), we have the following two constraints:

\[ et(a) - et_{len(a)}(a) = 0 \quad st(a) - st_{len(a)}(a) = 0 \]

These constraints merely specify that the presentation of a quasi-static or temporal object begins (ends) when its first (last) atomic sub-object’s display starts (ends).

In our mail system, all of the above constraints are assumed without explicit specification.

**Example 3.1.2.b**: Suppose there are three multimedia objects, \( O_1 \), \( O_2 \), \( O_3 \), with the following details:

a. \( O_1 \) should be activated within the 60 seconds of the presentation

b. \( O_2 \) should be activated within the last thirty seconds of the presentation of \( O_1 \)

c. \( O_3 \) should be activated along with object \( O_1 \)

d. \( O_3 \) and \( O_2 \) should finish at the same time
e. The duration of $O_1$ is 90 seconds
f. $O_1$ has a maximum time period of 150 seconds
g. The duration of $O_2$ is 120 seconds
h. The duration of $O_3$ is un-constrained
i. $O_3$ should be presented for at least 150 seconds

The constraints associated with the above are translated into the following:

a. $st(O_1) - st_p \leq 60$

b. $et(O_1) - st(O_2) \leq 30$ and $st(O_2) - et(O_1) \leq 0$

c. $st(O_3) - st(O_1) \leq 0$ and $st(O_1) - st(O_3) \leq 0$

d. $et(O_3) - et(O_1) \leq 0$ and $et(O_2) - et(O_3) \leq 0$

e. $et(O_1) - st(O_1) \leq 90$ and $st(O_1) - et(O_1) \leq -90$

f. $et(O_1) - st(O_1) \leq 150$

g. $et(O_2) - st(O_2) \leq 120$ and $st(O_2) - et(O_2) \leq -120$

h. no constraints

i. $st(O_2) - et(O_2) \leq -150$

Figure 3.1 shows a solution of these constraints with respect to different object durations. As can be seen, the presentation shapes itself according to the length of the objects. Our later system implementation will pick up the presentation schedule with the shortest presentation time according to our presentation-scheduling algorithm (b).
3.1.3 Spatial Constraints

Spatial constraints are defined for all objects in a multimedia presentation, which can be displayed on the screen or other output channels. In order to specify spatial constraints
each object has its channel attribute, which indicates what spatial channel it belongs to.

We use the following spatial variables for a display on the console window:

- $w$ and $h$ denote the width and height, respectively, of the multimedia mail
- $X_l$ and $Y_b$ denote the coordinates of the lower left corner of the multimedia media mail on the screen
- $X_r$ and $Y_t$ denote the coordinates of the upper right corner of the multimedia mail on the screen
- $X_r(O)$ and $X_l(O)$ denote the relative position of the right and left borders of the multimedia object with respect to $X_l$
- $Y_b(O)$ and $Y_t(O)$ denote the relative position of the bottom and top borders of the multimedia objects with respect to $Y_b$

Similar to temporal constraints, spatial constraints are of the following two forms:

$$X_1 - X_2 \leq \delta x, \quad Y_1 - Y_2 \leq \delta y,$$

where:

1. $X_1, X_2 \in \cup_j \{ X_r(O_j), X_l(O_j) \} \cup \{ X_l, X_r \},$

2. $Y_1, Y_2 \cup \cup_j \{ Y_r(O_j), Y_l(O_j) \} \cup \{ Y_l, Y_r \}$, and

3. $\delta x, \delta y$ are two constants.

Example 3.1.2.c: Suppose a multimedia mail contains two object $O_1$ and $O_2$. The left border of $O_1$ is 100 pixels from the left border of the mail, and the left border of $O_2$ is at most 10 pixels from the right border of $O_1$. These can be defined as:

$$X_l(O_1) - 100 \leq 0, \quad 100 - X_l(O_1) \leq 0,$$
\[ X_i(O_1) - X_i(O_2) \leq 0, \quad X_i(O_2) - X_i(O_1) \leq 10, \]

**Implicit Spatial Constraints**

For all multimedia mail \( m \) and for all objects \( O \) the following constraints must be implicitly satisfied. As in the case of temporal constraints, there are certain implicit spatial constraints.

\[
\begin{align*}
X_r - X_t &\leq 0 & Y_b - Y_i &\leq 0 \\
X_r - X_t &\leq \omega & X_r - X_t &\leq -\omega \\
Y_b - Y_i &\leq h & Y_t - Y_b &\leq -h \\
X_i(o) - X_r(o) &\leq 0 & Y_b(o) - Y_i(o) &\leq 0 \\
X_r - X_i(o) &\leq 0 & X_i(o) - X_r &\leq 0 \\
Y_b - Y_b(o) &\leq 0 & Y_i(o) - Y_i &\leq 0
\end{align*}
\]

### 3.2 Multimedia Mail Presentation Scheduling

A multimedia mail consists of media objects that are to be sequenced and presented according to temporal and spatial specifications at different time instances and for different time durations. The presentation of the multimedia objects has to be carried out in accordance with the specification of temporal and spatial relationships between the objects defined by users.
Suppose $M$ is a multimedia mail containing a set of objects $Obj_M$ with temporal constraint $T_M$ and spatial constraint $S_M$. The temporal constraints can be visualized as a labeled directed graph $G = (V, E, L)$ where:

- $V = \{v_i | i = 1, \ldots, n\}$ is the set of events (times),
- For each difference equation, say $v_i - v_j = d_{ij}$, we associate an edge $(v_i, v_j)$ in $E$, and $d_{ij}$ in $L$.

For convenience, we use $v_i$ and $v_j$ to represent $(v_i, v_j)$ in $E$ (instead of $st(O_i)$, $et(O_i)$ in authoring part).

Suppose $v_i$ is the time of occurrence of event $t_i$. Then the following holds in the graph $G$: $d_{ij}$ in $L$ iff $t_j - t_i = d_{ij}$

**Example 3.2.a:** Suppose $t_i$ and $t_j$ are the start and end events of a video clip and $d_{ij} = -5$. Then this means the duration of the video clip must be at least 5 time unit.

$$t_i - t_j = -5$$

**Example 3.2.b** $d_{ij} = 4$, $t_j$ is the start of an audio clip and $t_i$ is the start of a text display. Then this means the audio clip must start no later than 4 seconds after the start of the text display. $t_j - t_i = 4$
Example 3.2.c: Suppose $v_i$ and $v_j$ are the times of the start and end events of a video clip respectively; $v_k$ and $v_l$ are the times of the start and end events of a text display; $v_m$ and $v_n$ are the times of the start and end events of an audio clip.

$v_i - v_j \leq 5$  // the video clip should be displayed for at least 5 seconds

$v_m - v_i \leq 5$  // the video must start no later than 5 seconds after the end of the text presentation.

$v_m - v_i \leq 0$  // the start time of audio should be not less than or equal to the video file

Many researchers have studied scheduling. In [12], Geerber addresses the problem of scheduling with relative time constraints. He identifies the set of difference constraints as a special case (restricted standard constraints). On the other hand, he assumes that a given set of relative timing constraints is consistent at all times. In [7], Little describes a method that adapts the presentation schedule to the changes in the resource availability by modifying the overall QoS of the presentation. In [26], Moser also suggests a gracefully degrading quality of service management framework. His system benefits from the alternative implementations of a multimedia service to find the best possible match to a given request. In this thesis, quality of service is not part of our scheduling model.

