Optimal Multileaf Collimator Leaf Sequencing in IMRT Treatment Planning

Z. Caner Taşkın* J. Cole Smith† H. Edwin Romeijn‡
James F. Dempsey§

June 21, 2009

Abstract

We consider a problem dealing with the efficient delivery of Intensity Modulated Radiation Therapy (IMRT) to individual patients. IMRT treatment planning is usually performed in three phases. The first phase determines a set of beam angles through which radiation is delivered, followed by a second phase that determines an optimal radiation intensity profile (or fluence map). This intensity profile is selected to ensure that certain targets receive a required amount of dose while functional organs are spared. In order to deliver these intensity profiles to the patient, a third phase must decompose them into a collection of apertures and corresponding intensities. In this paper, we investigate this last problem. Formally, an intensity profile is represented as a nonnegative integer matrix; an aperture is represented as a binary matrix whose

*Department of Industrial and Systems Engineering, University of Florida, 303 Weil Hall, P.O. Box 116595, Gainesville, Florida 32611-6595; e-mail: taskin@ufl.edu.

†Department of Industrial and Systems Engineering, University of Florida, 303 Weil Hall, P.O. Box 116595, Gainesville, Florida 32611-6595; e-mail: cole@ise.ufl.edu.

‡Department of Industrial and Operations Engineering, The University of Michigan, 1205 Beal Avenue, Ann Arbor, Michigan 48109-2117; e-mail: romeijn@umich.edu. The work of this author was supported by the National Science Foundation under grant no. DMI-0457394.

§ViewRay Inc., 2 Thermo Fisher Way, Village of Oakwood, Ohio 44146; e-mail: jfdempsey@viewray.com. The work of this author was supported by the National Science Foundation under grant no. DMI-0457394.

ones appear consecutively in each row. A feasible decomposition is one in which the original desired intensity profile is equal to the sum of a number of feasible binary matrices multiplied by corresponding intensity values. In order to most efficiently treat a patient, we wish to minimize a measure of total treatment time, which is given as a weighted sum of the number of apertures and the sum of the aperture intensities used in the decomposition. We develop the first exact algorithm capable of solving real-world problem instances to optimality within practicable computational limits, using a combination of integer programming decomposition and combinatorial search techniques. We demonstrate the efficacy of our approach on a set of 25 test instances derived from actual clinical data and on 100 randomly generated instances.

1 Introduction and Literature Survey

Cancer is one of the leading causes of death throughout the world. In the last century, external beam radiation therapy has emerged as a very important and powerful modality for treating many forms of cancer, either in primary form or in conjunction with other treatment modalities such as surgery, chemotherapy, or medication. In the United States today, approximately two-thirds of all newly diagnosed cancer patients receive radiation therapy for treatment. Since the radiation beams employed in radiation therapy damages all cells traversed by the beams, both in targeted areas in the patient that contain cancerous cells as well as any cells in healthy organs and tissues, the treatment must be carefully designed. This can partially be achieved by delivering radiation from several different directions, also called beam orientations. Therefore, patients receiving radiation therapy are typically treated on a clinical radiation-delivery device that can rotate around the patient. The most common device is called a linear accelerator, and is typically equipped with a so-called *multileaf collimator (MLC)* system which can be used to judiciously shape the beams by forming apertures, thereby providing a high degree of control over the dose distribution that is received by a patient (see Figure 1). This technique has been named *intensity modulated radiation therapy (IMRT)*.

Since the mid 1990's, large-scale optimization of the fluence applied from a number of

(a) 

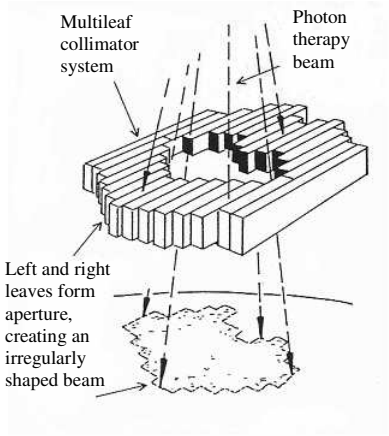
(b) 

Figure 1: (a) A multileaf collimator system[1]; (b) the projection of an aperture onto a patient.

beam orientations around a patient has been used to design treatments from MLC-equipped linear accelerators. A typical approach to IMRT treatment planning is to first select the number and orientations of the beams to use as well as an intensity profile or *fluence map* for each of these beams, where the fluence map takes the form of a matrix of intensities.

This problem has been studied extensively and can be solved satisfactorily, in particular when (as is common in clinical practice) the beam orientations are selected manually by the physician or clinician based on their insight and expertise regarding treatment planning. For optimization approaches to the fluence map optimization problem with fixed beam orientations we refer to the review paper by Shepard et al. [35]. More recently, Romeijn et al. [33] proposed new convex programming models, and Hamacher and Küfer [14] and Küfer et al. [23] considered a multi-criteria approach to the problem. Lee et al. [25, 26] studied mixed-integer programming approaches to the extension of the fluence map optimization problem that also optimizes the number and orientations of the beams to be used. However, to enable delivery of the optimal fluence maps by the MLC system, they need to be decomposed into a collection of deliverable apertures. (For examples of integrated approaches to fluence map optimization, also referred to as aperture modulation, we refer to Shepard et al. [34], Preciado-Walters et al. [29], and Romeijn et al. [32].)

The vast majority of MLC systems contain a collection of leaves that can be moved in parallel, thereby blocking part of the radiation beam. This architecture implies that we can

[1]Varian Medical Systems; `http://www.varian.com/orad/prd056.html`.

view each beam as a matrix of beamlets or *bixels* (the smallest deliverable square beam that can be created by the MLC), so that each aperture can be represented by a collection of rows (or, by rotating the MLC head, columns) of bixels, each of which should be convex. In other words, each fluence map should be decomposed into either constant-intensity row-convex apertures or constant-intensity column-convex apertures. Due to the time required for setup and verification, clinical practice prohibits using both types of apertures for a given fluence map, so that without loss of generality we will in this paper focus on row-convex apertures only. Note that while some manufacturers of MLC systems impose additional constraints on the apertures, we will assume throughout this paper that all row-convex apertures are deliverable. As an example, consider the fluence map given by the following 2×3 matrix of bixel intensities (see Baatar [5]):

$$\begin{bmatrix} 3 & 6 & 4 \\ 2 & 1 & 5 \end{bmatrix}.$$

If we represent an aperture by a binary matrix in which an element is equal to one if and only if the associated bixel is exposed (i.e., not blocked by either the left or right leaf of the MLC system), row-convexity corresponds to the property that, in each row of the corresponding matrix, the elements that are equal to one are consecutive (often referred to as the *consecutive ones property*). Now observe that this fluence map can be decomposed into three apertures with corresponding intensities:

$$\begin{bmatrix} 3 & 6 & 4 \\ 2 & 1 & 5 \end{bmatrix} = 1 \times \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 1 \end{bmatrix} + 2 \times \begin{bmatrix} 1 & 1 & 0 \\ 1 & 0 & 0 \end{bmatrix} + 4 \times \begin{bmatrix} 0 & 1 & 1 \\ 0 & 0 & 1 \end{bmatrix}.$$

