Hypermedia Synchronization: Modeling and Optimization with Graphs

Bruno Bachelet, Christophe Duhamel, Philippe Mahey, Luiz Fernando Soares

Hypermedia Synchronization: Modeling and Optimization with Graphs

Bruno Bachelet[1], Christophe Duhamel[2],
Philippe Mahey[3] and Luiz Fernando Soares[4]
LIMOS, UMR 6158-CNRS,
Université Blaise-Pascal, BP 10125, 63173 Aubière, France.

Draft - September 30, 2003

[1]bachelet@isima.fr - http://frog.isima.fr/bruno
[2]duhamel@isima.fr - http://frog.isima.fr/duhamel
[3]mahey@isima.fr
[4]lfgs@inf.puc-rio.br - Departamento de Informática, PUC-Rio, Rio de Janeiro, Brazil

Abstract

This article presents the modeling and optimization methods for a new kind of synchronization problems arising in the field of hypermedia presentations. It is shown that they can be seen as tension problems in graphs. Two of them are described in detail: the minimum cost tension problem and its discrete cost counterpart. The former is solved with pseudo-polynomial or polynomial methods. The latter is NP-complete and we discuss ways to improve its minimal and maximal objective bounds.

Keywords: hypermedia synchronization, temporal graphs, minimum cost tension, linear programming.

Résumé

Cet article présente une modélisation et des méthodes d'optimisation pour un nouveau genre de problèmes de synchronisation dans le domaine des présentations hypermédia. Il a été montré qu'ils peuvent être vus comme des problèmes de tension dans des graphes. Deux d'entre eux sont décrits en détails : le problème de la tension de coût minimum et son équivalent avec des coûts discrets. Le premier est résolu par des méthodes pseudo-polynômiales ou polynômiales. Le second est NP-complet et nous discutons des manières d'améliorer ses bornes objectif minimale et maximale.

Mots clés : synchronisation hypermédia, graphes temporels, tension de coût minimal, programmation linéaire.

Acknowledgements / Remerciements

This project was partially funded by France-Brazil cooperation project CAPES-COFECUB 398/02.

Abstract

This article presents the modeling and optimization methods for new kinds of synchronization problems arising in the field of hypermedia presentations. It is shown that they can be seen as tension problems in graphs. Two of them are described in detail: the minimum cost tension problem and its discrete cost counterpart. The former is solved with pseudo-polynomial or polynomial methods. The latter is NP-complete and we discuss ways to improve its minimal and maximal objective bounds.

Keywords: hypermedia synchronization, temporal graphs, minimum cost tension, linear programming.

Introduction

The exploding use of Internet and online presentations has turned crucial the need to dispose of robust real-time algorithms to deal with the complexity and the interactivity of the documents. One of the main challenges that has emerged recently is the ability to manage the synchronization of hypermedia documents by considering that each object can be compressed or delayed like an elastic spring. The heterogeneity of the objects that compose a hypermedia document turns their presentation in time and space a hard problem. On the other hand, interactivity imposes that real-time updates of the schedule of the document should be possible, increasing the need for faster decision-making algorithms.

As explained in [7], such documents are composed of media objects (audio, video, text, image...), which duration of presentation must be adjusted to satisfy a set of temporal constraints that express the progress of the animation as defined by the author. But for these constraints to be satisfied, the author must accept some flexibility on the duration (that will be called *ideal*) of presentation for each object, pauses being totally forbidden if not explicitly wanted. In order to estimate the quality of an adjustment, a cost function, usually convex (cf. Figure 1), is associated with each object. Therefore, the problem we address here is to find an adjustment of best quality, i.e. which minimizes the sum of the costs of the media objects.