3.2.1 Formulation of the Presentation Scheduling Problem

The difference constraints specified by an author have to be analyzed in order to decide their consistency and schedulability. The analysis is performed by first
transforming the equational constraints into a labeled directed graph, say \( G \), whose
vertices correspond to events and labeled edges correspond to timing separation between
conforming events. In particular, for a constraint of the form \( t_i - t_j \leq d_{ji} \), an edge leads
from vertex \( t_j \) to vertex \( t_i \) with a distance label of \( d_{ji} \). Then consistency and scheduling
will be solved directly using \( G \).

**Example 3.2.1.a:** Consider the following set of constraints:

\[
\begin{align*}
  a - b & \leq 0 \\
  b - a & \leq -1 \\
  c - a & \leq 0 \\
  b - a & \leq -1
\end{align*}
\]

The corresponding constraint graph \( G \) contains three vertices and four edges and
is shown in Figure 3.2.

The consistency of a given set of constraints can be ascertained by checking if the
corresponding constraint graph \( G \) contains negative cycles. The existence of a negative
cycle infers that a cyclic sequence of events have temporal precedence that cannot be
satisfied. The edges traversed in the cycle correspond to a subset of the constraints whose
sum will lead to \( 0 \leq \) a negative value, which is obviously an impossibility. Hence
consistency checking is reduced to checking for the absence of negative cycles in \( G \).

**Example 3.2.1.b:** Assume the following set of constraints

1). \( a - b \leq 0 \)  2). \( b - a \leq -1 \)

3). \( c - a \leq 0 \)  4). \( b - c \leq -1 \)
Here, equations 1) and 2), and equation 1), 3) and 4) form inconsistent subsets: the graph $G_m$, as shown in figure 3.2, contains two negative cycles.

![Figure 3.2 Negative Cycle in a Constraint Graph](image)

We formulate the scheduling problem in the form of a shortest path problem. We first augment the graph $G$ with an end vertex, say $v_0$ and edges $\{v_0, v_i \mid i = 1,...n\}$ with labels $\{d_{0i} = 0 \mid i = 1,...n\}$, the vertex $v_0$ represents the end of a presentation. Hence, $d_{0i} = 0$ means that $v_i$ must occur not later than $v_0$.

Let $d_i$ be the shortest distance from $v_0$ to $v_i$, $i = 1,...,n$. Suppose $G$ does not contain negative cycles.

**Theorem 3.2:** A feasible solution exists iff $G$ does not contain negative cycle, in which case $\{t_i = d_i \mid i = 1,...n\}$ is a feasible solution to the scheduling problem.

**Proof:** Suppose a negative cycle exists in $G$, say $v_1 \rightarrow v_2 \rightarrow ... \rightarrow v_1$, and a feasible solution exists. From the formulation in previous section, this cycle
corresponds to a set of constraints whose sum leads to \( 0 < 0 \), which is an obvious contradiction.

Next, consider the case \( G \) without negative cycles. Take \( \{t_i = d_i \mid i = 1, \ldots, n\} \) as solution. We will show that this is feasible: suppose for some \( i, j \), we violate the constraint, say \( d_j - d_i > d_{ij} \). Equivalently, we have \( d_j > d_i + d_{ij} \). This implies \( d_j \) cannot be the shortest distance from \( v_0 \) to \( v_j \), which could be reduced to \( d_i + d_{ij} \). Hence we have a contradiction, and the constraints must all be satisfied by the solution. QED

Along the same vein, spatial constraints can be solved in an analogous way, using space instead of time as dimension. Hence, the same scheduling algorithm can be used for both time and space allocation. In the sequel, we will focus only on time allocation.

### 3.2.2 Finding Single Source Shortest Paths Algorithm

In this section, we describe the solution to the following problem: “Given a set of difference constraints associated with a multimedia mail, derive a solution that satisfies the constraints if one exists.”

In the constraint graph, conflicting constraints are reflected by the existence of negative cycles. Though many shortest path algorithms cannot handle negative cycles, the well-known Bellman-Ford shortest path algorithm does not assume the absence of negative cycles. We will adapt the Bellman-Ford algorithm to solve our scheduling problem. Specifically, if there is no negative cycle, the algorithm produces the shortest
paths and their weights. If there is such a negative cycle, the algorithm terminates with an “error” indication.

From the above theorem, there are two sub-problems to be solved: (i) the existence or absence of negative cycles in \( G \), and (ii) the set of shortest distances \( \{d_i | i = 1, \ldots, n\} \). We adapt the Bellman-Ford shortest path (BF) algorithm to solve (ii) and show that the resulting algorithm also solves (i).

**Algorithm BF \((G_M = (V,E,L))\);**

Initialize all \( d_i \) \((i = 1, \ldots, n)\) to 9999 (infinity);

\(\text{count} := 0;\)

\(\text{repeat}\)

\(\text{changed} := \text{false};\)

\(\text{for each} \ (v_i,v_j) \text{ in E do}\)

\(\text{if} \ d_j < d_i + d_{ij} \text{ then} \{\ d_j := d_i + d_{ij};\)

\(\text{changed} := \text{true};\}\)

\(\text{end_for};\)

\(\text{increment count};\)

\(\text{until} \ \text{count} = |V| + 1 \text{ or changed} = \text{false};\)

\(\text{if} \ \text{changed} = \text{true} \text{ that report 'failure' else return} \ \{d_i \ | \ i = 1, \ldots, n\};\)

\(\text{end.}\)
The above algorithm is directly adapted from the Bellman-Ford algorithm with a slight modification. The algorithm runs through \(|V| + 1\) iterations. In each of the iterations, the shortest distances are updated by considering every edge in \(E\). Hence the values computed in iteration \(i\) would be no more than the shortest distance obtainable by traversing up to \(i\) edges in \(G\), starting from \(v_0\). Iteration \(|V| + 1\) should not change the distance; otherwise \(G\) must have contained some negative cycle. The complexity of the algorithm is given by \(O(V^2E)\).

If \(G_M\) is consistent, after the application of the BF algorithm, the shortest distances from \(V_0\) to \(\{V_i\}\) will lead to a subset of the edges of \(G_M\), which together will form a spanning tree \(T_M\) of \(G_M\). We call this the shortest path-spanning tree. It is obvious that it is a tree because

(i) \(V_0\) can reach every \(V_i\), and

(ii) There is only one path from \(V_0\) to each \(V_i\).