Since, in general, there are many ways of decomposing a given fluence map into row-convex apertures, it is desirable to select the decomposition that can be delivered most efficiently. The two main efficiency criteria that play a role are the total *beam-on-time*, i.e., the total amount of time that the patient is being irradiated, and the *total setup time*, i.e., the total amount of time that is spent shaping the apertures. The former metric is proportional to the sum of intensities used in the decomposition, while the latter is approximately proportional to the number of matrices used in the decomposition. Although closely related, these two efficiency criteria are not equivalent. The example given above shows the unique

decomposition using only three apertures and with a beam-on-time of 7. However, the minimum beam-on-time for this fluence map is 6, which can be realized by four apertures using the following decomposition:

$$\begin{bmatrix} 3 & 6 & 4 \\ 2 & 1 & 5 \end{bmatrix} = 1 \times \begin{bmatrix} 1 & 1 & 0 \\ 1 & 0 & 0 \end{bmatrix} + 1 \times \begin{bmatrix} 0 & 1 & 0 \\ 1 & 1 & 1 \end{bmatrix} + 2 \times \begin{bmatrix} 1 & 1 & 1 \\ 0 & 0 & 1 \end{bmatrix} + 2 \times \begin{bmatrix} 0 & 1 & 1 \\ 0 & 0 & 1 \end{bmatrix}.$$

The problem of decomposing a fluence map while minimizing beam-on-time is polynomially solvable and has been widely studied, leading to several solution approaches for this problem. Bortfeld et al. [7] proposed the sweep method, which Ahuja and Hamacher [2] (who derived an equivalent method) showed to indeed yield an optimal solution; other exact algorithms were proposed by Kamath et al. [18], and Siochi [36]. In addition, Baatar et al. [5], Boland et al. [6], Kalinowski [16], Kamath et al. [19, 20, 21, 22], Lenzen [27], and Siochi [36] studied the problem of minimizing beam-on-time under additional hardware constraints, while Kalinowski [17] studied the benefits of allowing rotation of the MLC head.

Although the time required by the MLC system to transition between apertures formally depends on the apertures themselves, the fact that these times are similar and the presence of significant (aperture-independent) verification and recording overhead times justifies the use of the total number of setups (or, equivalently, the total number of apertures) to measure the total setup time. In addition, delivering IMRT with a small number of apertures provides the additional benefits of less wear-and-tear on the collimators (less stopping and starting) and a less error-prone delivery as IMRT delivery errors are known to be proportional to the number of apertures (see Stell et al. [39]). The problem of decomposing a fluence map into the minimum number of row-convex apertures has been shown to be strongly NP-hard (see Baatar et al. [5]), leading to the development of a large number of heuristics for solving this problem. Notable examples are the heuristics proposed by Baatar et al. [5] (who also identify some polynomially solvable special cases), Agazaryan and Solberg [1], Dai and Zhu [8], Que [30], Que et al. [31], Siochi [36, 37, 38], Van Santvoort and Heijmen [40], Xia and Verhey [41]. In addition, Engel [11], Kalinowski [16], and Lim and Choi [28] developed heuristics to minimize the number of apertures while constraining the total beam-on-time to be minimal. Finally, Langer et al. [24] developed a mixed-integer programming formulation

of the problem, while Kalinowski [15] proposed an exact dynamic programming approach for the related problem of minimizing the number of apertures that yields the minimum beam-on-time. Baatar et al. [4] described integer programming and constraint programming models for the same problem, and Ernst et al. [12] proposed a constraint programming approach for minimizing the number of apertures. However, computational studies reported in [4, 12, 15, 24] reveal that these approaches can only be used to efficiently solve small problem instances to optimality. The primary contribution of this paper is that we develop the first algorithm capable of solving clinical problem instances to optimality (or to provably near-optimality) within clinically acceptable computational time limits.

In this paper, our focus is on the problem of finding a decomposition of a fluence map into row-convex apertures that minimizes total treatment time, as measured by the sum of the total setup time and beam-on-time. In Section 2 we develop our decomposition-based solution approach. In Section 3 we discuss the application of our algorithm on a collection of clinical and randomly generated test data, and compare its performance with alternative exact and heuristic techniques. We conclude the paper in Section 4.

2 Decomposition Algorithm

Throughout this paper, we will denote the fluence map to be delivered by a matrix $B \in \mathbb{N}^{m \times n}$, where the element at row i and column j, (i, j), corresponds to a bixel with required intensity b_{ij}. Let w_1 be the time required by the MLC to form an aperture and w_2 denote the time required for the delivery of one unit of fluence. We refer to the problem of minimizing the total treatment time, i.e., the sum of the aperture transition times and the total delivery time, as the *optimal leaf sequencing problem*.

We start this section by describing a decomposition framework for the optimal leaf sequencing problem in Section 2.1 and use this to formulate our master problem in Section 2.2. We introduce our subproblem in Section 2.3, prove its complexity, and provide a combinatorial search algorithm for its solution. We then enhance the empirical performance of our decomposition algorithm by introducing classes of valid inequalities to the master problem in Section 2.4, and finally describe an algorithm for constructing a feasible solution with

medically desired properties in Section 2.5.

2.1 Decomposition Framework

To establish motivation for our approach, observe that if the objective is to minimize beam-on-time, the optimal leaf sequencing problem is decomposable by the rows of the fluence map. In particular, if the beam-on-time is minimized for each bixel row, the maximum of the corresponding beam-on-time values is equal to the minimum beam-on-time for the overall fluence map (see, e.g., Ehrgott et al. [10]). However, this approach is not directly applicable when the objective is to minimize the total treatment time.

Even though the optimal leaf sequencing problem is not directly decomposable by rows, the fact that leaves corresponding to different rows can be positioned independently can still be exploited. Denote a particular positioning of left and right leaves for a row as a *leaf position*; an aperture is composed of a leaf position for each row of B. Our main observation is that *given* a collection of intensities, which can be used in apertures that collectively cover the fluence map, the rows are independent of one another. That is, we can determine the leaf positions to be used for covering each row independently, and then form apertures for covering the entire fluence map by combining individual leaf positions for each row that are assigned to the same intensity.

