ELSEVIER

Discrete Applied Mathematics 63 (1995) 101–116

DISCRETE APPLIED MATHEMATICS

Scheduling dyadic intervals

James R. Driscoll[a], Dennis M. Healy[b], Garth T. Isaak[c,*]

[a]Driscoll Brewing, Murray Hill, NJ 07974, USA
[b]Department of Mathematics and Computer Science, Dartmouth College, Hanover, NH 03755, USA
[c]Department of Mathematics, Lehigh University, Bethlehem, PA 18015, USA

Received 16 August 1993; revised 9 May 1994

Abstract

We consider the problem of computing the shortest schedule of the intervals $[j2^{-i},(j+1)2^{-i})$, for $0 \leqslant j \leqslant 2^i - 1$ and $1 \leqslant i \leqslant k$ such that separation of intersecting intervals is at least R. This problem arises in an application of wavelets to medical imaging. It is a generalization of the graph separation problem for the intersection graph of the intervals, which is to assign the numbers 1 to $2^{k+1} - 2$ to the vertices, other than the root, of a complete binary tree of height k in such a way as to maximize the minimum difference between all ancestor descendent pairs. We give an efficient algorithm to construct optimal schedules.

1. Introduction

The problem we consider arises in an application of wavelets to magnetic resonance imaging. Roughly speaking, it is possible to measure the inner product of the spatial density of an object to be imaged with a chosen function. If we focus on one of the three spatial dimensions, then measuring the inner product of the spatial density with the complex exponentials, a typical method, would in effect allow the measurement of the Fourier transform of the density along that dimension. In practice, the density is periodicized and assumed to be band-limited. Thus, a finite number of Fourier coefficients suffice to reconstruct the density.

One drawback of this method is that an inner product measurement can be made very quickly (tens of milliseconds), but it is necessary to let the region over which the inner product is taken recover for as long as 2 seconds before another measurement in that region can be made. One way of improving upon this situation is to use a different basis such as a wavelet basis. It is possible to construct an orthonormal wavelet basis W_{ij}, the Haar basis, for the unit interval such that basis elements are supported on the dyadic intervals, that is, W_{ij}, $i \geqslant 0$, $0 \leqslant j < 2^i$, is supported on the interval $[j/2^i, (j+1)/2^i)$. In practice, one must again assume that the density is band-limited,

*Corresponding author.

SSDI 0166-218X(94)00068-9

that is, that the inner product of the density W_{ij} is zero for i greater than some fixed constant k, thereby allowing a finite number of measurements to reconstruct the density. The advantage of using this basis is that it is not necessary to wait for the previously measured interval to recover completely before taking another measurement because there are many basis elements supported on non-overlapping intervals. For a more precise description of the application to magnetic resonance imaging, see [2].

The problem we consider in this paper is the computation of a minimum time schedule of the dyadic intervals such that every pair of intervals is scheduled at least the measurement time apart, and every overlapping pair of intervals is scheduled at least the recovery time apart. We first make some preliminary observations, and then prove a lower bound that provides the intuition for the upper bound to follow. We then give an efficient algorithm to compute an optimal schedule.

2. Preliminaries

We associate the dyadic intervals of length at least $1/2^k$ with a complete binary tree of height k, T_k. The root, v_{00}, corresponds to the unit interval, while the children of the node v_{ij}, $v_{i+1,2j}$ and $v_{i+1,2j+1}$ correspond to its left and right half intervals. This tree is the diagram of the partial order corresponding to interval containment. Call the intersection graph of these intervals I_k. This graph can alternatively be viewed as the comparability graph of the partial order corresponding to the tree diagram.

Normalize the measurement time to 1, and call the resulting recovery time R. What we seek is a map $S: V(T_k) \mapsto \mathfrak{R}$ such that
(1) $S(v) \geqslant 0$,
(2) $|S(u) - S(v)| \geqslant 1$, for $u \neq v$,
(3) $|S(u) - S(v)| \geqslant R$, for u an ancestor of v or vice versa, and
(4) $|S| = \max_{v \in V(T_k)} S(v)$ is minimized.
If S satisfies (1)–(3) it is called a *schedule*. If it additionally satisfies (4), it is called a *minimum schedule*. The parameters to the problem are the recovery time R and the tree T_k. The recovery time R can be any positive real number. We do not assume that R is an integer.

As an example, consider the tree T_2 with recovery time $R = 4$. An appealing idea is to schedule all of the leaves first, since they are all unrelated and can be scheduled one right after the other, then schedule their parents as soon as possible, and so on until finally the root has been scheduled. The resulting schedule is illustrated in Fig. 1(a); the vertices are labelled with $S(v)$. Fig. 1(b) shows a different presentation of that schedule. The horizontal axis is the unit interval on which all the intervals lie. The vertical axis is the scheduling time of the interval. In this diagram, any vertical line that intersects two intervals must do so at least R units apart. Fig. 1(c) shows a different schedule with $|S|$ smaller than the previous example. This is in fact a minimum schedule.

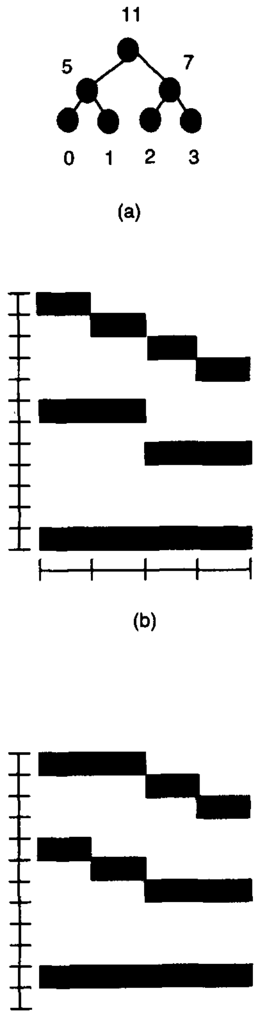


Fig. 1.