**Example 3.2.2.a:** (Suppose the edges are chosen from top-to-bottom, and left-to-right in the above diagram in each iteration):

![Figure 3.3 $G_M$](image-url)
<table>
<thead>
<tr>
<th>Iteration</th>
<th>$d_0$</th>
<th>$d_1$</th>
<th>$d_2$</th>
<th>$d_3$</th>
<th>$d_4$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>999</td>
<td>999</td>
<td>999</td>
<td>999</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>-3</td>
<td>-2</td>
<td>0</td>
<td>-3</td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>-3</td>
<td>-5</td>
<td>0</td>
<td>-4</td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td>-3</td>
<td>-5</td>
<td>0</td>
<td>-4</td>
</tr>
</tbody>
</table>

no change → stop

If $G_M$ is not consistent, for example, $T_{12}$ is changed from $-2$ to $-3$ in the given $G_M$, then the algorithm terminates with a failure message. The following table illustrates the corresponding scenario:

![Figure 3.4 Shortest Path Spanning Tree of G](image-url)
<table>
<thead>
<tr>
<th>Iteration</th>
<th>$d_0$</th>
<th>$d_1$</th>
<th>$d_2$</th>
<th>$d_3$</th>
<th>$d_4$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>999</td>
<td>999</td>
<td>999</td>
<td>999</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>-3</td>
<td>-3</td>
<td>0</td>
<td>-3</td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>-3</td>
<td>-6</td>
<td>-1</td>
<td>-5</td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td>-4</td>
<td>-7</td>
<td>-2</td>
<td>-6</td>
</tr>
<tr>
<td>4</td>
<td>0</td>
<td>-5</td>
<td>-8</td>
<td>-3</td>
<td>-7</td>
</tr>
<tr>
<td>5</td>
<td>0</td>
<td>-6</td>
<td>-9</td>
<td>-4</td>
<td>-8</td>
</tr>
<tr>
<td>6</td>
<td>0</td>
<td>-7</td>
<td>-10</td>
<td>-5</td>
<td>-9</td>
</tr>
</tbody>
</table>

Change in iteration 6 → presence of cycle.
Chapter 4  Incremental Modifications

Easy-to-use tools for creating, manipulating, and presenting multimedia content, are important facilities in the creation of a multimedia mail system for normal users. A modification facility is also an important functionality to take into account when designing an authoring environment, since building an interactive multimedia mail is a "specify, test and modify" process.

A user could incrementally specify his modification in constraint form. The system processes that change made through a GUI and then returns either a valid new schedule or a message indicating that the changes lead to inconsistency.

Example 4.0.a: The user defines two media objects O₂, O₃ in a mail as follows (the reference starting time of the whole presentation is stp):

- O₃ starts at time 0
- O₂ starts no earlier than 50 seconds and no later than 70 seconds after O₃ has started
- O₂ and O₃ must end together

The system translates these specifications into the following set of difference constraints:

\[ stp - s_t(O_3) \leq 0, \quad s_t(O_3) - stp \leq 0 \quad s_t(O_3) - s_t(O_2) \leq 70 \]

\[ 50 \leq s_t(O_3) - s_t(O_2) \quad e_t(O_3) - e_t(O_2) \leq 0 \quad e_t(O_2) - e_t(O_3) \leq 0, \]
A presentation schedule is computed and shown as schedule A in Figure 4.1.a.

Next, suppose the user wants to add $O_1$ to the presentation with the following specification:

- $O_1$ starts at the same time as $O_3$;
- $O_2$ ends no earlier than 40 time units and no later than 60 time units after $O_2$ has started

The additional constraints are,

$$st(O_1)-st(O_3) \leq 0, \quad st(O_3)-st(O_1) \leq 0 \quad et(O_1)-st(O_2) \leq 50$$

Subsequently, a new presentation schedule is computed by the system, and shown in Figure 4.1b

![Figure 4.1 Presentation Schedule before and after modifying](image-url)
4.1 Filters for accessing the Multimedia Mail

Depending on their interests and needs, users can view a mail in different ways through the use of filters. These filters describe the specific portions of the mail that can be viewed or edited by a user.

Figure 4.2 shows the structure of the presentation system environment. In particular, the presentation shown in Figure 4.1 may be presented in two different ways to two different users, depending upon their interests, their access rights, and the local system capabilities.

```
+----------------+                        +----------------+                        +----------------+
| Objects        |                        | Objects        |                        | Objects        |
| Constraints    |                        | Constraints    |                        | Constraints    |
+----------------+                        +----------------+                        +----------------+
                  +----------------+                        +----------------+                        +----------------+
                  | Access Rights   |                        | Communication   |                        | Local capabilities |
                  +----------------+                        +----------------+                        +----------------+
                  +----------------+                        +----------------+                        +----------------+
                  | Filter 1        |                        | Filter 2        |                        | Filter 3        |
                  +----------------+                        +----------------+                        +----------------+
                  +----------------+                        +----------------+                        +----------------+
                  | Filter 4        |                        | Filter 4        |                        | Filter 4        |
                  +----------------+                        +----------------+                        +----------------+

User1-Author +----------------+                        +----------------+                        +----------------+ User2-Reader1 +----------------+                        +----------------+ User3-Reader2
```

Figure 4.2: Multimedia Mail Filters

Hence, we try to use some filters for each form of presentation based on the following factors:
• Filter: User Access Rights

Access rights describe the portions of the multimedia mail accessible to the user. An author has the full access right of his own mail, and at the same time, he could define different access levels to different mail recipients. For example, the author sends a mail to both Reader\textsubscript{A} and Reader\textsubscript{B} in which he hides Object\textsubscript{A} from Reader\textsubscript{B}, and Object\textsubscript{C} from Reader\textsubscript{A}. The specification of access rights is part of the creation of the mail by the author;

• Filter: Communication and Quality of Service Requirement

Because of network delay, an object may not be delivered to the reader's site in time. In that case, the system may omit the object and re-schedule the rest of the presentation.

• Filter: Local Presentation Capabilities

The local system capabilities describe the facilities a user has for handling multimedia objects. For example, if the local system cannot handle MPEG video streams, the user might filter out MPEG video objects and view the rest of the mail.

• Filter: Reader Modification

The "projection" enables a reader to access different portions of the document that he is interested in. The presentation adapts itself to the changes, such as omissions or delay of objects. A reader can invoke certain edit functions defined by the author that
modify only the local view of the multimedia mail. When the reader has chosen the desired edit functions, the changes can be realized in the view of the mail.

**Example 4.1.a:** An author defines a multimedia mail consisting of three multimedia objects, O₁, O₂, O₃ targeting three readers, R₁, R₂ and R₃.

- O₂ should display after O₁ finishes
- O₃ must start at the same time as O₂
- O₂ is hidden from R₁

In viewing the mail,

- R₁ takes 3 seconds to get O₁ through network delay, 5 seconds to get O₂ and O₃
- R₂ gets access to O₂, O₃ immediately, 1 second to get O₁, and he wishes to start the presentation of O₃ within 8 second of the beginning of the presentation
- R₃ gets the access to O₂, O₃ immediately, 1 second to get O₁, but he does not want to see O₂.

Figures 4.3 shows some possible presentations of the multimedia mail.

![Diagram](image)

**Figure 4.3 Possible Presentation for the sample mail**
For R₃, the O₂ is omitted from the presentation and all related constraints are suppressed. In this case, the display of O₃ overlaps O₁;

### 4.2 Incremental Modifications of Constraints

With each object O ∈ Objᵢ in a multimedia mail (M), we associate a set, T, of temporal constraints and a set, S, of spatial constraints. When changes occur due to the addition/deletion of objects, these are captured within the existing presentation schedule for the multimedia mail as changes to the constraints governing the presentation of those objects.

For example, when a new object O₃ is added to a mail and we want to present O₃ immediately after an existing object O₁ and immediately before another existing object O₂, then the corresponding temporal constraints involving this object may affect the presentation schedule. The effect of such a modification is captured by the changes in the set of constraints to be used.