We define an *allowable intensity multiset* to be a collection of (potentially non-unique) intensity values, each of which can be assigned to a single aperture in our solution. We say that an allowable intensity multiset is *compatible* with a row if there exists a feasible decomposition of the row into leaf positions using a subset of that allowable intensity multiset. If an allowable intensity multiset is compatible with all rows, then it corresponds to a feasible decomposition of the fluence map and we call it a *feasible intensity multiset*. As an example, consider the fluence map given by the following 3×3 matrix:

$$B = \begin{bmatrix} 1 & 4 & 8 \\ 3 & 8 & 5 \\ 4 & 5 & 3 \end{bmatrix}. \tag{1}$$

Consider the allowable intensity multiset $\{1, 3, 5\}$. Assigning each of these values to at most

one leaf position, the first row can be decomposed as

$$[1 \;\; 4 \;\; 8] \;\; = \;\; 1 \times [1 \;\; 1 \;\; 0] + 3 \times [0 \;\; 1 \;\; 1] + 5 \times [0 \;\; 0 \;\; 1]\,, \tag{2}$$

so that the allowable intensity multiset is compatible with the first row. Similarly, the second row can be decomposed as

$$[3 \;\; 8 \;\; 5] \;\; = \;\; 3 \times [1 \;\; 1 \;\; 0] + 5 \times [0 \;\; 1 \;\; 1]\,. \tag{3}$$

However, the first bixel in the third row must be covered by two leaf positions assigned to intensities 1 and 3, and the second bixel must be covered by a single leaf position assigned to intensity 5. Therefore, all allowable intensities must be used to cover the first two bixels, and the third bixel with required intensity 3 cannot be covered. Hence, the allowable intensity multiset is not compatible with the third row. Alternatively, consider an allowable intensity multiset that contains the values 1, 3, and 4 for the same fluence map. The rows can be decomposed as

$$\begin{aligned} [1 \;\; 4 \;\; 8] \;\; &= \;\; 1 \times [1 \;\; 1 \;\; 1] + 3 \times [0 \;\; 1 \;\; 1] + 4 \times [0 \;\; 0 \;\; 1]\,, \\ [3 \;\; 8 \;\; 5] \;\; &= \;\; 1 \times [0 \;\; 1 \;\; 1] + 3 \times [1 \;\; 1 \;\; 0] + 4 \times [0 \;\; 1 \;\; 1]\,, \text{ and} \\ [4 \;\; 5 \;\; 3] \;\; &= \;\; 1 \times [0 \;\; 1 \;\; 0] + 3 \times [0 \;\; 0 \;\; 1] + 4 \times [1 \;\; 1 \;\; 0]\,. \end{aligned} \tag{4}$$

Since the allowable intensity multiset is compatible with all rows, it is a feasible intensity multiset having three leaf positions and a beam-on-time of 8. Furthermore, observe that the intensity requirements of the bixels in the first row strictly increase from left to right, implying that a leaf position must start at each bixel. Thus, any feasible decomposition of the first row uses at least three leaf positions, which yields a lower bound on the number of apertures. Also, the largest element of B is 8, which yields a lower bound on the beam-on-time. Since the given decomposition achieves the lower bounds on both objectives, we have an optimal solution to the optimal leaf sequencing problem.

2.2 Master Problem Formulation and Solution Approach

We represent an allowable intensity multiset by an integer vector $\mathbf{x} = (x_1, \ldots, x_L)$, where $L = \max_{i=1,\ldots,m;\, j=1,\ldots,n} b_{ij}$ is the maximum intensity value in the fluence map, and where x_ℓ

is the number of times that intensity value ℓ occurs in the allowable intensity multiset. It is easy to see that, assuming all allowable intensity values are used, the number of apertures and the beam-on-time are, respectively, equal to

$$\sum_{\ell=1}^{L} x_\ell \text{ and } \sum_{\ell=1}^{L} \ell x_\ell. \tag{5}$$

The master problem can therefore succinctly be written as

$$\text{minimize } w_1 \sum_{\ell=1}^{L} x_\ell + w_2 \sum_{\ell=1}^{L} \ell x_\ell \tag{6}$$

subject to

$$\mathbf{x} \quad \text{is compatible with row } i \qquad \forall\, i = 1, \ldots, m \tag{7}$$

$$x_\ell \quad \text{integer} \qquad \forall\, \ell = 1, \ldots, L. \tag{8}$$

Clearly, our model contains the problem of minimizing the number of apertures as a special case by setting $w_1 = 1$ and $w_2 = 0$. Moreover, if we wish to minimize the number of apertures required while limiting the beam-on-time to no more than $\tilde{T}$, we simply add the following constraint to the model:

$$\sum_{\ell=1}^{L} \ell x_\ell \leq \tilde{T}, \tag{9}$$

where of course $\tilde{T}$ cannot be less than the minimum achievable beam-on-time $\tilde{z}$ (which can be found in polynomial time using the algorithms mentioned in Section 1).

In order to formulate our master problem as an integer programming problem, we introduce binary variables $y_{\ell r}$, $\forall\, \ell = 1, \ldots, L$, $r = 1, \ldots, R_\ell$, where $y_{\ell r} = 1$ if and only if $x_\ell = r$, and R_ℓ is an upper bound on the number of apertures having intensity ℓ used in an optimal solution. (We can compute R_ℓ by computing an initial upper bound on the optimal objective function value via any of the heuristics mentioned in Section 1, and then setting R_ℓ to the largest value such that $w_1 R_\ell + w_2 \ell R_\ell$ is no more than this bound.) Using these decision variables, we can reformulate the master problem (MP) as follows:

$$\text{minimize } w_1 \sum_{\ell=1}^{L} x_\ell + w_2 \sum_{\ell=1}^{L} \ell x_\ell \tag{10}$$

subject to

$$\sum_{r=1}^{R_\ell} r y_{\ell r} = x_\ell \qquad \forall\ \ell = 1, \ldots, L \tag{11}$$

$$\sum_{r=1}^{R_\ell} y_{\ell r} \leq 1 \qquad \forall\ \ell = 1, \ldots, L \tag{12}$$

$$\mathbf{x} \quad \text{is compatible with row } i \qquad \forall\ i = 1, \ldots, m \tag{13}$$

$$x_\ell \quad \text{integer} \qquad \forall\ \ell = 1, \ldots, L \tag{14}$$

$$y_{\ell r} \quad \text{binary} \qquad \forall\ \ell = 1, \ldots, L,\ r = 1, \ldots, R_\ell. \tag{15}$$

We will next formulate (13) as a set of linear inequalities by deriving valid inequalities that cut off precisely those vectors $\mathbf{x}$ that violate (13). To this end, consider a particular allowable intensity multiset represented by $\hat{\mathbf{x}}$ that is incompatible with at least one row. It is then clear that we should only consider vectors $\mathbf{x}$ that are different from $\hat{\mathbf{x}}$ in at least one component. We can achieve this by imposing the following constraint:

$$\sum_{\ell=1}^{L} \sum_{\substack{r=1 \\ r \neq \hat{x}_\ell}}^{R_\ell} y_{\ell r} \geq 1. \tag{16}$$