The most constraining vertex to schedule is, of course, the root. Since it is an ancestor of every other vertex, it is impossible to schedule any vertex between $S(v_{00}) - R$ and $S(v_{00}) + R$. In fact, it is easy to see that we might as well schedule it first.

Lemma 1. *There is a minimum schedule with* $S(v_{00}) = 0$.

Proof. Consider a minimum schedule where the root is not scheduled first. If it is scheduled last, reverse the schedule and it is now first. (This is best understood by considering diagrams of the form of Figs. 1(b) and (c). Many symmetries of these diagrams preserve properties (1)–(4) above.) If it is not last, pull it out of the middle and "close up" the schedule by R. Slide down the whole schedule by R and place the root first. This resulting schedule is a minimum schedule for recovery time R. $\square$

Since the root may be scheduled first, and no other interval may be scheduled before time R, after which constraint 3 above is no longer binding with respect to the root interval, it suffices to construct a minimum schedule for the two subtrees of the root alone. This is the minimum schedule problem for $T_{k-1} \cup T_{k-1}$, the form of the problem we will consider in the remainder of the paper. By $T_{k-1} \cup T_{k-1}$ we mean the disjoint union of two T_{k-1} on different vertex sets.

A related question is, what is the largest recovery time R such that $T_{k-1} \cup T_{k-1}$ can be scheduled one immediately after the other, that is, when $S: V(G) \mapsto \{0, \ldots, |V(G)| - 1\}$. This problem is the graph separation problem [1, 3, 4], which is a "dual" of the graph bandwidth problem. The separation number of a graph G is the largest s such that there is a bijection $f: V(G) \mapsto \{0, \ldots, |V(G)| - 1\}$ such that $|f(u) - f(v)| \geqslant s$ if $(u, v) \in E(G)$. The problem of determining whether the separation number of a graph is $> k$ is in general NP-complete: a graph has a Hamiltonian path if and only if its complement has separation number > 1 [3]. The separation number of I_k, the interval graph corresponding to T_k, is 1 because the unit interval is adjacent to every other interval. However, the separation number of $I_{k-1} \cup I_{k-1}$, which corresponds to our reduced problem without the root, is not as easily determined. Later, we will compute it, as the ordering of the intervals that achieves maximum separation will prove in this case (though not in general, as seen by I_k) to be the optimum order for a minimum schedule of $T_{k-1} \cup T_{k-1}$, for every positive real recovery time R.

We now turn to a lower bound for $|S|$ on $T_{k-1} \cup T_{k-1}$.

3. A lower bound

Theorem 2. *If S is a schedule for $T_{k-1} \cup T_{k-1}$ with separation R then*

$$|S| \geqslant \begin{cases} n - 1 & \text{if } R \leqslant \lfloor \frac{n}{k} \rfloor, \\ (n - k\lfloor \frac{n}{k} \rfloor)(\lceil \frac{n}{k} \rceil - R) + kR - 1 & \text{if } \lfloor \frac{n}{k} \rfloor < R < \lceil \frac{n}{k} \rceil, \\ (k-1)R + \lceil \frac{n}{k} \rceil - 1 & \text{if } R \geqslant \lceil \frac{n}{k} \rceil, \end{cases}$$

where $n = 2(2^k - 1)$, the number of vertices in $T_{k-1} \cup T_{k-1}$.

Proof. A schedule S gives a sequence of vertices $u_1, u_2, \ldots, u_n$ according to their relative order. Divide this sequence up into *epochs* in the following way. Place u_1 in the first epoch. Continue with the vertices in sequence, adding them to the first epoch until the interval of the next vertex overlaps the interval of a vertex already in the epoch.

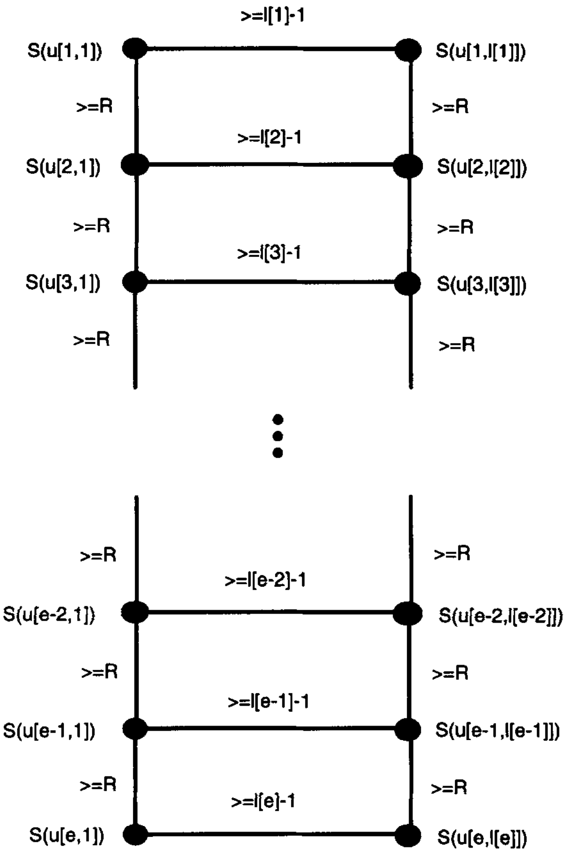


Fig. 2.

Use this vertex to start the second epoch, and continue to add vertices to this epoch in the same way. When done, there will be some number of epochs, e, such that any pair of vertices in the same epoch will not overlap. Call the length of the ith epoch l_i, the first vertex of the ith epoch u_{i1}, and the last vertex of the epoch u_{il_i}.