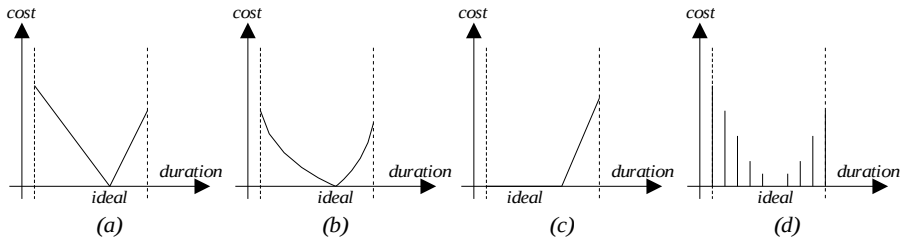

Figure 1: Examples of cost functions. a) Piecewise linear with a single ideal value. b) Non-linear, but convex and derivable. c) Piecewise linear with several ideal values. d) Discrete values.

The set of temporal constraints can be modeled as a directed graph $G = (X; U)$ where X is a set of nodes, U a set of arcs, $m = |U|$ and $n = |X|$. The nodes are associated with events (the start or the end of presentation of an object). The arcs express duration constraints between nodes. With each arc u is associated a time domain S_u, an ideal duration o_u and a cost function $c_u : S_u \longmapsto \mathbb{R}$. An arc $u = (x; y)$ between two nodes x and y means the event x precedes y and they are separated by a duration $\theta_u \in S_u$.

Let $\pi : X \mapsto \mathbb{R}$ be a potential function which assigns a date to each event node of the graph. Then the duration θ_u of an object associated with an arc $u = (x; y)$ can be seen as the difference of potentials $\theta_u = \pi_y - \pi_x$. In other words, $\theta = (\theta_u)_{u \in U}$ is a *tension* vector on the graph (e.g. [12]). Let A denote the incidence matrix of the graph, i.e. A is of dimension $(m \times n)$ with the

elements a_{xu} equal to -1 (if u leaves x), $+1$ (if u comes to x) or 0 (any other case). The problem, called the *minimum cost tension* problem, is simply formulated as follows:

$$(P_1)\begin{cases} \text{minimize } \sum_{u\in U} c_u(\theta_u) \\ \\ \text{with } \theta = A^T\pi,\ \theta \in S \end{cases}$$

There is another way to define a tension on a graph. It relies on the property (cf. [6]) that a vector θ is a tension on the graph G if and only if for any cycle γ of G:

$$\sum_{u\in\gamma^+} \theta_u - \sum_{u\in\gamma^-} \theta_u = 0 \tag{1}$$

γ^+ being the arcs of γ in some direction and γ^- the arcs in the opposite direction. The minimum cost tension problem can then be formulated as follows:

$$(P_2)\begin{cases} \text{minimize} \quad \sum_{u\in U} c_u(\theta_u) \\ \\ \text{with} \quad \sum_{u\in\gamma^+} \theta_u - \sum_{u\in\gamma^-} \theta_u = 0,\ \forall \text{ cycle } \gamma \in G \\ \\ \theta \in S \end{cases}$$

In this article, we study the minimum cost tension problem subject to two kinds of objective functions. First, we consider the domain S_u of an arc u as a continuous interval $[a_u; b_u]$ and assume the cost function c_u to be convex two-piecewise linear as shown in Figure 1a. The adaptation to more pieces of the methods described here is straightforward. Hence, we consider the cost function c_u as follows (and refer to the problem as the *linear* problem):

$$c_u(\theta_u) = \begin{cases} c_u^1(o_u - \theta_u)\text{ , if } \theta_u < o_u \\ c_u^2(\theta_u - o_u)\text{ , if } \theta_u \geq o_u \end{cases}$$

As explained in [14], authors of hypermedia documents often need to minimize the number of media objects that are not scheduled at their ideal value, because stretching or shrinking the presentation of an object is time consuming, and so, does not suit for real time scheduling. While the domain S_u of an arc u is still a continuous interval $[a_u; b_u]$, we propose to investigate the problem (and refer to it as the *discrete* problem) with binary cost functions defined as follows:

$$c_u(\theta_u) = \begin{cases} 0\text{ , if } \theta_u = o_u \\ 1\text{ , if } \theta_u \neq o_u \end{cases} \tag{2}$$

Section 1 presents an overview of the methods available to solve the linear problem. Section 2 introduces the discrete problem and discusses ways to reduce the gap between the lower and the upper bounds of the problem in order to improve its resolution speed, either with exact or approximate algorithms. It also explains how resolution techniques for the discrete problem can take advantage of methods used for the linear problem. Section 3 shows numerical results concerning the speed of some methods and the quality of their solution when approximated. We conclude this article with our thoughts on how to exploit the results presented here to improve the resolution of the discrete problem.