Since all integer solutions except for $\hat{\mathbf{x}}$ satisfy (16), it is indeed a valid inequality. Constraint (16) can be tightened by observing that if the solution $\hat{\mathbf{x}}$ is incompatible with row i, then any solution $\mathbf{x}$ such that $x_\ell \leq \hat{x}_\ell$, $\forall\ \ell = 1, \ldots, L$, is also incompatible with row i. Therefore, we require that $\mathbf{x}$ contain at least one component that is larger than its corresponding component in $\hat{\mathbf{x}}$, which yields the stronger valid inequality

$$\sum_{\ell=1}^{L} \sum_{r=\hat{x}_\ell+1}^{R_\ell} y_{\ell r} \geq 1. \tag{17}$$

Constraint (17) can, in turn, be tightened further by explicitly considering the rows for which $\mathbf{x}$ is incompatible. Let $L_i = \max_{j=1,\ldots,n} b_{ij}$ be the maximum intensity in the fluence map for row i. By the same argument as above, if the current solution $\hat{\mathbf{x}}$ is incompatible with row i, then any solution $\mathbf{x}$ such that $x_\ell \leq \hat{x}_\ell$, $\forall\ \ell = 1, \ldots, L_i$, is also incompatible with row i, since no leaf positions with intensity greater than L_i can be used in decomposing row i. Therefore,

we require that $\mathbf{x}$ is larger than $\hat{\mathbf{x}}$ in at least one component $1, \ldots, L_i$:

$$\sum_{\ell=1}^{L_i} \sum_{r=\hat{x}_\ell+1}^{R_\ell} y_{\ell r} \geq 1 \quad \forall \text{ rows } i \text{ incompatible with } \hat{\mathbf{x}}. \tag{18}$$

Since (18) is stronger than (16) or (17), we use the latter inequalities in our model. Note also that (18) stated for row i_1 dominates a cut generated for row i_2 if $L_{i_1} < L_{i_2}$. Thus, we consider the bixel rows in nondecreasing order of their L_i-values, halt when an infeasible row is detected, and add a single inequality of the form (18). This sequence also tends to minimize subproblem execution time, since rows having a small maximum intensity are easier to solve by the nature of the backtracking algorithm discussed in Section 2.3.

Since the collection (18) contains an exponential number of valid inequalities, we add them only as needed in a cutting-plane fashion. In particular, this means that we will relax (18), solve the relaxation of (MP) and generate an $\mathbf{x}$-solution representing a candidate allowable intensity multiset. We then solve a subproblem for each bixel row to determine if the allowable intensity multiset is incompatible with that row. If not, we have found an optimal solution to (MP). Otherwise, we add a constraint of the form (18) to (MP) that cuts off that solution.

2.3 Subproblem Analysis and Solution Approach

In this section, we will consider the subproblem of checking whether a given intensity multiset $\mathbf{x}$ is compatible with a particular bixel row. For convenience and wherever the interpretation is clear from the context, we will suppress the index i of the bixel row and denote a typical row of the fluence map B by $\mathbf{b} = (b_1, \ldots, b_n)$.

We represent a feasible decomposition as a collection of n-dimensional binary vectors $\mathbf{v}_{\ell r}$ ($\ell = 1, \ldots, L$; $r = 1, \ldots, x_\ell$). The values of $\mathbf{v}_{\ell r}$ that equal 1 correspond to the (consecutive) exposed bixels in the r^{th} aperture having intensity ℓ. For example, the decomposition in equation (2) corresponds to $\mathbf{v}_{11} = (1, 1, 0)$, $\mathbf{v}_{31} = (0, 1, 1)$, $\mathbf{v}_{51} = (0, 0, 1)$, and $\mathbf{v}_{\ell r} = \mathbf{0}$ for other ℓ, r. (Note that this decomposition would be feasible as long as $x_1, x_3, x_5 \geq 1$.) The subproblem can then formally be presented as follows:

C1-Partition

INSTANCE: An n-dimensional vector of nonnegative integers $\mathbf{b}$ and an integer vector $\mathbf{x} = (x_1, \ldots, x_L)$.
QUESTION: Do there exist n-dimensional binary vectors $\mathbf{v}_{\ell r}$ ($\ell = 1, \ldots, L$; $r = 1, \ldots, x_\ell$) that satisfy the consecutive ones property such that $\sum_{\ell=1}^{L} \sum_{r=1}^{x_\ell} \ell \mathbf{v}_{\ell r} = \mathbf{b}$?

Proposition 1. C1-PARTITION is strongly NP-complete.

Proof. See Appendix B. □

In principle, the C1-PARTITION problem can be formulated and solved as an integer program. However, we will develop a computationally more effective backtracking algorithm that focuses on partitioning intensity requirements individually for each bixel. An integer vector $\mathbf{p}^j = (p_1^j, \ldots, p_L^j)$ provides a *bixel decomposition* of bixel $j \in \{1, \ldots, n\}$ in row $\mathbf{b}$ if and only if $b_j = \sum_{\ell=1}^{L} \ell p_\ell^j$. We then attempt to form a collection of leaf positions that realizes the individual bixel partitions. We call such a collection of leaf positions a *leaf decomposition* of $\mathbf{b}$.

In order to more effectively conduct our subproblem searches, we describe a property that will hold in some leaf decomposition (if one exists) that satisfies the given collection of bixel decompositions.

Lemma 1. Consider candidate bixel decompositions for bixels j and $j+1$, for some $j \in \{1, \ldots, n-1\}$, and suppose that these have a common decomposition intensity value ℓ, i.e., $p_\ell^j, p_\ell^{j+1} > 0$. Then, if a leaf decomposition exists, one exists in which a leaf position having intensity ℓ exposes both bixels j and $j+1$.

Proof. Assume that there exists a leaf decomposition $\mathcal{V}$ in which bixels j and $j+1$ are exposed by two separate leaf positions, $\mathbf{v}_1$ and $\mathbf{v}_2$, respectively, each having intensity ℓ. Now consider the leaf position $\mathbf{v}_3 = \mathbf{v}_1 + \mathbf{v}_2$ having intensity ℓ. Then $\mathcal{V}' = \{\mathbf{v}_3\} \cup \mathcal{V} \backslash \{\mathbf{v}_1, \mathbf{v}_2\}$ is also a leaf decomposition that realizes the given bixel decomposition. □

We next derive a necessary condition that any feasible bixel decomposition must satisfy so that the corresponding set of leaf positions is compatible with a given allowable intensity

multiset $\mathbf{x}$. Similar to the idea behind Lemma 1, if $p_\ell^j > p_\ell^{j+1}$, then $p_\ell^j - p_\ell^{j+1}$ leaf positions having intensity ℓ must expose bixel j but not $j+1$. Lemma 2 formalizes this idea.