Some simple facts about S can be determined from this decomposition into epochs. For instance, since u_{i1} overlaps some interval in the previous $(i-1)$st epoch, it must be the case the $S(u_{i1}) - S(u_{i-1,1}) \geqslant R$. Also, it must be the case that $S(u_{il_i}) - S(u_{i1}) \geqslant l_i - 1$, by repeated application of condition (2) for a schedule. Thus, we have the ladder of difference bounds illustrated in Fig. 2 (except for the bounds on the right which will be justified shortly). It might be helpful at this point to glance at Fig. 3 which illustrates a schedule for $T_5 \cup T_5$. In effect, we will argue that a minimum schedule must look something like this schedule: a sequence of k epochs, all essentially the same length.

We now consider three cases according to the number of epochs e as compared to k.

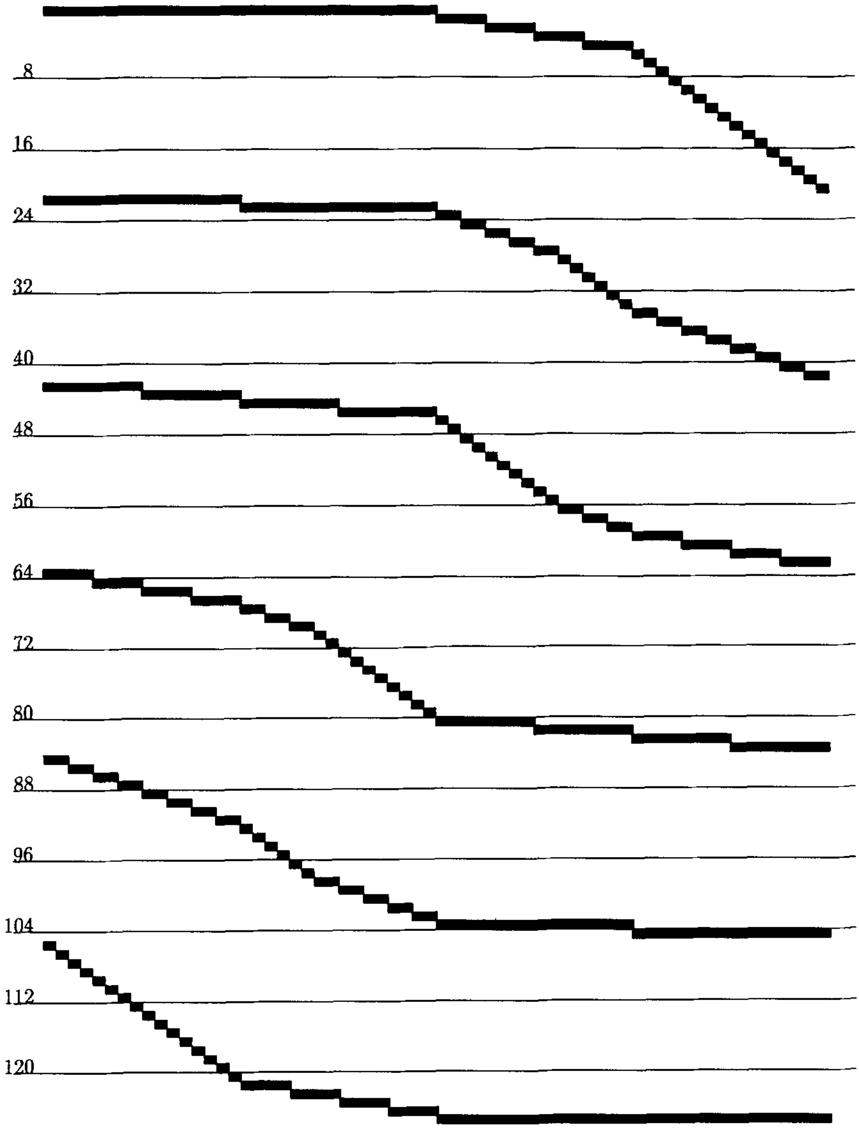


Fig. 3.

It is not possible that e is less than k. Pick any path from root to leaf in one of the trees. This gives k mutually overlapping intervals, no two of which can be in the same epoch, so there must be at least k epochs.

The interesting case is $e = k$. If there are exactly k epochs then it must be the case that every vertex v overlaps some vertex in every other epoch. This can be seen by considering a path from a root to a leaf that passes through v. This path consists of k mutually overlapping vertices that must be in different epochs, and thus v overlaps a vertex in every other epoch. From this we can deduce that $S(u_{il_i}) - S(u_{i-1,l_{i-1}}) \geqslant R$.

The simplest way to see this is to reverse the schedule and apply the reasoning used to determine that $S(v_{i1}) - S(v_{i-1,1}) \geqslant R$, which now applies since we know that $v_{i-1,l_{i-1}}$ must overlap some vertex in epoch i. We now seek the longest forward path in the ladder diagram of Fig. 2 which corresponds to a telescoping sum that gives a lower bound on $S(u_{el_e}) - S(u_{11}) = |S|$. Such a path will go down the left side, cut across to the right, and then continue down to the bottom. The total distance spent travelling down the left side and the right side is $(k-1)R$. Since there are n vertices that must fit in k epochs, one of the epochs is at least $\lceil n/k \rceil - 1$ long, and we will cut across to the right at the longest epoch which is at least this long. Thus $|S|$ is at least $(k-1)R + \lceil n/k \rceil - 1$, and the theorem holds for $R \geqslant \lceil n/k \rceil$.