1 The Minimum Linear Cost Tension Problem

1.1 Methods on Non-Specific Graphs

When piecewise linear costs are considered, it is possible to model the problem with linear programming. This solution is widely used for the synchronization problem (e.g. [8], [13]), using either model (P_1) or model (P_2). In practice, instances of model (P_2) can be solved faster than instances of model (P_1), but it raises the problem of generating cycles from graph G, which is discussed further in Section 2.

Another way to solve the problem is to apply the *out-of-kilter* algorithm first introduced for the minimum cost flow problem by Fulkerson in [10] and later for the minimum cost tension problem by Pla in [15]. An adaptation of this method to piecewise linear costs has been proposed in [3]. This algorithm is pseudo-polynomial in $O(m^2(A+B)$ operations where $A = \max_{u \in U}\{a_u; b_u\}$ and $B = \max_{u \in U}\{c_u^1; c_u^2\}$. The approach is based on the search of specific cycles or cocycles that progressively modifies the flow and the tension of the graph to converge on the optimal tension. The efficiency of this approach is equivalent to the linear programming in terms of speed, but has the advantage of being incremental. This means the algorithm only needs a few iterations (i.e. cycles or cocycles searches), from the previous optimal solution, to go back to the optimal tension if the constraints of the problem are slightly altered.

Hadjiat [12] presents a strongly polynomial method based on *cocycle canceling*, its complexity is $O(n^4 m^3 \log m)$ operations. Unfortunately, this approach is only really efficient in practice for a special class of graphs: *Penelope*'s graphs.

More recently, Ahuja et al. [1] proposed an algorithm to solve a more generic problem called the *convex cost integer dual network flow* problem. The algorithm consists in transforming the minimum cost tension problem into a minimum cost flow problem, solved with the well-known *cost-scaling* method (cf. [2]). This algorithm is polynomial, $O(mn^2 \log nA)$ operations, and proves to be very efficient in practice.

1.2 Method on Series-Parallel Graphs and Quasi Series-Parallel Graphs

Tension graphs expressing constraints from hypermedia synchronization are well structured and very close to a peculiar class of graphs: the *series-parallel* graphs or *SP-graphs*. These graphs result from two operations on arcs: the series one that links two arcs $u = (x; y)$ and $v = (y; z)$, and the parallel one that links two arcs $u = (x; y)$ and $v = (x; y)$. A SP-graph is only composed by applying these two specific relationships.

We proposed a method called *aggregation* to solve the minimum linear cost tension problem on this class of graphs. The approach is strongly polynomial, $O(m^3)$ operations, and provides not only the optimal tension, but also enough information to optimally change the tension with only a linear number of operations, when adding the simple constraint that the tension between the source (i.e. the node with no predecessor) and the target (i.e. the node with no successor) of the graph must be equal to a given tension t. That means a whole SP-graph can be seen as a single arc with a tension t and a piecewise linear cost function (with more than 2 pieces) as explained in [4].

This assessment allowed us to propose a method called *reconstruction* to solve the problem on quasi SP-graphs, i.e. SP-graphs with some additive arcs that are disrupting the series-parallel property. The idea of the method is to decompose the graph into k SP-subgraphs and solve the problem with the aggregation technique on each of them. Hence, each subgraph can be aggregated and replaced by a single arc in the whole problem, which can be solved then with the out-of-kilter

method. This approach, presented further in [5], is pseudo-polynomial with $O(m^3 + km(A+B))$ operations, and proves to be the most time efficient in practice on quasi SP-graphs.