Lemma 2. Let $\mathbf{x}$ represent an allowable intensity multiset, and $\mathbf{p}^{j_\eta}$ denote candidate bixel decompositions for bixels j_η, $\forall \eta = 1, \ldots, n'$ such that $1 \leq j_1 < \cdots < j_{n'} \leq n$. The following set of conditions must be satisfied in any feasible solution.

$$\sum_{\eta=2}^{n'} \max\{0, p_\ell^{j_{\eta-1}} - p_\ell^{j_\eta}\} + p_\ell^{j_{n'}} \leq x_\ell \quad \forall \ell = 1, \ldots, L. \tag{19}$$

Proof. If $p_\ell^{j_{\eta-1}} > p_\ell^{j_\eta}$, at least $p_\ell^{j_{\eta-1}} - p_\ell^{j_\eta}$ leaf positions having intensity ℓ must expose bixel $j_{\eta-1}$ but not j_η. Also, at least $p_\ell^{j_{n'}}$ leaf positions having intensity ℓ must expose bixel $j_{n'}$. Since all leaf positions listed above are necessarily disjoint, the lemma holds. □

We next describe our backtracking algorithm. In this algorithm, we first enumerate all possible ways of decomposing the bixel intensities in $\mathbf{b}$ using a subset of the allowable intensity multiset given by $\mathbf{x}$. We denote the set of all candidate bixel decompositions for bixel j by $\mathcal{P}_j$, where for each $\mathbf{p} \in \cup_{j=1}^n \mathcal{P}_j$, we must have $p_\ell \leq x_\ell$, $\forall\ \ell = 1, \ldots, L$.

The backtracking algorithm for solving the subproblem is stated formally in Algorithm 1. We begin by enumerating each possible element of $\mathcal{P}_j$, $\forall\ j = 1, \ldots, n$. We denote the set of processed bixels by $\mathcal{F}$ (for which a candidate "active" bixel decomposition has been established), and the set of unprocessed bixels by $\mathcal{R}$. In each iteration, we check to see if the set of candidate bixel decompositions $\mathcal{P}_j$ for any $j \in \mathcal{R}$ is empty. If so, the current active bixel decompositions do not yield a feasible solution, and the algorithm backtracks. Otherwise, we consider an unprocessed bixel $\hat{\jmath} \in \mathcal{R}$, and choose an untried bixel decomposition $\mathbf{p}^{\hat{\jmath}} \in \mathcal{P}_{\hat{\jmath}}$ to be active for bixel $\hat{\jmath}$. Next, we move $\hat{\jmath}$ from $\mathcal{R}$ to $\mathcal{F}$, creating updated sets $\mathcal{R}'$ and $\mathcal{F}'$, and invoke Lemma 2 to update the set of bixel decompositions for the bixels in $\mathcal{R}'$. Specifically, for each $j \in \mathcal{R}'$ and $\mathbf{p}^j \in \mathcal{P}_j$, we calculate the number of leaf positions that would be required due to selecting $\mathbf{p}^j$ as the active bixel decomposition for bixel j, in addition to those already selected for bixels in $\mathcal{F}'$. We eliminate $\mathbf{p}^j$ if a condition of type (19) is violated. We then recursively call the procedure to continue with a new bixel $j' \in \mathcal{R}'$.

We stop either when we find a feasible bixel decomposition for all bixels, or when we exhaust all bixel decompositions without finding a feasible solution. In the former case, a

leaf decomposition that realizes the bixel decompositions for bixels $j \in \{1, \ldots, n\}$ can be found by invoking Algorithm 3, which is based on the repeated application of Lemma 1. To see that Algorithm 3 recovers a feasible leaf decomposition, note that Algorithms 1 and 2 provide bixel decompositions that satisfy Lemma 2, and in particular, the condition

$$\sum_{j=2}^{n} \max\{0, p_\ell^{j-1} - p_\ell^j\} + p_\ell^n \leq x_\ell \quad \forall \ell = 1, \ldots, L. \tag{20}$$

Algorithm 3 recovers a feasible leaf decomposition if, in the outer while-loop corresponding to each $\ell = 1, \ldots, L$, the counter r is never incremented more than x_ℓ times. Note that r is incremented each time the inner while-loop terminates, which occurs either when $\tilde{\jmath} > n$ (a total of p_ℓ^n times), or when $p_\ell^{\tilde{\jmath}} = 0$ ($p_\ell^{\tilde{\jmath}-1} - p_\ell^{\tilde{\jmath}}$ times) for $\tilde{\jmath} = 2, \ldots, n$. The total number of times that r is incremented in the outer while-loop for $\ell = 1, \ldots, L$ is thus the left-hand-side of (20), which is no more than x_ℓ, as required.

If we exhaust all bixel decompositions without finding a feasible solution, we conclude that the current allowable intensity multiset is incompatible with the current row.

Algorithm 1 C1-PARTITION(**b**, **x**)

Input: **b** {n-dimensional vector representing bixel intensity requirements}
Input: **x** {L-dimensional vector representing an allowable intensity multiset}
{This algorithm finds whether there exists a C1-PARTITION of **b** compatible with **x**}
$\mathcal{F} \leftarrow \emptyset$ {$\mathcal{F}$ is the set of processed bixels}
$\mathcal{R} \leftarrow \{1, \ldots, n\}$ {$\mathcal{R}$ is the set of unprocessed bixels}
for all $j \in \{1, \ldots, n\}$ **do**
  $\mathcal{P}_j \leftarrow$ Enumerate all bixel decompositions compatible with **x** for bixel j
$\mathcal{P} \leftarrow \{\mathcal{P}_1, \ldots, \mathcal{P}_n\}$
return C1-PARTITIONRECURSIVE(**b**, **x**, $\mathcal{F}$, $\mathcal{R}$, $\mathcal{P}$)

Since Algorithm 1 is a backtracking algorithm, and therefore in the worst case investigates all possible bixel decompositions, it is of exponential time complexity (as expected, due to Proposition 1). However, the empirical running time of the algorithm can be reduced using the following observations:

(i) If two adjacent bixels in a row have the same required intensity value, there must exist an optimal solution in which they are exposed by the same leaf positions. This result