This bound can be strengthened in two ranges of R. For any R, the trivial lower bound of $n-1$ applies, and we rely on this for $R \leqslant \lfloor n/k \rfloor$. For $\lfloor n/k \rfloor < R < \lceil n/k \rceil$ the situation is more delicate. Augment the ladder of bounds on the left and right side with constraints based on the lengths of the epochs: $S(u_{i+1,1}) - S(u_{i1}) \geqslant l_i$ and $S(u_{i+1,l_{i+1}}) - S(u_{il_i}) \geqslant l_{i+1}$. Call an epoch i long if its length l_i is greater than R, and call it short otherwise. Since $\lfloor n/k \rfloor < R$, some epoch must be long. Follow the left side of the ladder down to the first long epoch, cut across to the right side, and then down the remainder of the right side. Using the newly added bounds, we see that $|S|$ is at least

$$\left(\sum_{i \text{ long}} l_i\right) + \left(\sum_{i \text{ short}} R\right) - 1.$$

We now argue that this is at least $(n - k\lfloor n/k \rfloor)(\lceil n/k \rceil - R) + kR - 1$. Consider adding the n intervals to the k epochs without regard to any constraint other than minimizing the above sum. Every epoch will add at least R to the sum so, without loss of generality, assume that each of the k epochs has at least $\lfloor R \rfloor = \lfloor n/k \rfloor$ intervals. This leaves $n - k\lfloor n/k \rfloor$ intervals to be accounted for. Each of these will add at least $\lceil n/k \rceil - R$ to the sum. The bound now easily follows.

The remaining case is $e > k$. If there are more than k epochs, then it is no longer the case that every vertex overlaps some vertex in every other epoch. Thus, the "right side" difference bounds $(S(u_{il_i}) - S(u_{i-1,l_{i-1}}) \geqslant R)$ are not valid in this case, but the "left side" bounds still hold by the construction of the epochs. For $R > \lceil n/k \rceil$ the lower bound we wish to establish is $(k-1)R + \lceil n/k \rceil - 1$. The path down the left side of the ladder gives us $|S| \geqslant kR$ since there are at least $k+1$ epochs. If $\lfloor n/k \rfloor < R < \lceil n/k \rceil$ then we again use the bounds based on the length of the intervals. For each of the first k epochs call it long if it has at least $\lceil R \rceil$ intervals; short otherwise. If we go down the left side we get the lower bound.

$$\left(\sum_{i \text{ long}} l_i\right) + \left(\sum_{i \text{ short}} R\right).$$

If all the epochs were short this sum would be kR. But this leaves $n - k\lfloor n/k \rfloor$ intervals unaccounted for. Placing them all in the $(k+1)$st epoch or in subsequent epochs would increase our bound on $|S|$ by 1 for each beyond the first. Placing one in a short epoch thereby making it long adds $\lceil n/k \rceil - R \leqslant 1$ to $|S|$. Thus, $|S|$ is at least $(n - k\lfloor n/k \rfloor - 1)(\lceil n/k \rceil - R) + kR$. This is larger than the lower bound we wish to prove when $R = \lfloor n/k \rfloor$, and comparison of the derivatives of the bounds with respect to R shows that it remains so in the interval $\lfloor n/k \rfloor \leqslant R \leqslant \lceil n/k \rceil$. □

4. An upper bound and an algorithm

In this section we give a matching upper bound to the lower bound of the previous section. The lower bound proof actually tells us quite a lot about what we are looking for in the way of a schedule: a sequence of k epochs, all of essentially the same length, each of which covers the entire unit interval. Fig. 3 illustrates one such schedule for $T_5 \cup T_5$, where $k = 6$ and $R = \lfloor n/k \rfloor = 21$. What we will do in general is to find k epochs of approximately equal size such that each epoch consists of vertices of one subtree (the "left" subtree of intervals with right endpoint less than $\frac{1}{2}$) followed by vertices of the other subtree (the "right" subtree of intervals with left endpoint greater than or equal to $\frac{1}{2}$) with the intervals of each epoch covering the unit interval from left to right.

Actually, more than just a size constraint will be important in constructing optimal schedules. It will also be required that if epoch $i + 1$ is started R units after the start of epoch i, that it can proceed with each vertex one unit after the previous vertex such that for every overlapping pair of intervals from the epochs in the left subtree they are scheduled at least R units apart. This separation constraint is, of course, the crux of the problem. Think of an epoch as a runner sweeping across the unit interval, sometimes very fast with long intervals, sometimes slow with short intervals. To insure the separation constraint, we will arrange that within the left subtree the $(i + 1)$st epoch initially runs slower than does the ith epoch. Further, if it ever runs faster than the previous interval, then it will continue to run faster. Thus, if the runners are started at the same time and the second runner takes at least as long as the first to run the whole interval, then the first runner will never be overtaken by the second. If the first runner is started R units before the second, it will always remain at least R units ahead. Separation in the second subtree will be insured by symmetry.

Call A_i the sequence of vertices from the left subtree in the $(k + 1 - i)$st epoch, and B_i the sequence of vertices from the right subtree in the $(k + 1 - i)$st epoch. We must have $\sum_i(|A_i| + |B_i|) = n$. If it can be arranged that $|A_i| = |B_{k+1-i}|$, then, as it is made clear by the rotational symmetry of Fig. 3, it suffices to find only the A_i. We also wish to have the epochs as equal in size as possible. The best that could be hoped for is to find $r = n - k\lfloor n/k \rfloor$ epochs of size $\lceil n/k \rceil$ and the remaining $k - r$ of size $\lfloor n/k \rfloor$. It turns out to be always possible to accomplish this.