2 The Minimum Discrete Cost Tension Problem

The minimum cost tension problem with discrete costs has been recently addressed in [14] concerning hypermedia synchronization and in [9] about compilation techniques. Darte et al. [9] proved that the problem is NP-complete, and Ribeiro et al. [16] showed through another demonstration that even on SP-graph the problem is still NP-complete.

2.1 Mixed Integer Model

The problem can be modeled as a mixed integer program, as proposed in [14]. The binary cost functions c_u as defined by (2) in Section 1 are modeled with binary variables y_u and constraints (a), (b) and (e) in the following model:

$$(P_3)\begin{cases} \text{minimize} \quad \displaystyle\sum_{u\in U} y_u & \\ \\ \text{with} \quad -\theta_u - (o_u - a_u)y_u \le -o_u, \forall u \in U & (a) \\ \quad\quad \theta_u - (b_u - o_u)y_u \le o_u, \forall u \in U & (b) \\ \\ \quad\quad \theta \text{ is a tension} & (c) \\ \\ \quad\quad \theta_u \in [a_u; b_u], \forall u \in U & (d) \\ \quad\quad y_u \in \{0,1\}, \forall u \in U & (e) \end{cases}$$

The constraints (c), which mean θ is a tension, can be modeled as in (P_1) or in (P_2). In the latter, a constraint (cf. (1) in Section 1) should be added for each cycle of the graph G. If we consider a cycle base, the satisfaction of the constraint for each cycle of the base is proved to be sufficient to satisfy the constraint for all the cycles of the graph. The easiest way to determine such a base is to find a covering tree T of the graph G and, for each arc u of $G \setminus T$, to identify the cycle in $T \cup \{u\}$. However empirical tests showed us that the time to solve the minimum cost tension problem (with either linear or discrete costs) is highly dependent on the structure of the cycle base used in the linear program.

2.2 Continuous Relaxation

Let consider now the continuous relaxation (P_3^C) of problem (P_3) where constraints (e) are relaxed to $y_u \in [0;1]$. As each variable y_u appears in only two constraints (a) and (b), one of them will be tight when the relaxed problem is solved. That means either $-\theta'_u - (o_u - a_u)y'_u = -o_u$ (if $\theta'_u < o_u$) or $\theta'_u - (b_u - o_u)y'_u = o_u$ (if $\theta'_u \ge o_u$) in the optimal solution (θ', y'). In other words, in the relaxed program (P_3^C), y_u can be seen as the piecewise linear cost function c_u defined as follows:

$$c_u(\theta_u) = \begin{cases} (o_u - \theta_u)/(o_u - a_u), \text{ if } \theta_u < o_u \\ (\theta_u - o_u)/(b_u - o_u), \text{ if } \theta_u \ge o_u \end{cases}$$

Constraints (a), (b) and (e) are not useful anymore, so the continuous relaxation of problem (P_3) can be summarized as:

$$(P_3^C)\begin{cases} \text{minimize} \quad \displaystyle\sum_{u\in U} c_u(\theta_u) & \\ \text{with} \quad \theta \text{ is a tension} & (c) \\ \qquad \theta_u \in [a_u; b_u], \forall u \in U & (d) \end{cases}$$

The continuous relaxation is finally a minimum linear cost tension problem as addressed in Section 2, and it can be solved by any method discussed there. Section 3 shows that some of them are really more efficient in terms of speed than linear programming, so working on this continuous relaxation appears to be interesting to get time-efficient algorithms.

2.3 Lower Bound

The lower bound z^C provided by the resolution of the continuous relaxation (P_3^C) is poor. Results in Section 3 show that the gap between the lower bound z^C and the minimum objective value z^* is about $45\ \%$ of z^*. We propose additional constraints to reduce the gap between z^C and z^*. The idea is to solve the continuous relaxation to obtain the lower bound z^l and select one variable y_u, for instance one that is not equal to 1. For this variable, we try to find an additional constraint that is violated in the current relaxed solution.