#### 4.2.1 Incremental Scheduling Algorithm

The easiest way to handle the addition/deletion of a constraint would be to use the shortest path algorithm described earlier from scratch. However, in a multimedia system where there are many dynamic changes, or in a system where there are hard deadlines for the presentation, this may not be the best approach. Hence, we present an algorithm, which dynamically modifies the presentation schedule according to given an incremental
change by the users. The use of such an algorithm is crucial to support runtime scheduling. Recall that the insertion/deletion of a constraint into a constraint set is equivalent to the insertion/deletion of an edge in the corresponding temporal graph.

The input to the algorithm includes

(i) the temporal constraint graph, G, and

(ii) a shortest path spanning tree T of G as derived earlier from algorithm BF

(iii) The new edge (constraint) $E' = (e_i, e_j)$ to be inserted/deleted in G with temporal weight $T_{ij}$

In this section, we describe an incremental scheduling algorithm that modifies the shortest path spanning tree when G is augmented with a new edge (constraint), say $(e_i, e_k)$ with $d_{ik} = 0$.

The incremental algorithm is to update the spanning tree by propagating the effects of the new edge. The propagation either ends when no $d_i$ (the time scheduled I) is affected any more, or when a negative cycle is detected, reflected by changing some $d_i$ more than once. The complexity is hence given by $O(E)$.

Algorithm Edge_insertion ($G$, $T$, $d_{ik}$);

$T$ is a shortest path-spanning tree of $G$ as derived from BF

$$E' := \{(v_i, v_j)\};$$

repeat
choose an edge in \( E' \), say \((v_i, v_j)\), and remove it from \( E' \);

if \( d_i + d_{ij} < d_j \) then

\{
\[ d_j := d_i + d_{ij}; \]

if \( v_j.\text{visited} \) then report cycle and return;

update \( T \) with \((v_i, v_j)\);

\( v_j.\text{visited} := \text{true}; \)

insert all edges emanating from \( v_j \) into \( E' \); \}

until \( E' = \text{empty}; \)

Let us take a look at the previous example 3.2.2.a again,

\[
\begin{align*}
\text{Figure 4.3.a Original } G \\
\end{align*}
\]

With shortest path shown as in Figure4.3.b
Example 4.2.1.a: Suppose we want to insert a constraint between $e_1,e_4$ with $t_{14} = 0$, (see figure 4.3.c below);

Figure 4.3.c Edge inserted into G

Figure 4.3.d: Edge Insertion Example $T/G$
Algorithm Edge_Insertion;

Take an edge \((V_0, V_4)\) in \(E'\)
\[d_1 = -3 \quad d_{14} = 0 \quad d_4 = -3,\]
\[d_1 + d_{14} < d_4\]
so \(d_1 + d_{14} < d_4\), according to the algorithm, \(d_4 := d_1 + d_{14} = -3\), update \(T\),

The algorithm terminates in one iteration of the repeat loop: \(d_4\) is unaffected and the old \(T\) remains the shortest:

<table>
<thead>
<tr>
<th>Iteration</th>
<th>(d_0)</th>
<th>(d_1)</th>
<th>(d_2)</th>
<th>(d_3)</th>
<th>(d_4)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>999</td>
<td>999</td>
<td>999</td>
<td>999</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>-3</td>
<td>-2</td>
<td>0</td>
<td>-3</td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>-3</td>
<td>-2</td>
<td>0</td>
<td>-3</td>
</tr>
</tbody>
</table>

no change → stop

![Figure 4.3.b Shortest Path Tree - T of Example](image)

For the deletion of an edge in a constraint graph, the following algorithm can be used.
Algorithm Edge deletion \((G, T, d_m)\);

if \((v_l, v_m)\) is not in \(T\) then return;

\(T' := \) subtree of \(T\) rooted at \(v_m\) and \(V' =\) set of vertices in \(T'\);

For each vertex \(v_j\) in \(V'\) initialize \(d_j\) with the \(\min \{d_k + d_{kj}\}\) where \(d_k\) is a vertex in \(V-V'\).

Apply BF algorithm on \(G/V'\) (subgraph of \(G\) containing nodes in \(V'\)) with the initial distances given by \(\{d_i\}\);

return;

The above algorithm reduces the work by restricting the iteration of BF to the subtree that is affected, while the other subtree given by \(T-T'\) is guaranteed not affected because of the optimality: deletion of an edge will not reduce the shorted distance in every vertex in \(V-V'\).

Example 4.2.1.b: Consider the deletion of \(d_{12}\) (which is equivalent to resetting \(d_{12}\) to 9999) in the example of previous chapter.
According to the initial value in previous computed schedule,

<table>
<thead>
<tr>
<th>Iteration</th>
<th>$d_0$</th>
<th>$d_1$</th>
<th>$d_2$</th>
<th>$d_3$</th>
<th>$d_4$</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>0</td>
<td>-3</td>
<td>-5</td>
<td>0</td>
<td>-4</td>
</tr>
</tbody>
</table>

This gives rise to the following sub-graph:

![Graph diagram](image)

$T'$ rooted at $V_2$, and $V' = \{V_2, V_4\}$;

Set $d_2 = 0$, $d_{24} = 1$, $d_4 = -4$ (initial value), so, update $d_4'$ with $\min\{d_4, d_{24}\} = -4$;

<table>
<thead>
<tr>
<th>Iteration</th>
<th>$d_2$</th>
<th>$d_4$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>-4</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>-4</td>
</tr>
</tbody>
</table>

and this stops the iteration of BF.

### 4.3 Incremental Consistency-Check and Re-Scheduling

The feasibility of a presentation schedule can be checked against changes of temporal constraints. If all constraints remain consistent, the changes can be accepted. If infeasibility is encountered, and the reader possesses the modification-right, the system will provide intelligent feedback to the reader by indicating the bad consequences of the present change of constraint. The reader may choose to accept the inconsistency
introduced in viewing the mail based on the schedule generated by either Edge_Insertion or Edge_Deletion.

Since we introduce the dynamic user modification in our mail system, an important aspect of the practical use of a consistency-checking algorithm is its incrementality. Such an algorithm can be used to process one constraint at a time while retaining all previously encountered constraints. Our solution actually seeks the shortest presentation time of the given constraints.

4.3.1 Author Interface

The author interface includes a temporal specifier. In particular, the author is allowed to enter the mail specification in equation form in the GUI incrementally. During the course of composition, he can introduce new objects (hence events) and associated constraints incrementally. The system can invoke at any time the scheduling algorithms presented earlier, depending on the scenario. The system is able to provide a consistency check and return a new schedule in each invocation or identify an inconsistency.

4.3.2 Reader Interface

The reader interface supports a different set of editing tools to facilitate the reading of a mail. We distinguish what a reader is allowed to do during a reader session from what an author is allowed to do. While the author behavior is conceptually
unrestricted (an author can change the document specification completely), the set of operators made available to the readers is more abstract and a bit restrictive, and a reader tailors them to reasonable uses. These include:

(i) Object omission
(ii) Object migration
(iii) Object expansion
(iv) Object contraction

In each of these operations, we must also define their semantics so that they are consistent with the original intent of the specification.

Suppose \( v_i \) and \( v_j \) are the start and end events of the presentation of the object to be omitted, and \( t_i \) and \( t_j \) are the corresponding times.