Definition 1. Let

$$|A_i| = \begin{cases} \lfloor \frac{n}{k} \rfloor + \delta_i - 2^{i-1} & \text{if } 1 \leqslant i \leqslant \frac{k}{2}, \\ \frac{1}{2}\lfloor \frac{n}{k} \rfloor & \text{if } i = \frac{k+1}{2},\ r \text{ even, } k \text{ odd}, \\ \frac{1}{2}(\lfloor \frac{n}{k} \rfloor + 1) & \text{if } i = \frac{k+1}{2},\ r \text{ odd, } k \text{ odd}, \\ 2^{k-i} & \text{if } \frac{k}{2} < i \leqslant k, \end{cases}$$

and $|B_i| = |A_{k+1-i}|$, where $\delta_i = 1$ if i is less than or equal to $r/2$ for $r = n - k\lfloor n/k \rfloor$, and $\delta_i = 0$ otherwise.

By noting that the remainder r is odd if and only if both k and $\lfloor n/k \rfloor$ are odd, it is not difficult to see that the sizes in Definition 1 are integral. Note that $|A_i| + |B_i| = \lfloor n/k \rfloor + 1$ for $i \leqslant r/2$ and $i \geqslant k + 1 - r/2$, $|A_i| + |B_i| = \lfloor n/k \rfloor + 1$ for $i = (k+1)/2$ and r odd, and $|A_i| + |B_i| = \lfloor n/k \rfloor$ otherwise. Thus, we see that $\sum_i (|A_i| + |B_i|) = n$.

Definition 2. Call a sequence of vertices $v_1, v_2, \dots, v_i$ in a complete binary tree of height h a *monotone cut* if

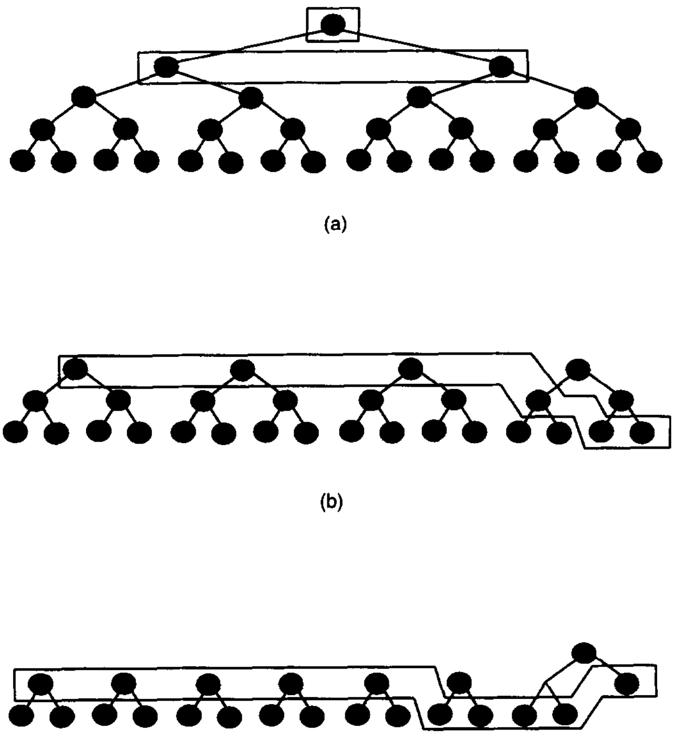
(1) the interval corresponding to v_{i+1} is to the right of that of v_i,
(2) the length of the interval of v_{i+1} is at most as large as that of v_i, and
(3) every path from root to leaf has a vertex in the cut (that is, it goes left to right, top to bottom, and covers the entire interval corresponding to the root of the tree).

The following simple lemma is central to the proof that epochs of the desired size can always be found.

Lemma 3. *Let T be a complete binary tree of height h. If $1 \leqslant x \leqslant 2^h$, then there is a monotone cut of size x.*

Proof. By induction on h. The case $h = 0$ is trivial. For a given h, if $x = 1$ or $x = 2^h$ the monotone cut will be the root or the leaves, respectively. If $1 < x \leqslant 2^{h-1} + 1$, then select the left child of the root and, by induction, a monotone cut of size $x - 1$ from the right subtree. The resulting sequence is a monotone cut. For $x > 2^{h-1} + 1$ select a monotone cut of size $x - 2^{h-1}$ in the left subtree (by induction) followed by all the leaves in the right subtree. $\square$

We now begin to construct the A_i. Initially, we are faced with the complete left subtree. We will first extract the vertices corresponding to A_k, then A_{k-1}, and so on, until finally being left with only A_1. Let $F_k = T_{k-1}$ and for $i = k-1, k-2, \dots, 1$, define F_i to be the forest which is the diagram of the intervals remaining after $A_k, A_{k-1}, \dots, A_{i+1}$ have been removed. That is, F_{i-1} is obtained from F_i by removing the vertices of A_i.



(a)

(b)

(c)

Fig. 4.

A_k, which is of size 1, can only be the root of the left subtree, which is the root of F_k. F_{k-1} consists of two complete binary trees of height $k-2$, and since $|A_{k-1}| = 2$, we take A_{k-1} to be the roots of the two trees. For $i > \lceil k/2 \rceil$, we continue this way, taking A_i to be the 2^{k-i} roots of F_i, which correspond to the first $\lfloor k/2 \rfloor$ levels of T_{k-1}.

The key to showing that we can continue to extract the A_i for $i \leqslant \lceil k/2 \rceil$ is that the diagram F_i of the remaining elements consists of trees with height $i-1$, *almost all* of which are binary. For A_i, we will select one of these subtrees as a "center". The roots of all trees to the left of the center will be selected, a monotone cut of an appropriate size will be selected in the center, and all the leaves of the trees to the right will be selected. If the center is binary, then Lemma 3 can be used to insure that a monotone cut of the appropriate size can be found in the center. *It is important to note that relative to T_{k-1} the A_i do not have such a simple monotone structure, but relative to F_i, A_i is monotone.* Fig. 4 shows how this procedure works for $k = 5$. Fig. 4(a) shows $T_4 = F_5$ with the sequences A_5 and A_4, (b) shows F_3 with A_3, and (c) shows F_2 and A_2. The remaining vertices of F_2 form A_1.