We propose to select a cycle γ containing the arc u and then compute the optimal solution y^γ of the minimum discrete cost tension problem (P_3^γ) reduced to this single cycle. The objective value $z^*_\gamma = \sum_{u\in\gamma} y^\gamma_u$ is the minimum $\sum_{u\in\gamma} y_u$ can reach in the continuous relaxation (P_3^C). Hence, if $\sum_{u\in\gamma} y_u \geq z^*_\gamma$ is not satisfied in (P_3^C), this constraint is added into (P_3^C) in order to increase z^l. Constraints are added until z^l can not be improved significantly anymore. During our tests, we used mixed integer programming to solve (P_3^γ), but one major improvement will be to find a method to obtain the lower bound more efficiently.

2.4 Lagrangean Relaxation

Let consider the Lagrangean relaxation (P_3^L) of problem (P_3): constraints (a) and (b) are relaxed and introduced into the objective function with the associated Lagrangean coefficients $\lambda^1 \leq 0$ and $\lambda^2 \leq 0$. The Lagrangean function is thus defined as follows:

$$\begin{aligned} L(\lambda_1, \lambda_2) &= \sum_{u\in U} c'_u y_u + \sum_{u\in U} (\lambda^1_u - \lambda^2_u)\theta_u + \sum_{u\in U} (\lambda^2_u - \lambda^1_u) o_u \\ &\text{where } c'_u = 1 + a_u\lambda^1_u - b_u\lambda^2_u \end{aligned} \tag{3}$$

Variables y_u do not appear in the constraints anymore, therefore they can be fixed either to 0 or 1 according to the sign of c'_u. Thus, only constraints (c) remain along with θ_u variables. This system is totally unimodular (cf. [11]) which raises the integrality property in the Lagrangean relaxation. As a matter of fact, the bound z^L it provides can not be better than the continuous relaxation bound z^C, i.e. $z^C = z^L$ (cf. [17]).

2.5 Upper Bound

Solving the discrete problem with mixed integer programming would be too time consuming on large instances. For this reason, Medina et al. [14] propose a heuristic to find quickly a good

solution that is an upper bound to problem (P_3). This technique is based on the resolution of the continuous relaxation (P_3^C). After (P_3^C) has been solved, the closest variables to 0 ($y_u \leq \alpha$) and to 1 ($y_u \geq 1 - \alpha$) are fixed, i.e. constraints ($y_u = 0$ or $y_u = 1$) are added into (P_3^C). The problem is solved again and the process iterates until all the variables are fixed to 0 or 1. This approach provides a good upper bound z^u to the problem. But this fixing policy is somewhat too restrictive since variables seem to be fixed a bit early to 1. It might close too soon their possibility to be fixed to 0 later, which would be better. Hence, we investigated various alternate fixing policies, but practical tests showed that the problem is not very sensitive to the fixing strategy, so that this greedy approach is limited.

For this reason, Medina et al. [14] propose a more time consuming method to find a better upper bound. The idea is to assign the reduced cost c'_u from the Lagrangean relaxation (cf. (3)) to each variable y_u in the cost function z of problem (P_3^C), i.e. $z = \sum_{u \in U} c'_u y_u$. These reduced costs are computed from the solution found after some iterations of the subgradient algorithm in the resolution of the Lagrangean relaxation (P_3^L). Unfortunately, the improvement of the upper bound z^u is not significant compared with the time spent to try to decrease it.