Algorithm 2 C1-PartitionRecursive($\mathbf{b}$, $\mathbf{x}$, $\mathcal{F}$, $\mathcal{R}$, $\mathcal{P}$)

if $\mathcal{R} = \emptyset$ **then**
  return true {all bixels have been processed, $\mathcal{P}$ represents a feasible solution}
else
  if $\exists j \in \mathcal{R} : \mathcal{P}_j = \emptyset$ **then**
    return false {there is no remaining way of decomposing bixel j}
else
  $\hat{\jmath} \leftarrow \text{argmin}_{j \in \mathcal{R}} |\mathcal{P}_j|$ {$\hat{\jmath}$ is a bixel having the smallest number of bixel decompositions}
  for all $\mathbf{p}^{\hat{\jmath}} \in \mathcal{P}_{\hat{\jmath}}$ **do**
    $\mathcal{P}' \leftarrow \mathcal{P}$, $\mathcal{P}'_{\hat{\jmath}} \leftarrow \{\mathbf{p}^{\hat{\jmath}}\}$ {$\mathbf{p}^{\hat{\jmath}}$ is now the active decomposition for bixel $\hat{\jmath}$}
    $\mathcal{F}' \leftarrow \mathcal{F} \cup \{\hat{\jmath}\}$, $\mathcal{R}' \leftarrow \mathcal{R} \setminus \{\hat{\jmath}\}$
    for all $j \in \mathcal{R}'$ **do**
      $\mathcal{P}'_j \leftarrow \mathcal{P}_j \setminus$ {all elements eliminated by Lemma 2, given the active decompositions $\mathbf{p}^{\tilde{\jmath}}$ for $\tilde{\jmath} \in \mathcal{F}'$}
    if C1-PartitionRecursive($\mathbf{b}$, $\mathbf{x}$, $\mathcal{F}'$, $\mathcal{R}'$, $\mathcal{P}'$) **then**
      return true {a feasible solution that uses $\mathbf{p}^{\hat{\jmath}}$ to decompose bixel $\hat{\jmath}$ is found}
  return false {all bixel decompositions of bixel $\hat{\jmath}$ have been exhausted}

Algorithm 3 RecoverLeafDecomposition($\mathbf{b}$, $\mathbf{x}$, $\mathcal{P}$)

Require: $\mathcal{P}_j = \{\mathbf{p}^j\}$ $\forall j \in \{1, \ldots, n\}$ {all bixels have been processed}
Output: $\mathbf{v}_{\ell r}$ ($\ell = 1, \ldots, L$; $r = 1, \ldots, x_\ell$)
{$\mathbf{v}_{\ell r}$ is an n-dimensional binary vector that represents a leaf position}
for all $\ell \in \{1, \ldots, L\}$, $r \in \{1, \ldots, x_\ell\}$ **do**
  $\mathbf{v}_{\ell r} \leftarrow \mathbf{0}$
for all $\ell \in \{1, \ldots, L\}$ **do**
  $r \leftarrow 1$, $j \leftarrow 1$
  while $j \leq n$ **do**
    if $p_\ell^j > 0$ **then**
      $\tilde{\jmath} \leftarrow j$ {a new leaf position must start at bixel j}
      while $\tilde{\jmath} \leq n$ and $p_\ell^{\tilde{\jmath}} > 0$ **do**
        {expand the new leaf position as much as possible}
        $v_{\ell r \tilde{\jmath}} \leftarrow 1$
        $p_\ell^{\tilde{\jmath}} \leftarrow p_\ell^{\tilde{\jmath}} - 1$, $\tilde{\jmath} \leftarrow \tilde{\jmath} + 1$
      $r \leftarrow r + 1$
    else
      $j \leftarrow j + 1$ {all leaf positions that start at bixel j have been recovered}

can be proven in a similar way as Lemma 1, and is therefore omitted for brevity. This observation implies that we can preprocess the data by merging all adjacent bixels in a bixel row having the same intensity requirement, thereby reducing the dimensionality of the problem instance.

(ii) In choosing the next bixel to be processed, we pick a bixel $j \in \mathcal{R}$ having the smallest number of remaining candidate bixel decompositions. In this manner, we can quickly enumerate all possible bixel decompositions for a few key bixels, and eliminate a significant portion of bixel decompositions for the remaining bixels without wasting effort by unnecessary backtracking steps.

(iii) In choosing the next candidate bixel decomposition $\mathbf{p}^j \in \mathcal{P}_j$ for a chosen bixel $j \in \mathcal{R}$, we select an untried bixel decomposition having the fewest number of intensity values. Since each intensity value used in decomposing a bixel needs to be assigned to a different aperture, this rule favors a bixel decomposition using the fewest number of apertures to decompose the chosen bixel. Therefore, it tends to retain the availability of more elements of the allowable intensity multiset (and hence apertures) for the remaining bixels, making it easier to find a feasible solution (if one exists).

2.4 Valid Inequalities for the Master Problem

The initial optimal solution to the relaxation of (MP) in which none of the inequalities (18) have yet been added to the model will set all variables equal to zero, which is clearly incompatible with all rows. In this section, we derive some characteristics of all feasible solutions and use these to define valid inequalities for (MP). In this way, we attempt to improve the convergence rate of the decomposition algorithm by eliminating some clearly infeasible solutions before the initial execution of the master problem.

2.4.1 Inequalities Based on Beam-on-time and Number of Apertures

Our first observation uses and generalizes the fact that the beam-on-time, number of apertures, and total treatment time required for the decomposition of any single row into leaf

positions provide lower bounds on the minimum beam-on-time, number of apertures, and total treatment time, respectively, needed to deliver the entire fluence map. More generally, consider any collection of nonnegative objective weights w_1' and w_2' in place of w_1 and w_2, and let $T_i(w_1', w_2')$ be the minimum value of the objective with respect to these weights over all decompositions for row i only. Then the following are valid inequalities for (MP):

$$w_1' \sum_{\ell=1}^{L} x_\ell + w_2' \sum_{\ell=1}^{L} \ell x_\ell \geq T_i(w_1', w_2') \quad \forall\ i = 1, \ldots, m. \tag{21}$$

We formulate an integer programming model to determine $T_i(w_1', w_2')$ for a given row i. First, denote the set of possible leaf positions for that row by $\mathcal{K}$, and define n-dimensional binary vectors $\mathbf{v}_k$ for $k \in \mathcal{K}$ (where $|\mathcal{K}| = O(n^2)$), such that $v_{kj} = 1$ if and only if bixel j is exposed by leaf position k. In addition to decision variables x_ℓ as in (MP), define binary decision variables $z_{k\ell}$, $\forall\ k \in \mathcal{K}$, $\ell = 1, \ldots, L^k$ such that $z_{k\ell} = 1$ if and only if leaf position k is used with intensity ℓ (where $L^k = \min_{j:v_{kj}=1} b_j$ is an upper bound on the intensity of leaf position k.) Then $T_i(w_1', w_2')$ is the optimal objective function value of the following optimization problem, (SR):

$$\text{minimize } w_1' \sum_{\ell=1}^{L} x_\ell + w_2' \sum_{\ell=1}^{L} \ell x_\ell \tag{22}$$

subject to

$$\sum_{k \in \mathcal{K}} \left(v_{jk} \sum_{\ell=1}^{L^k} \ell z_{k\ell} \right) = b_j \qquad \forall\ j = 1, \ldots, n \tag{23}$$

$$\sum_{\ell=1}^{L^k} z_{k\ell} \leq 1 \qquad \forall\ k \in \mathcal{K} \tag{24}$$

$$\sum_{k \in \mathcal{K}: L^k \geq \ell} z_{k\ell} = x_\ell \qquad \forall\ \ell = 1, \ldots, L \tag{25}$$

$$z_{k\ell} \in \{0, 1\} \qquad \forall\ k \in \mathcal{K},\ \ell = 1, \ldots, L^k \tag{26}$$

$$x_\ell \geq 0 \text{ and integer} \qquad \forall\ \ell = 1, \ldots, L. \tag{27}$$

Constraints (23) ensure that each bixel receives exactly its required amount of dose while constraints (24) guarantee that each leaf position is either not used or is assigned to a single intensity value. Finally, constraints (25) relate the x- and z-variables.