**Analysis 1:** **OD** - An object is to be omitted from the presentation.

There are several possibilities in omitting an object and their semantics are explored below:

- **Blanking omission:** This leaves the presentation of the object blank but otherwise does not change the schedule. Blanking omission does not require any rescheduling.

It involves only the "blanking" of the parts of the presentation involving the object. This is always possible and easily supported.
- **Skipping omission:** This replaces the constraint involving \( t_i \) and \( t_j \) by \( t_i = t_j \). Skipping omission removes the object completely from the schedule by shrinking it to a zero-time component. Such a skip is sometimes infeasible.

- **Adaptive omission:** This replaces the constraints involving \( t_i \) and \( t_j \) with \( t_j - t_i \geq 0 \) and \( t_i - t_j \geq -\infty \), with the understanding that the object will actually involve a "blank" presentation if \( t_j > t_i \). Adaptive omission gives the scheduler the flexibility of retiming \( T \), leading to a solution possibly somewhere between blanking and skipping omissions. It brings flexibility to accommodate other changes invoked by the reader.

**Analysis 2:** **OE/OC** - An object is to be expanded/contracted in the presentation.

**Semantics:** This amounts to modifying the presentation time (or duration) of the object concerned by \( t_j - t_i \geq c \) to \( c' \) or \( t_i - t_j \geq c \) to \( c' \). The scheduler provides the reader the range of values of \( c' \) that is allowed under the existing presentation schedule.

**Analysis 3:** **OM** - An object is to be migrated in time.

**Semantics:** The start-time of an object, say \( t_i \), is moved to \( t_i' \), and by the same amount, the end-time of an object is also moved. This migration is possible provided the resulting schedule remains consistent with the given constraints provided by the author. The user-interface provides a table of values that specify the range of migration allowed.

All of the above can be supported under the three algorithms presented in Chapter 3.
For OD - Object Deletion, simply replace those constraints involved with \( t_i, t_j \) with the corresponding values. In Skipping Omission, replace \( d_i = t_i - d_j = t_j \), and re-assign \( d_{ij} = 0 \) in \( E \), it is equivalent to deletion of a constraint;

For OE/OC - Object Expansion/Contraction, by reassigning \( d_{ji} = t_j - t_i \geq c \) to \( c' \) or \( d_{ij} = t_i - t_j \geq c \) to \( c' \) in \( E \), then applying Algorithm BF;

For OM - Object Migration, by changing the value of \( d_i \) from \( t_i \) to \( t_i' \) and re-applying those three algorithms.

4.3.3 Computation of Modification Tables for Reader Interface.

**Definition:** Given a schedule \( \{d_i \mid i = 1, \ldots, k\} \), the maximal change \( s_i \) to \( d_i \) (to become \( d_i + s_i \) without changing other \( d_j 's \)) while retaining consistency, is the slack of \( d_i \).

The value of \( s_i \) can be computed from the given constraints:

- **Static Increase (delay) of a single event:**
  
  Suppose \( s_{ij} = d_j - d_i \). From the given difference constraint, we know \( s_{ij} \geq -d_{ij} \). It is straightforward to deduce \( s_{ij} - s_i \geq -d_{ij} \) or equivalently \( s_{ij} + d_{ij} \geq s_i \).

  On the other hand, from \( s_{ji} = d_i - d_j \) or \( s_{ji} \geq -d_{ji} \), we could similarly deduce \( s_i \geq s_{ji} - d_{ji} \). Hence the static delay to \( s_i \) is restricted by

  \[
  [\text{max}_s_i] = \min (d_{ij} + s_{ij}) \geq s_i \geq \max (-s_{ji} - d_{ji}) = [\text{min}_s_i]
  \]
• Pair-wise increase (delay) of the start/end events of an object:

Given a schedule \( \{d_i | i = 1, \ldots, k\} \), the maximal changes that could be applied to some \( d_i \) and \( d_j \), where \( d_i \) and \( d_j \) could be the start and end events of an object. We could apply the previous deduction to both \( d_i \) and \( d_j \) independently by removing the edge \((v_i, v_j, c_{ij})\) and/or \((v_j, v_i, c_{ji})\) from \( G \) before deriving \( s_i \) and \( s_j \) respectively. Then

\[
\min (\max_{s_i}, \max_{s_j}) \geq s \geq \max (\min_{s_i}, \min_{s_j})
\]

The range of these slack values is valuable information to a user in making adjustments.

4.3.4 Spatial Constraints

Spatial constraints are easily dealt with using the algorithms presented earlier.

Because of the use of

(i) various channels (such as audio, video and other output desires ), and

(ii) sharing of a visual display window that can be partitioned dynamically into sub-windows, a multimedia mail specification should also involve spatial constraints.

In the former case, since the presentation of some object may require exclusive use of some channel, it is important that even though two objects are unordered temporally, their spatial conflicts may require them to be ordered in the actual schedule to preserve their exclusiveness. This is apparent in the case of audio clips whose temporal presentations are not ordered but they need the exclusive use of the audio channel. Another example exists in the case of the visual sub-windows occupied by two different display objects, say graphics and text. If the two sub-windows overlap, then again the
presentation of the graphics and text must be serialized, even though the temporal specification has not been so specified.

The implication of having both spatial and temporal constraints in a multimedia specification has not been addressed in detail here because of the NP-completeness of the corresponding scheduling problem. We assume the system leaves the responsibility of serializing spatial conflicts to the author. If two objects must atomically share a common presentation channel, then they must be temporally ordered in the given temporal specification.

4.4 NP-completeness of Schedulability under both Temporal and Spatial Constraints

Problem: Given a temporal constraint graph $G$ and additional spatial constraints of the form: $O_i = \{ O_{i1},..., O_{ik} \}$ where $O_i$ denotes the set of objects that use channel $i$, $i = 1,..,j$.

Decision: Decide if the objects are schedulable, i.e., there exists $\{t_i\}$ that satisfies $G$ and at the same time, if $O_{ij}$ and $O_{ik}$ are in $O_i$, then the start/end interval of $O_{ij}$ does not overlap with that of $O_{ik}$.

The above decision problem is NP-complete. It is provable by reducing the 2-stage job-shop-scheduling problem.
Proof: An instance of the job-shop scheduling problem includes a set of tasks to be scheduled in a two-stage process, say A and B. Task i has some duration, say \( t_{i1} \) for stage 1 and \( t_{i2} \) for stage 2.

The above is transformed into a scheduling problem with temporal and spatial constraints as follows.

For each \( t_{i1} \) or \( t_{i2} \) we do:

\[
\begin{align*}
t_{i1\_end} - t_{i1\_start} &= t_{i1} \\
t_{i2\_end} - t_{i2\_start} &= t_{i2} \\
t_{i2\_start} - t_{i1\_end} &= 0
\end{align*}
\]

We also assign \( \{t_{i1}\} \) to channel 1 and \( \{t_{i2}\} \) to channel 2.

If the latter problem is decidable for a schedule time of \( T \), so is the original problem.

Hence the claim. QED

4.5 Our approach Vs. Buchanan’s Simplex Algorithm

In [15], Buchanan and Zellweger use the simplex algorithm to design a presentation schedule for a set of objects and their presentation constraints. Although the simplex algorithm involves the system to allow general (linear) constraints, it has some major disadvantages:

- it is very costly (exponential time complexity);
Their model does not take the advantage of the restricted syntax of the constraints, the time required for the simplex algorithm to determine that a set of specifications is inconsistent increases dramatically with an increase in the number of objects and constraints.