3 Numerical Results

3.1 The Minimum Linear Cost Tension Problem

Tables 1 and 2 show a practical comparison of the methods, which is always difficult because of all kinds of biases. But the goal here is to get an idea of how the methods behave on graphs with non-specific structure or on quasi SP-graphs. The notion of *SP-perturbation* is introduced in [5] to identify a quasi SP-graph. It is a percentage of arcs among the whole graph that are disrupting the series-parallel property of the graph. Table 1 shows results for quasi SP-graphs when size varies and the SP-perturbation is set to $4\ \%$. Table 2 points the performances of the methods for various SP-perturbation, the graph size being fixed to $n = 500$ and $m = 3000$. Results are expressed in seconds on a Celeron 500 MHz processor under a Linux operating system. We used GNU C++ 2.95 compiler and its object-oriented features to implement the methods. For the linear programming, we use the simplex method provided with GLPK 4.0 software. These results are the means of series of 10 tests on randomly generated graphs. Both A and B are fixed to 1000.

Nodes	Arcs	GLPK (s)	Out-of-Kilter (s)	Dual Cost Scaling (s)	Reconstruction (s)
50	200	0.08	0.02	0.03	0.05
50	400	0.23	0.04	0.07	0.10
100	400	0.25	0.07	0.09	0.12
100	800	1.05	0.16	0.18	0.24
500	2000	7.56	1.48	1.57	0.95
500	4000	34.79	3.21	3.24	1.97
1000	4000	35.51	5.27	4.54	2.69
1000	8000	249.61	11.43	8.70	5.62

Table 1: Numerical results for the linear problem, graph size influence.

It appears from both tables that linear programming can be replaced by any of the other methods to solve the continuous relaxation P_3^C. However their efficiency depends on both the SP-perturbation and the size of the graph. The reconstruction method is well suited for a SP-perturbation below $8\ \%$ and for large quasi SP-graphs, whereas the dual cost scaling method is better suited for non-specific graphs or small quasi SP-graphs.

SP-Perturbation (%)	GLPK (s)	Out-of-Kilter (s)	Dual Cost Scaling (s)	Reconstruction (s)
2	12,58	1,85	2,28	1,13
4	17,65	2,21	2,45	1,45
6	21,63	2,63	2,27	1,78
7	26,50	2,80	2,11	2,03
8	25,17	2,83	2,12	2,26
9	28,51	3,07	2,11	2,42
10	28,32	3,27	2,12	2,78
15	37,38	3,86	2,06	4,41
20	45,05	4,47	2,02	6,43

Table 2: Numerical results for the linear problem, SP-perturbation influence.

3.2 The Minimum Discrete Cost Tension Problem

Table 3 presents the bounds found for 8 instances of the minimum discrete cost tension problem: the lower bound z^C that is the optimal solution of (P_3^C), the improved lower bound z^l that is the optimal solution of (P_3^C) with the additional constraints introduced in Section 2, the optimal solution z^* obtained by CPLEX 8.0 or its best lower z_l^* and upper z_u^* bounds if the optimal solution is not found in 100 seconds, and the upper bound z^u found with the heuristic presented in Section 2. The time limit for CPLEX is realistic in the context of hypermedia synchronization, since we want some quick answer from the solver. Therefore, it may be better to have a good solution quickly than an optimal one later.

Nodes	Arcs	Lower Bound z^C (z^C/z_l^* %)	Lower Bound z^l (z^l/z_l^* %)	CPLEX z^* or z_l^*/z_u^*	Upper Bound z^u (z^u/z_u^* %)
10	40	15 (48 %)	23 (74 %)	31*	32 (103 %)
10	80	31 (44 %)	46 (66 %)	70*	71 (101 %)
20	80	30 (48 %)	39 (63 %)	62*	63 (102 %)
20	160	66 (47 %)	88 (63 %)	139*	141 (101 %)
50	200	70 (60 %)	89 (76 %)	117/153	154 (101 %)
50	400	152 (56 %)	194 (71 %)	273/355	355 (100 %)
100	400	134 (64 %)	161 (77 %)	209/312	307 (98 %)
100	800	313 (70 %)	348 (78 %)	447/708	705 (100 %)

Table 3: Numerical results for the discrete problem.