A practical difficulty in implementing the valid inequalities of the form (21) is that we must determine appropriate values for the weights w'_1 and w'_2. However, Baatar [3] shows that, when decomposing a single bixel row, there exists a set of leaf positions that simultaneously minimizes both beam-on-time and the number of apertures. If we let $N_i = T_i(1,0)$ represent the minimum number of apertures for row i, and $\tilde{z}_i = T_i(0,1)$ represent the minimum beam-on-time for row i, this implies that $T_i(w'_1, w'_2) = w'_1 N_i + w'_2 \tilde{z}_i$, so that we can replace (21) by

$$w'_1 \sum_{\ell=1}^{L} x_\ell + w'_2 \sum_{\ell=1}^{L} \ell x_\ell \geq w'_1 N_i + w'_2 \tilde{z}_i \quad \forall\, i = 1, \ldots, m. \tag{28}$$

It is easy to see that we can capture *all* of these valid inequalities by restricting ourselves to the coefficient pairs $(w'_1, w'_2) = (1,0)$ and $(0,1)$ only:

$$\sum_{\ell=1}^{L} x_\ell \;\geq\; \max_{i=1,\ldots,m} \{N_i\} \tag{29}$$

$$\sum_{\ell=1}^{L} \ell x_\ell \;\geq\; \max_{i=1,\ldots,m} \{\tilde{z}_i\}. \tag{30}$$

We can generalize this idea as follows. Let $R(\mathcal{L})$ denote the set of rows for which the maximum intensity requirement is bounded by $\mathcal{L}$ for some $\mathcal{L} \in \{1, \ldots, L\}$, i.e., $R(\mathcal{L}) = \{i \in \{1, \ldots, m\} : L_i \leq \mathcal{L}\}$. Since intensity values greater than $\mathcal{L}$ cannot be used in decomposing the rows in $R(\mathcal{L})$, a similar approach to the one above can be used to derive the following family of valid inequalities

$$\sum_{\ell=1}^{\mathcal{L}} x_\ell \;\geq\; \max_{i \in R(\mathcal{L})} \{N_i\} \quad \forall\, \mathcal{L} = 1, \ldots, L \tag{31}$$

$$\sum_{\ell=1}^{\mathcal{L}} \ell x_\ell \;\geq\; \max_{i \in R(\mathcal{L})} \{\tilde{z}_i\} \quad \forall\, \mathcal{L} = 1, \ldots, L. \tag{32}$$

Finally, note that the values of N_i and $\tilde{z}_i$ can be found by solving (SR) with $w'_1 = 1, w'_2 = 1$ or by using the method of Kalinowski [15], since there exists a solution that minimizes both beam-on-time and the number of apertures [3].

2.4.2 Inequalities Based on Bixel Subsequences

Recall that (16)–(18) represent necessary conditions for feasibility of an allowable intensity multiset with respect to a particular row. It is possible to develop stronger necessary conditions if we examine subsequences of a row, i.e., a subset of the required intensity values in a row that preserves their order in the fluence map. First, Lemma 3 shows that, if a given allowable intensity multiset is incompatible with a subsequence $\mathbf{s}$ of row i, then it also must be incompatible with row i.

Lemma 3. Consider an allowable intensity multiset $\mathbf{x}$, an n-dimensional vector $\mathbf{b}$ that represents the intensity requirements of the bixels in some row of B, and an n'-dimensional vector $\mathbf{s} = (b_{j_1}, \ldots, b_{j_{n'}})$, where $1 \leq j_1 < \cdots < j_{n'} \leq n$. If $\mathbf{x}$ is not compatible with $\mathbf{s}$, then it is also not compatible with $\mathbf{b}$.

Proof. We prove the equivalent statement that if $\mathbf{x}$ is compatible with $\mathbf{b}$, then it is also compatible with $\mathbf{s}$. Assume that $\mathbf{x}$ is compatible with $\mathbf{b}$. By definition, there exists a bixel decomposition for each bixel $j = 1, \ldots, n$ so that the resulting set of leaf positions is compatible with $\mathbf{x}$. The bixel decompositions corresponding to only the bixels in $\mathbf{s}$ are also compatible with $\mathbf{x}$, since the order of the bixels in $\mathbf{s}$ is the same as that in $\mathbf{b}$. □

Note that we can invoke Lemma 3 to associate a subproblem with each of the $O(2^n)$ subsequences of a bixel row $\mathbf{b}$. Each of these subproblems can then be used to generate cutting planes of the form (18), as well as valid inequalities of the form (31) and (32). However, since the strength of (18), (31) and (32) depend on the largest intensity value in a bixel row, we form subsequences of each bixel row by, for $\mathcal{L} = 1, \ldots, L$, considering only those bixels having required intensity less than or equal to $\mathcal{L}$. The valid inequalities generated by the $O(\min(n, L))$ subsequences generated in this fashion imply *all* $O(2^n)$ valid inequalities associated with all possible subsequences.

2.5 Constructing a Feasible Matrix Decomposition

Our algorithm finds an optimal allowable intensity multiset and a bixel decomposition for each bixel row. In order to construct a corresponding matrix decomposition, we need to

apply Algorithm 3 to find a leaf decomposition for each row. We can then generate aperture matrices by arbitrarily combining leaf positions using the same intensity values in different rows. We have found empirically that this simple approach yields a feasible matrix decomposition very quickly.

Since any pair of leaf positions assigned to the same intensity value in different rows can be combined, there are up to $\left(\prod_{\ell=1}^{L}(x_\ell!)\right)^m$ aperture matrices that can be constructed from a given feasible leaf decomposition for each row. Even though each such choice represents an alternative optimal solution to the optimal leaf sequencing problem, some matrix decompositions may clinically be preferable to others based on their structural properties. Perhaps the most challenging structural consideration pertains to the so-called "tongue-and-groove" effect observed in MLCs. We refer the reader to the works of Deng et al. [9] and Que et al. [31] for technical details of the tongue-and-groove effect in dynamic MLC dose delivery. For the purposes of this study, it is sufficient to understand that leaves in adjacent rows often interlock with a tongue on the bottom of one row sliding along a groove in the top of another row. Tongue-and-groove underdosage occurs since a leaf's tongue blocks dosage intended for cells beneath it. Therefore, it is desirable to limit such underdosages.