- it is not optimized to handle difference constraints;

Their model does not benefit from the special structure of the special structure of the difference constraints to increase the speed of solution generation.

- it does not handle inconsistencies well.

They are assuming that a given set of constraints are consistent and they do not handle inconsistent presentation, whereas our main aim is to be able to handle conflicting specifications for the easier composing and editing tool set offered to mail users.

Hence, although the use of the difference constraints somewhat limits the expressive power of the specification language, the resulting scheduling problem is more practicable. Besides, difference constraints already allow the author/reader a very broad and powerful range of temporal/spatial specification.
Chapter 5  Prototype Implementation

The input and output of the multimedia mail presentation system are described below:

**Input:** The input to the system consists of

- A set of temporal and spatial specifications
- A list of object sizes and their storage locations

**Output:** A presentation with a feasible schedule or a warning message together with the inconsistency detected.

To approach this, we divide our prototype system into three major parts,

1. **User Interface:** It collects the raw data and transforms them into difference constraints.

2. **Authoring:** A formatter derives a feasible schedule or detects an inconsistency.

3. **Presentation:** A translator translates the schedule into a SMIL program that is used by a composer to delivery the presentation. In the absence of a feasible schedule, an error message and inconsistency feedback is returned to the user, and then the edit session can be continued. Figure 5.0.1 shows the prototype system architecture.
5.1 GUI

A Multimedia System must manage a complicated range of run-time tasks. It may be required to retrieve and display many different formats of media objects. Presentation functions such as analysis of data and scheduling may be required before the actual presentation.

5.1.1 Prototype User Interface

The mail system should have a constraint-based authoring environment, where the author/reader can describe the spatial and temporal organization of a mail by setting constraints between presentation events. These constraints can express spatial placement
and temporal synchronization. Then the spatial and the temporal formatter compute the position of objects events in both spatial and temporal dimensions.

Figure 5.1.1 shows the first window (the login window) that appears when the system is invoked. The login window asks users to enter a user name and a password. This window functions as a security and access level handler, which adjusts the functionalities of the system user interface according to the access level of the user. In our prototype, we support two levels of access, author level and reader level.

![Figure 5.1.1 Login Screen for Multimedia Mail System](image)

Figure 5.1.2 shows an interface after the login of an author. The MediaObject Attribute window allows a user to create and place multimedia objects in time and in space by specifying the respective constraints. The system supports the addition or deletion of objects, or the modification of the attributes of an existing object. Based on the object type, the system can allocate an object to a particular spatial device. Other details such as actual placement are obtained from the user input.
The spatial characteristics of a display media object are captured by its minimum bounded rectangle. A user can create a new media object by using the "Add" button, and then the object property specification window (Figure 5.1.2) will pop onto the screen. This window allows the author specify the expected attributes of the given media object including

- Name: the name of the object
- File type: the associated spatial channel is implicitly specified
- Width and height: spatial attribute of a display object
- Min, Max duration: the scope of time interval for displaying the object
- Location: storage location of the file

Figure 5.1.2 MidiaObject Attribute Define Screen
A "Temporal Relation" window is provided for the entry of all temporal constraints among objects, as shown in Figure 5.1.3. An author can select two of the objects to specify the temporal relationships between the events of these two objects. At any point in time, a user can check the consistency of the specifications given so far by clicking Consistency Check, which opens the Consistency Checking Window, as shown in Figure 5.1.4. Clicking All Done Button on the Temporal Relationship window shown in Figure 5.1.3 will generate a schedule.

![Temporal Relation Window](image)

**Figure 5.1.3 Temporal Relationship Screen**

![Consistency Check Window](image)

**Figure 5.1.4 Consistency Check Window**
By clicking the View Structure button on the Consistency Checking window, the system opens the Mail Display Window, as shown in Figure 5.1.5. This window shows the detailed temporal and spatial attributes of the generated schedule. The Modify button returns control to the edit session in Figure 5.1.2 for changes of objects and other constraints. The Display button in the Consistency Checking window is used to start a presentation.

![Mail Display Window](image)

Figure 5.1.5 Mail Display Window

5.1.2 Author Interface Vs. Reader Interface

Author Interface

An author is allowed to enter a partial mail specification incrementally. During the course of composition, he can introduce new objects (hence events) and new constraints. The system can be invoked to run one of the three algorithms presented in Chapter 3,
depending on the situation involved: (i) edge_insertion, (ii) edge_deletion, or (iii) a new 
G. The system is able to provide a consistency checking and return a new schedule in 
each invocation.

Reader Interface

The reader interface supports a different set of editing tools to facilitate the reading of a 
mail. We distinguish what a reader is allowed to do during a reader session from what an 
author is allowed to do. While the author behavior is conceptually unrestricted, i.e., an 
author can change the document specification completely, the set of operators made 
available to the readers is more abstract and specifically tailored to reasonable uses by a 
reader. These include:

- Object omission
- Object migration
- Object expansion
- Object contraction

5.1.3 Sample Sequence of Interaction at Reader Site

In this subsection, a sequence of interactions in a reader session is shown as a 
demonstration of the key features in the reader presentation subsystem.

Event 1: the reader clicks on Receive button when there is a new mail coming, as shown 
in Fig 5.3.1.a
Event 2: The abstract Temporal and Spatial structure of the mail is displayed, as shown in Figure 5.3.1.b.

Event 3: the reader triggers a "Modify" window as shown in Figure 5.3.1.c. In this window, the reader can make changes by choosing from the four options provided:
Object Migration, Object Expansion, Object Contraction or Object Deletion. Consistency check will be invoked when the Finish Modification button is pressed.

![Modification Window](image)

Figure 5.3.1.c

Event 4: if consistency is maintained, the User Notification Window will display the details and allow the reader to view the details or modify again as shown in Figure 5.1.3.e.

![Consistency Checking for User Modification](image)

Figure 5.1.3.e Consistency Check Window

70
Event 5: The reader chooses to view the structure of the presentation, leading to the display of Mail Display Structure window, as shown in Figure 5.1.3.f.

![Mail Display Structure Window](image)

**Figure 5.1.3.f**

Event 6: Finally, mail presentation can be chosen and the full details of the mail will be presented according to the formatter's schedule.

### 5.2 Implementation of authoring – the Formatter

The presentation schedule generation system is responsible for deriving a solution to the given temporal/spatial specifications. There are two kinds of results the system may produce:

1. A feasible presentation schedule with the start-time, end-time, spatial position of each media object, in tabulated data form, or
2. A graph indicating a subset of edges whose weights could be modified to make the graph consistent. The possible changes are also presented to the user.

5.3 Implementation Presentation – Translator and Composer

Developing a multimedia authoring and presentation system is a difficult task, as many different problems have to be solved. The temporal nature of information in a multimedia mail requires new functions such as temporal synchronization between components and scheduling of the presentation in a distributed and sometimes unpredictable environment.

Two approaches are possible depending on how close the mail description is to the presentation. In an operational approach, the user specifies the ways in which a scenario must be executed by means of either a script language or an operational structure. Therefore the presentation phase directly implements the operational semantics provided by the used structure. In a constraint-based approach, the author specifies what presentation scenarios are allowed without choosing a particular one.