Let N_i denote the number of subtrees in the diagram F_i *and* f_i^j denote the number of leaves in the jth tree from the right in F_i. Let R_i be such that

$$\sum_{j=1}^{R_i-1} f_i^{(j)} + (N_i - R_i + 1) < |A_i| \leqslant \sum_{j=1}^{R_i} f_i^{(j)} + (N_i - R_i),$$

that is, the least value such that taking all the leaves of the "rightmost" R_i subtrees together with the roots of the rest is not too small. Such an R_i exists because $|A_i| \leqslant |F_i|$.

Let $L_i = N_i - R_i$ for $i \leqslant \lceil k/2 \rceil$. In F_i, A_i will consist of the roots of the L_i subtrees to the left of center, a monotone chain in the R_ith subtree from the right (the center), and the leaves of the $R_i - 1$ subtrees to the right of center. Our goal is to show that the center is binary, so that Lemma 3 can be applied to insure that $|A_i|$ is the proper size. Let $C_i = |A_i| - \sum_{j=1}^{R_i-1} f_i^{(j)} + (N_i - R_i + 1)$ denote the size of the monotone cut needed in the center subtree.

Observe that removing $|A_i|$ from F_i produces F_{i-1} by creating two new trees (of height $i-2$) beneath each of the L_i roots which are removed, a new possibly nonbinary tree of height $i-2$ from the center subtree and trees of height $i-2$ in the remainder. The number of subtrees to the right of the center $R_i - 1$ is unchanged. At most one new nonbinary tree is formed at each step, so the total number of nonbinary subtrees in F_i is at most $\lceil k/2 \rceil - i$. Furthermore, if all nonbinary trees in F_i were among the $R_i - 1$ rightmost subtrees then all nonbinary trees in F_{i-1} will be among the R_i rightmost subtrees. Thus, it is enough to show that $R_{i-1} > R_i$ to insure that the center in F_i will be binary. We obtain upper and lower bounds on R_i to show that this holds. First we get a bound on N_i which appears in the bounds for R_i.

Lemma 4.

$$N_i \leqslant \frac{i\lfloor n/k \rfloor + i - 2^i + 1}{2^i - 1}.$$

Proof. Each of the trees in F_i has at least $2^i - 1$ elements. Then,

$$N_i(2^i - 1) \leqslant \sum_{j=1}^{N_i} f_i^j = \sum_{k=1}^{i} |A_k| \leqslant i\lfloor n/k \rfloor + i - 2^i + 1. \qquad \square$$

Lemma 5.

$$R_i \geqslant \frac{\lfloor n/k \rfloor - N_i - (\lceil k/2 \rceil - i)2^{i-1}}{2^{i-1} - 1}.$$

Proof. Use $N_i - R_i = L_i$, the size of A_i in terms of subtrees in F_i, the fact that each binary tree in F_i has height $i-1$ and thus 2^{i-1} leaves, and each of the at most $(\lceil k/2 \rceil - i)$ nonbinary trees has at most twice this number of leaves. (The last observation can be seen by noticing that these trees come from trees of height i with a monotone cut removed.) Then,

$$\begin{aligned}\lfloor n/k \rfloor - 2^{i-1} &\leqslant |A_i| \\ &= L_i + \sum_{k=1}^{R_i - 1} f_i^k + C_i \\ &\leqslant N_i - R_i + R_i 2^{i-1} + (\lceil k/2 \rceil - i)2^{i-1}. \qquad \square\end{aligned}$$

Lemma 6.

$$R_j \leqslant \frac{\lfloor n/k \rfloor - L_j}{2^{j-1}}.$$

Proof. Using the same bounds as in the previous lemma,

$$\begin{aligned}\lfloor n/k \rfloor + 1 - 2^{j-1} &\geqslant |A_j| \\ &= L_j + \sum_{k=1}^{R_j - 1} f_i^k + C_j \\ &\geqslant L_j + (R_j - 1)2^{j-1} + 1. \qquad \square\end{aligned}$$

Theorem 7. *There exist A_i, of the sizes in Definition 1, such that in F_i, A_i, is a sequence of roots followed by a monotone chain in a complete binary tree, followed by all the leaves of the remaining trees.*

Proof. This is easily seen to be so for $i > k/2$. Assume that each subtree in F_i has height $i - 1$ and that "almost all" of these are complete binary trees. That is, there are at most $\lceil k/2 \rceil - i$ nonbinary subtrees and these are among the rightmost R_{i+1} subtrees. Furthermore, assume that the nonbinary subtrees have at most 2^i leaves. From the discussion preceding the lemmas, it is enough to show that $R_i > R_{i+1}$. If this is the case, we can find A_i and additionally, F_{i-1} will have the properties noted above.