We can clearly see that the lower bound z^C provided by the continuous relaxation (P_3^C) is not very sharp. This suggests the integer model (P_3) is not strong enough to ensure fast computational results from any linear solver. The additional constraints help to reduce the gap between z^l and z^*. However they are still not sufficient and some more work should be done in order to further reduce the gap. We can also note that the heuristic gives good results. The approximate solution seems to stay close to the optimum. This is even true when graph size grows. For the larger instances, CPLEX upper bound is dominated by the bound given by the heuristic.

Conclusion

We have presented the minimum cost tension problem arising during the optimization of hypermedia document schedules. Several dedicated algorithms have been introduced for solving the continuous case. They have proved to be very efficient in practice and are a very good alternative to linear programming tools. The discrete case turns out to be more complex, especially because the integer formulation is quite weak. We are investigating other kinds of constraints to develop a polyhedral approach (such as branch and cut). The model could also be extended to take into account discrete interval for the duration of hypermedia objects.

References

[1] R.K. Ahuja, D.S. Hochbaum, and J.B. Orlin. Solving the Convex Cost Integer Dual Network Flow Problem. In *Lecture Notes in Computer Science*, volume 1610, pages 31–44, 1999.

[2] R.K. Ahuja, T.L. Magnanti, and J.B. Orlin. *Network Flows - Theory, Algorithms, and Applications*. Prentice Hall, 1993.

[3] B. Bachelet and P. Mahey. Optimisation de la présentation d'un document hypermédia. In *Annales Scientifiques de l'Université Blaise Pascal*, volume 110-42, pages 81–90, 2001.

[4] B. Bachelet and P. Mahey. Minimum Convex-Cost Tension Problems on Series-Parallel Graphs. To appear in *RAIRO Operations Research*, 2003.

[5] B. Bachelet and P. Mahey. Minimum Piecewise Linear Cost Tension Problem on Almost Series-Parallel Graphs. Technical report, LIMOS, Université Blaise Pascal, Clermont-Ferrand, France, 2003.

[6] C. Berge and A. Ghoula-Houri. *Programmes, jeux et réseaux de transport*. Dunod, 1962.

[7] M.C. Buchanan and P.T. Zellweger. Specifying Temporal Behavior in Hypermedia Documents. In *European Conference on Hypertext '92*, pages 262–271, 1992.

[8] M.C. Buchanan and P.T. Zellweger. Automatically Generating Consistent Schedules for Multimedia Documents. In *Multimedia Systems*, pages 55–67. Springer-Verlag, 1993.

[9] A. Darte and G. Huard. New Complexity Results on Array Contraction and Related Problems. Technical report, Laboratoire de l'Informatique du Parallélisme, Ecole Normale Supérieure de Lyon, France, 2002.

[10] D.R. Fulkerson. An Out-of-Kilter Method for Minimal Cost Flow Problems. In *SIAM Journal on Applied Mathematics*, volume 9, pages 18–27, 1961.

[11] M. Hadjiat. *Problèmes de tension sur un graphe - Algorithmes et complexité*. PhD thesis, Université de la Méditerranée, Marseille, France, 1996.

[12] M. Hadjiat. Penelope's Graph: a Hard Minimum Cost Tension Instance. In *Theoretical Computer Science*, volume 194, pages 207–218. Elsevier Science, 1998.

[13] M.Y. Kim and J. Song. Multimedia Documents with Elastic Time. In *Multimedia '95*, pages 143–154, 1995.

[14] M.T. Medina, C.C. Ribeiro, and L.F.G. Soares. Automatic Scheduling of Hypermedia Documents with Elastic Times. In *Parallel Processing Letters*, 2002.

[15] J.M. Pla. An Out-of-Kilter Algorithm for Solving Minimum Cost Potential Problems. In *Mathematical Programming*, volume 1, pages 275–290, 1971.

[16] C.C. Ribeiro and E. Sanlaville. On the Complexity of Scheduling with Elastic Times. Submitted for publication, 2002.

[17] L.A. Wolsey. *Integer Programming*. Wiley-Interscience, 1998.