It is easier to build an authoring environment based on an operational approach than a constraint-based approach. Indeed, some theoretical problems associated with constraint-based approach are known to be difficult. This explains why currently available commercial tools are not based on the constraint technique. The same is true for the standard promoted by W3C for synchronized multimedia documents called SMIL [W3C98b], which we use as the prototype platform.
5.3.1 Synchronization and SMIL

During a presentation, the synchronization module must ensure that the synchronization relationships of the events are satisfied. Hence, we need a specification language that supports

1. the presentation of all relevant media objects, and
2. synchrony in the form of a real-time schedule

We use the Synchronized Multimedia Integration Language (SMIL) for this purpose. The proposed SMIL standard defines an XML-based language that allows control over the what, where, and when attributes of media elements in a multimedia presentation with a simple, clear markup language similar to HTML. It enables authors to specify synchronization between media objects by precise timing.

In a SMIL presentation, all of the media elements - images, audio clips, video clips, animations, and formatted text are referenced from a SMIL file, similar to the way in which an HTML page references its images and applets. A number of advantages come with such an approach.

- The plain-text nature of the SMIL file means that it is easy to create, easy to edit, and can even be assembled on the fly by Java applets or CGI scripts.
- It provides a bandwidth-friendly environment for multimedia presentation. Rather than streaming images and text as many redundant frames of encoded video, one can stream the image or text data just once, and display it in different ways.
SMIL - Concepts explained

SMIL is a declarative, XML-based timing and synchronization language. Using SMIL, an author can describe the temporal behavior of a multimedia presentation, associate hyperlinks with media objects, describe the layout of the presentation on a screen, and specify interactivity in multimedia presentations. SMIL allows integrating a set of independent multimedia objects into a synchronized multimedia presentation. Using SMIL, presentations such as a slide show synchronized with audio comments or a video synchronized with a text stream can be described. A typical SMIL presentation has the following characteristics. First, the presentation is composed from several components with different media type. Second, the begin and end times of different components are specified relative to events in other media components.

5.3.2 SMIL Document Presentation

General Semantics

A SMIL document may contain a head part and a body part. Either part may contain XML comments and XML processing instructions.

SMIL Timing

SMIL solves fundamental media synchronization problems and defines a powerful way of choreographing multimedia content. SMIL Timing defines elements and
attributes to coordinate and synchronize the presentation of media over time. Three synchronization elements support common timing use-cases:

The `<seq>` element plays the child elements one after another in a sequence.

The `<excl>` element plays one child at a time, but does not impose any order.

The `<par>` element plays child elements as a group (allowing "parallel" playback).

SMIL Timing also provides attributes that can be used to specify an element’s timing behavior. Each element has a beginning time and a fixed duration.

Figure 5.3.1 illustrates the basic support of a repeating element within a simple `<par` time container. The corresponding syntax is included with the diagram.

![Diagram of basic timing support](image)

Figure 5.3.1 - Strip diagram of basic timing support

```xml
<par begin="0s" dur="33s">

  <video begin="1s" dur="10s" repeatCount="2.5" fill="freeze" .../>

</par>
```
The SMIL Timing Model graph is a model of the presentation schedule and synchronization relationships. It is a dynamic structure that changes with the occurrence of user events or media deliveries.

5.3.3 SMIL - Example: Interactive Newscast

Consider a news broadcast on the growth of the Web. In the first scene (as shown in Figure 5.3.3), a graph on the left hand side of the screen displays the growth of the Web. The right hand side of the screen is taken up by a video of an anchorperson commenting the graph. The graph and the commentators' video are set up as a background.

In the second scene (right hand side in Figure 5.3.3), the graph is replaced by a video showing the interview guest John Doe, and the anchorperson starts to interview him. During the interview, the user can click on John's video, and his homepage will be brought up (via a hyperlink).

![Interactive News Broadcast]

*Figure 5.3.3: Interactive newscast screenshots newscast scenario*
Figure 5.3.4: Schedule for interactive

- The image "Web Growth" is shown from time 0:00 to time 1:00
- The text "John Doe" is shown from time 0:00 until the end of the John Doe Video
- The text "John Doe" is shown while the video of Tim is being displayed
- The user can follow a hyperlink connected to John's video and text while they are shown

This scenario can be implemented using the following SMIL document:
5.3.4 Translator

The translator is a key module to transform a presentation schedule into a SMIL formatted file, which can be presented by a SMIL composer. We can easily get the following basic object information along with the temporal and spatial attribute of each object in a multimedia mail with a valid presentation schedule;
The translation algorithm is abstractly described below:

The Translator has to partition the objects into subsets each of which can be started within the duration of the first object in the subset. Each subset is term a “PAR” in SMIL. The algorithm to extract such subsets from a given schedule is given below.

Recall that we assume $\text{Obj}_M = \{ O_1, O_2, \ldots, O_n \}$

**Algorithm 5.3.4:**

\[ j := 1; \]

repeat

Let $O_i$ = the earliest start event in $\text{Obj}_M$:

$\text{Par}_j := \{ O_i \}$;

Remove every $O_i$ from $\text{Obj}_M$ and insert it into $\text{Par}_j$ whenever $st(O_i)$ happens between $st(O_j)$ and $et(O_i)$;

\[ j := j + 1; \]

until $\text{Obj}_M = \emptyset$

**Example 5.3.4.a** : the Demo-Presentation, the system calculates a schedule as following (for the convenience, we skip the layout part in the following example):
<table>
<thead>
<tr>
<th>ID</th>
<th>ObjectName</th>
<th>Begin</th>
<th>End</th>
<th>Channel</th>
<th>Source</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Image1</td>
<td>0</td>
<td>5</td>
<td>0(window)</td>
<td>Newyear1.jpg</td>
</tr>
<tr>
<td>25</td>
<td>Presentation</td>
<td>9</td>
<td>14</td>
<td>0</td>
<td>computer.jpg</td>
</tr>
<tr>
<td>23</td>
<td>Audio1</td>
<td>0</td>
<td>5</td>
<td>1(audio)</td>
<td>music.au</td>
</tr>
<tr>
<td>78</td>
<td>Video1</td>
<td>5</td>
<td>9</td>
<td>2(Video)</td>
<td>arrowhit.avi</td>
</tr>
<tr>
<td>43</td>
<td>Audio2</td>
<td>11</td>
<td>20</td>
<td>1</td>
<td>sonata8.wav</td>
</tr>
</tbody>
</table>