Assume for contradiction that $R_i \leqslant R_{i+1}$. We will proceed with some detail in order to cover the cases when i is small. From Lemmas 4, 5–6 we get

$$\frac{\lfloor n/k \rfloor - N_i - (\lceil k/2 \rceil - i)2^{i-1}}{2^{i-1} - 1} \leqslant \frac{\lfloor n/k \rfloor}{2^i}$$

$$\Leftrightarrow 2(\lfloor n/k \rfloor - N_i - (\lceil k/2 \rceil - i)2^{i-1}) \leqslant \left(1 - \frac{1}{2^{i-1}}\right)\lfloor n/k \rfloor$$

$$\Leftrightarrow \lfloor n/k \rfloor\left(1 + \frac{1}{2^{i-1}}\right) \leqslant 2N_i + (\lceil k/2 \rceil - i)2^i$$

$$\Rightarrow \lfloor n/k \rfloor\left(1 + \frac{1}{2^{i-1}}\right) \leqslant 2\,\frac{i\lfloor n/k \rfloor + i - 2^i + 1}{2^i - 1} + (\lceil k/2 \rceil - i)2^i$$

$$\Rightarrow \lfloor n/k \rfloor\left(1 + \frac{1}{2^{i-1}} - \frac{2i}{2^i - 1}\right) \leqslant \frac{2i - 2^{i+1} + 2}{2^i - 1} + (\lceil k/2 \rceil - i)2^i. \qquad (1)$$

Since $\lfloor n/k \rfloor = \lfloor (2^{k+1} - 2)/k \rfloor \geqslant (2^{k+1} - 2)/k - 1$, (1) implies

$$\left(\frac{2^{k+1}}{k} - \frac{2+k}{k}\right)\left(1 + \frac{1}{2^{i-1}} - \frac{2i}{2^i - 1}\right) \leqslant \frac{2i - 2^{i+1} + 2}{2^i - 1} + (\lceil k/2 \rceil - i)2^i.$$

For integral $i \geqslant 2$, $k \geqslant 2$, one can easily check that

$$-\frac{2+k}{k}\left(1 + \frac{1}{2^{i-1}} - \frac{2i}{2^i - 1}\right) \geqslant \frac{2i - 2^{i+1} + 2}{2^i - 1}.$$

Thus, the following is implied:

$$\frac{2^{k+1}}{k}\left(1 + \frac{1}{2^{i-1}} - \frac{2i}{2^i - 1}\right) \leqslant (\lceil k/2 \rceil - i)2^i$$

$$\Rightarrow \left(1 + \frac{1}{2^{i-1}} - \frac{2i}{2^i - 1}\right) \leqslant \frac{k}{2^{k+1}}(\lceil k/2 \rceil - i)2^i$$

$$\Rightarrow \frac{1}{2} \leqslant \frac{k}{2^{k+1}}(\lceil k/2 \rceil - i)2^i.$$

Simple calculus shows that for a constant a, the function $(a - i)2^i$ is maximized with respect to i at $a - (1/\ln 2)$ with value $2^a/(e \ln 2) \approx (0.531)2^a$. Thus, we have

$$\frac{1}{2} \leqslant \frac{k}{2^{k+1}}(0.54)2^{\lceil k/2 \rceil} \Rightarrow 1 \leqslant 0.54\frac{k}{2^{\lceil k/2 \rceil}}.$$

For $i \geqslant 2$ and $k \geqslant 2$ this is a contradiction. The last step removing A_1 from F_1 is trivial as F_1 consists of isolated vertices. □

We now show that these epochs will satisfy the separation constraints. Let $A_i[j]$ denote the jth element of A_i.

Lemma 8. *Suppose* $S(A_i[j]) = S(A_i[j-1]) + 1$, $1 < j \leqslant |A_i|$, *and* $S(A_{i+1}[j]) = S(A_{i+1}[j-1]) + 1, i < j \leqslant |A_{i+1}|$.*Then if* $S(A_i[1]) \geqslant S(A_{i+1}[1]) + R, S(A_i[q]) \geqslant S(A_{i+1}[r]) + R$ *for any pair of intersecting intervals* $A_i[q]$ *and* $A_{i+1}[r]$.

Proof. First note that $|A_{i+1}| \leqslant |A_i|$ by Definition 1. Define two flows on the half unit interval, $v_i(p) = 1/l_i(p)$ where $l_i(p)$ is the length of the interval containing position p in A_i, and $v_{i+1}(p) = 1/l_{i+1}(p)$ where $l_{i+1}(p)$ is the length of the interval containing position p in A_{i+1}. Integrate these flows to get the functions $t_i(p)$ such that $t_i(0) = S(A_i[1])$ and $t_{i+1}(p)$, such that $t_{i+1}(0) = S(A_{i+1}[1])$. These are nothing more than a piecewise linear interpolation of the schedule S for A_i and A_{i+1}. The utility of these functions is that $\min\{t_i(p) - t_{i+1}(p)\}$ is a lower bound on the separation of the two intervals. This is illustrated by Fig. 5.

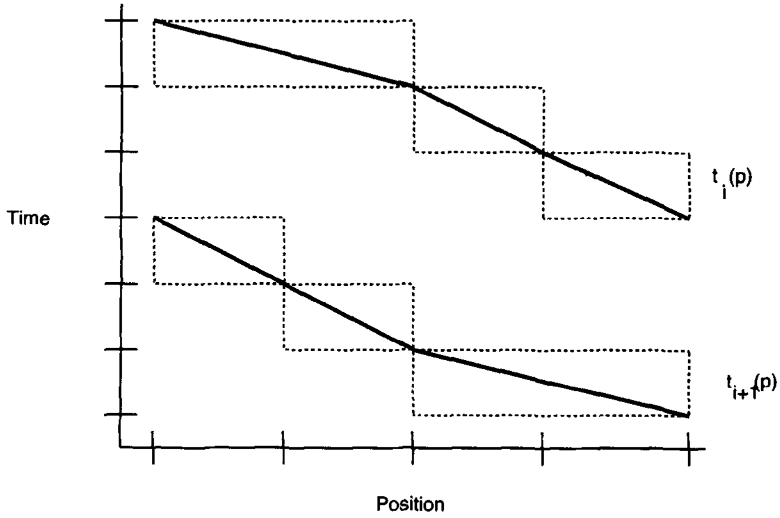


Fig. 5.