Subsequently, the translator translates the schedule into a corresponding SMIL file. (The layout part is omitted here.)

```xml
<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE smil PUBLIC "-//W3C//DTD SMIL 1.0//EN"
  "http://www.w3.org/TR/REC-smil/SMIL10.dtd">
<smil>
  <head>
    <meta name="base" content="http://www.cwi.nl/GRIINS/examples/"/>
  </head>
  <body>
    <seq id="SEQID1">
      <par id="PARID1">
        <img id="Image1" src="Newyear1.jpg" begin="0s" dur="5s" />
        <audio id="Audio1" src="music.au" begin="0s" dur="5s" />
        ...
      </par>
      <par id="PARID2">
        <audio id="Audio2" src="sonata8.wav" begin="6s" dur="9s" />
        <video id="Video1" src=".:\Schedule\arrowhit.avi" begin="0s" dur="4s" />
        <img id="presentation" src="computer.jpg" begin="4s" dur="5s" />
        ...
      </par>
    </seq>
  </body>
</smil>

5.3.5 Composer - GRIINS Player for SMIL

We use the CWI GRIINS player to display the Mail formatted in SMIL. (CWI is able to support the SMIL standardization activity). They support the full SMIL V1.0
specification, with/without MPEG hardware and MMX support. Figure 5.3.3 shows the presentation screen for the example "Presentation".

![Image of GriNS display screens](image)

Figure 5.3.5 A GriNS display Screens for example “Presentation”

5.4 Experimental Analysis

We are convinced that the constraint model could bring significant advantages to multimedia authoring and presentation. The aim of this section is to list the key advantages as demonstrated through our prototype system.

5.4.1 Multimedia Mail System authoring

1. A secured and incremental authoring of multimedia mail
Authoring an interactive multimedia mail is a cyclic "specify, test and modify" process: the author can rarely complete the desired mail in one iteration. Easy-to-use is an important requirement in such a system. The use of constraint paradigms in our system has the added advantage of easy modification. For instance, if the author wants to insert a new object, the temporal checker can be used to check the consistency of the modification without involving the author directly. A new schedule can be generated whenever consistency is preserved. If consistency is violated, the author is alerted of the inconsistencies. This incremental system-assisted modification is provided.

In our system, different users may have access to some part of the original mail, and others have access to the other part of the mail. Hence, it is necessary to guarantee that collaborators can access only to the information that they are authorized.

2. Reusing multimedia specification

The reusability of the components of a mail is an important asset. A composite object in a mail is not a fixed object but an adjustable one, under the constraint-based module. This makes the reuse of different mail components easy.

Another kind of reuse occurs when the user wants to change some basic attribute of a existing object in a scheduled mail while preserving its temporal and spatial organization. One typical case is the translation of a document from one language to
another. The user has to replace each textual message and each audio comment. There is a high probability that the duration of the new audio clips may change. Such changes can be easily supported in our system.

5.4.2 Multimedia Mail System Presentation

1. Portability

Portable standards of media objects are evolving. Examples include MPEG audio, MPEG video, GIF, PNG and JPEG images. Often the required players are available on most existing platforms. Our system takes advantage of these formats and combines them into documents using a high level and portable language. Due to its XML structure, our system can easily import from and export to other standard languages such as SMIL.

2. Extendibility

An authoring environment should be extensible to support newly emerging media types or formats. In our system, extensibility is achieved by the system’s object model that ensures a widest variety of available plug in application.
Chapter 6 Conclusion and Future Work

In this thesis, we have developed a multimedia mail model that is applicable to specifying the detailed temporal and spatial relationships among the media objects. Our model is complemented by a set of algorithms that can be used to compose or read a mail incrementally. We have focused on flexibility and efficiency, as well as adjustability to runtime changes, such as system delays.

The authoring and presentation module of our prototype Multimedia Mail system handles features as:

1. temporal, spatial, and user interaction specification of a multimedia mail
2. dynamic modification of multimedia mail due to user editing, and
3. increment consistency-checking and re-scheduling of the modified specification automatically

6.1 Our contributions

As a multimedia mail is typically constructed over a period of time and has different users, it is inevitable that its structure may change over time. In addition to the uniform modeling of temporal and spatial constraints, we have developed incremental algorithms that:

1. determine if such constraints are consistent, and
2. incrementally finds a new solution once the constraints are changed.

Our major contributions are summarized as follows:

- Temporal, spatial and user interaction specifications of a multimedia mail: we use a simple but powerful constraint-based module. The system uses the flexibility inherent in the user specification to adapt presentations to changes in the mail or in the system load. We have developed techniques to handle inconsistencies as detected.

- Dynamic Modification of multimedia documents: A presentation may need to be modified due to the interest of the users. Hence, a presentation scheduler should cope with such changes, provided they are acceptable. Our prototype involves a scheduler that gives a wide latitude to user changes, and interacts with the user until a suitable presentation is achievable.

Much of the work in this thesis is equally applicable to a multimedia document presentation system. However, our distinct intent to bias our emphasis on a mail system is to make user of the possibly closer relationship between the mail author and the mail readers. In a general document presentation system, the audience is usually an open public. With the closer relationship between the author and readers, it is reasonable to provide an interface through which a reader may make some changes to the presentation constraints to facilitate the reading of the mail. This reader interface is an interesting aspect of the mail system proposed.
6.2 Future Work

We have described our approach to the design of a multimedia mail environment, how the users can specify temporal, spatial characteristics of a multimedia mail document, and how a multimedia system can use these characteristics to create a flexible multimedia presentation. Our method allows a multimedia presentation to adapt to changes in the mail. An extension of the prototype is to make it adaptable to runtime system parameters such as network delays/server delays and take QoS into schedulability consideration in optimizing a presentation.

As the combined temporal and spatial scheduling problem is provably NP-complete, we have solved the scheduling problem by separating these two concerns individually, with the help of the author. It would be reasonable to investigate a heuristic solution that can relieve the author of much of the responsibility in this process.
Bibliography

ActiveM3- An Authoring System for Active Multimedia Mail, Computer Graphics Center (ZGDV), Dec. 1995

Authoring Environment for Interactive Multimedia Documents, OPERA project, INRIA Rhone-Alpes (1995)

[3] Nabil Layaida, Loay Sabry-Ismail
Maintaining Temporal Consistency of Multimedia Documents Using Constraint Networks, OPERA Project, INRIA Rhone-Alpes

Collaborative Multimedia Documents: Authoring and Presentation, University of Maryland, 1996


Multimedia Documents with Elastic Time, ACM Multimedia Conference'95, 1995


[8] Allen J.F.,

Object Composition and Playback Models for Handling Multimedia Data, ACM Multimedia Conference'93, pp.273-281,1993

[10] Fargier H., Lang J., Schiex T
Mixed Constraint Satisfaction: a framework for Decision Problems under Incomplete Knowledge, IJCAI 95


Metadata for Multimedia Documents, SIGMOD Record, vol.23, no.4, pp.21-26, Dec.1996


Temporal Constraint Networks, Artificial Intelligence, vol.49, pp.61-95, 1991


Structured Multimedia Authoring, ACM Multimedia Conference '93, pp.283-289, 1993


Hyperstories: Combining Time, Space and Asynchrony in Multimedia Documents, Rc19277(83726), IBM Computer Science/Mathematics Research Report, 1993


Interval-Based Conceptual Models for Time-Dependent Multimedia Data, IEEE Transactions on Knowledge and Data Engineering, vol.5, no.4, pp.551-563, Aug.1993

89


[31] Blakowski G., Steinmetz R.A.


[33] Erfle R.


[34] International Standard ISO 8879

[35] Kim W., Kenchammana-Hosekote D.


[36] Legall D.


[37] Perez-Luque M.J., Little T.D.C.A


[38] Van Rossum G., Jansen J., Mullender K. and Bulterman D.


[39] W3c


[40] W3C

Working draft specification of SMIL (Synchronized Multimedia Integration Language). W3C: http://www.w3.org/TR/WD-smil, November 1997

[41] Macromedia Director

"User’s Guilde", Macromedia Inc., 1995

[42] IconAuthor 6.0

"User’s Guide", Aim Tech
[43] Price R.,

[44] Little T.D.C, Ghafoor A.