By the monotone construction of A_{i+1} and A_i in F_{i+1} and F_i it follows that $v_i(0) > v_{i+1}(0)$, and if $v_i(p_0) \leqslant v_{i+1}(p_0)$, then $v_i(p) \leqslant v_{i+1}(p)$ for all $p \geqslant p_0$. Suppose for $0 < p_1 < \frac{1}{2}$ it was the case that $t_i(p_1) - t_{i+1}(p_1) \leqslant R$. Then $v_i(p_1) < v_{i+1}(p_1)$, and hence $v_i(p) < v_{i+1}(p)$ for all $p \geqslant p_1$. Thus, we conclude that $t_i(\frac{1}{2}) - t_{i+1}(\frac{1}{2}) < R$. But

$$\begin{aligned} t_i(\tfrac{1}{2}) - t_{i+1}(\tfrac{1}{2}) &= (S(A_i[1]) + |A_i|) - (S(A_{i+1}[1]) + |A_{i+1}|) \\ &\geqslant (S(A_{i+1}[1]) + R + |A_i|) - (S(A_{i+1}[1]) + |A_{i+1}|) \\ &\geqslant R. \end{aligned}$$

The last inequality follows since $|A_i| \geqslant |A_{i+1}|$. □

Theorem 9. *There exists a minimum schedule for $T_{k-1} \cup T_{k-1}$ with separation R such that*

$$|S| \geqslant \begin{cases} n-1 & \text{if } R \leqslant \lfloor \frac{n}{k} \rfloor, \\ (n - k\lfloor \frac{n}{k} \rfloor)(\lceil \frac{n}{k} \rceil - R) + kR - 1 & \text{if } \lfloor \frac{n}{k} \rfloor < R < \lceil \frac{n}{k} \rceil, \\ (k-1)R + \lceil \frac{n}{k} \rceil - 1 & \text{if } R \geqslant \lceil \frac{n}{k} \rceil, \end{cases}$$

where $n = 2(2^k - 1)$, the number of vertices in $T_{k-1} \cup T_{k-1}$.

Proof. Let $S(A_k[1]) = 0$, $S(A_i[j+1]) = S(A_i[j]) + 1$, $S(A_{i-1}[1]) = S(A_i[1]) + \max\{R, l_i\}$, and $S(B_{k-i}[j]) = |S| - S(A_i[j])$, where $|S|$ is as in the statement of the theorem and $l_i = |A_i| + |B_i|$. It is clearly minimum by the lower bound. That it is a schedule follows from the previous theorem, and the fact that for every i, the last element of A_i is scheduled at least 1 unit before the first element of B_i. □

Theorem 10. *An optimal schedule for $T_{k-1} \cup T_{k-1}$ with separation R can be found in $\mathrm{O}(n)$ time, where $n = 2(2^k - 1)$, the number of vertices in $T_{k-1} \cup T_{k-1}$.*

Proof. Keep a data structure representing F_i such that every tree has the leaves threaded and every vertex had a pointer to its parent. Then A_i can easily be found in $\mathrm{O}(n)$ time by the method of Theorem 7. It is then trivial to apply Theorem 9. □

Finally, we restate Theorem 9 in terms of the graph theoretical separation problem. Recall that $I_{k-1} \cup I_{k-1}$ denotes the interval graph that is the comparability graph of the diagrams $T_{k-1} \cup T_{k-1}$. Alternatively, $I_{k-1} \cup I_{k-1}$ is the intersection graph of the intervals $[j2^{-i}, (j+1)2^{-i})$, for $0 \leqslant j \leqslant 2^i - 1$ and $1 \leqslant i \leqslant k$.

Theorem 11. *The separation of $I_{k-1} \cup I_{k-1}$ is $\lfloor k/2 \rfloor$ where $n = 2(2^k - 1)$, the number of vertices in $I_{k-1} \cup I_{k-1}$.*

Proof. Same as Theorem 9 except delete the R from $\max\{R, l_i\}$ when defining $S(A_{i-1}[1]) = S(A_i[1]) + \max\{R, l_i\}$. □

5. Conclusion

The scheduling problem considered here arises from using the Haar wavelet basis in a particular diagnostic application. Many other scheduling problems arise from other wavelet bases and diagnostic procedures.

In our case, the Haar basis is supported on the dyadic intervals, but other wavelet bases have different support, and thus scheduling algorithms for these would be of use.

In the Haar case, the separation constraint we used fairly completely model the physics of the problem, for other bases the true separation constraint is more complex. Essentially, for each point on the unit interval, the separation constraint for that point after an inner product measurement depends on the intensity of the function with which the inner product is being measured at that point. For many wavelet bases, a clinically useful schedule will have to take this into account.

For imaging in higher dimensions is would be useful to find good schedules for 2-dimensional wavelet bases.

Acknowledgements

The work of the first two authors was supported in part by a grant from DARPA, as monitored by the AFOSR under contract AFOSR-90-0291. Part of this research was carried out while the first author was visiting the School of Computer Science at

Carnegie Mellon University and while the first and third authors were at Dartmouth College.

References

[1] C.C. Han and K.J. Lin, Job scheduling with separation constraints, UIUCDCS-R-90-1635, University of Illinois at Urbana-Champaign, IL (1990).
[2] D.M. Healy, J.B. Weaver, Yansun Xu and J.R. Driscoll, Wavelet encoded MR imaging, Magnetic Resonance Med. 24 (1992) 275–287.
[3] J.Y.-T. Leung, O. Vornberger and J.D. Witthoff, On some variants of the bandwidth minimization problem, SIAM J. Comput. 13 (1984) 650–667.
[4] Z. Miller and D. Pritikin, On the separation number of a graph, Networks 19 (1989) 651–666.