In order to measure the amount of tongue-and-groove effect in a treatment plan, Que et al. [31] note that it is generally not desirable to deliver one aperture in which some bixel (i, j) is blocked by a leaf while bixel $(i+1, j)$ is not blocked, if another aperture is being delivered where (i, j) is not blocked by a leaf while $(i + 1, j)$ is blocked. Based on this observation, Que et al. [31] derive the following *tongue-and-groove index* (TGI). Suppose a treatment plan consists of K apertures described by binary values v_j^{ik}, where $v_j^{ik} = 0$ if cell (i, j) is blocked by a leaf in aperture k and $v_j^{ik} = 1$ otherwise, for each $i = 1, \ldots, m$, $j = 1, \ldots, n$, $k = 1, \ldots, K$. Let I_k be the intensity delivered in aperture $k = 1, \ldots, K$. Then the TGI of a matrix decomposition is defined as:

$$\sum_{i=1}^{m-1}\sum_{j=1}^{n}\sum_{k=1}^{K-1}\sum_{\ell=k+1}^{K}\min\{I_k, I_\ell\}\left[v_j^{ik}(1-v_j^{i+1,k})(1-v_j^{i\ell})v_j^{i+1,\ell} + (1-v_j^{ik})v_j^{i+1,k}v_j^{i\ell}(1-v_j^{i+1,\ell})\right]. \quad (33)$$

We thus can calculate the TGI component induced by rows 1 and 2 (of all aperture pairs),

then rows 2 and 3, and so on, down to rows $m-1$ and m. This observation allows us to focus on pairs of rows instead of pairs of entire aperture matrices while reducing TGI, allowing us to design an efficient algorithm for TGI reduction given a set of bixel decompositions for each row.

Given a pair of adjacent rows, we attempt to match individual leaf positions in the two rows to minimize the TGI induced by the adjacent row pair. In order to limit computational overhead in this phase of our algorithm, we reduce TGI indirectly by the following scheme. Let us denote a leaf position for row i by a binary n-vector $\mathbf{v}^i$, where $v^i_j = 1$ if the leaf position exposes bixel j in row i. We measure the *overlap* between two leaf positions having the same intensity value in consecutive rows by counting the number of columns that both leaf positions expose simultaneously. Formally, we define the overlap between leaf positions $\mathbf{v}^i$ and $\mathbf{v}^{i+1}$ as $\theta(\mathbf{v}^i, \mathbf{v}^{i+1}) = \sum_{j=1}^{n} v^i_j v^{i+1}_j$. Our approach is to heuristically minimize TGI by maximizing the total overlap between all leaf position pairs, which can efficiently be solved as an assignment problem. The efficiency of the assignment problems can be further improved by noting that the problem decomposes over the intensity values $\ell \in \{1, \ldots, L\}$, since only leaf positions having the same intensity value can be combined. Therefore, we can generate a matrix decomposition by finding a leaf decomposition for each row, and then matching leaf positions in adjacent rows having the same intensity value by solving an assignment problem so that the total overlap is maximized.

The TGI minimization step described in the previous paragraph can be improved as follows. Typically, multiple bixel decompositions exist for each row that are compatible with a given feasible intensity multiset. Algorithm 2 can be modified in a straightforward manner so that it finds *all* leaf decompositions of a row, instead of stopping once the first feasible bixel decomposition for all bixels is found. Since different bixel decompositions for a bixel row correspond to different leaf decompositions, considering alternative bixel decompositions can lead to a matrix decomposition having a smaller TGI.

Given alternative leaf decompositions for each row, the problem of minimizing TGI can be formulated as a shortest path problem as follows. We create a layered network in which each layer corresponds to a bixel row $i \in \{1, \ldots, m\}$, and node N_{id} represents the d^{th} leaf decomposition of row i. We add a directed arc from each node N_{id} to all nodes $N_{(i+1)d'}$, for

all $i = 1, \ldots, m-1$. The cost of the arc from node N_{id} to $N_{(i+1)d'}$ is given by the TGI value resulting from the assignment solution corresponding to the candidate leaf decompositions represented by d for row i, and d' for row $i+1$. Finally, we add a start node S and a finish node F. We create zero-cost arcs from S to all nodes in the first layer, and from all nodes in the last layer to F. A shortest S–F path in this graph represents a matrix decomposition having a minimum TGI from among the provided options. Since the graph is acyclic, the shortest path problem can be solved in $O(|\mathcal{A}|)$ time, where $\mathcal{A}$ is the set of all arcs.

Remark 1. The shortest path approach to minimizing TGI can be difficult to solve quickly when bixel rows have a large number of alternative leaf decompositions, since an arc joins each pair of nodes corresponding to adjacent bixel rows. In order to partially overcome this difficulty, we limit the number of bixel decompositions found by Algorithm 2 by terminating once 250 feasible bixel decompositions have been identified. Next, note that a straightforward acyclic shortest path implementation processes layers one at a time, and does not generate a feasible S–F path before processing the last layer. Since being able to specify a time limit is a desired feature in a practical setting, we use a hybrid algorithm for solving the shortest path problem. Our algorithm starts by processing layers one-by-one, updating node labels as usual. If a shortest path is not found when a given initial time limit expires, our algorithm switches to a depth-first-search (DFS) procedure, which we terminate after a given final time limit. We start DFS from an unprocessed node N_{id} having a smallest label, select a minimum-cost arc $(N_{id}, N_{(i+1)d'})$ exiting that node, and update the label of $N_{(i+1)d'}$ if we have found a new shortest S–$N_{(i+1)d'}$ path. Else, the algorithm backtracks and seeks another arc from N_{id}. We then return the shortest S–F path found by this procedure when the final time limit is reached.

3 Computational Results and Comparisons

3.1 Problem Instances

In our experiments we have used two classes of problem instances. Our base set of test problem instances consists of 25 clinical problem instances that were obtained from treatment

plans for five head-and-neck cancer patients treated using five beam angles each. Table 1 reports the problem characteristics for these problem instances in terms of the matrix dimensions m and n. The maximum intensity value is $L = 20$ for all these instances. In addition, to allow comparison of our results with published results on other approaches to the problem, we generated 100 random problem instances of dimensions 20×20 having maximum intensity value $L = 10$.

However, since these problem instances are generally too large to be solvable by the integer programming model from Langer et al. [24] and its modification described in Appendix A, we also randomly generated eight instances ("test5x5a", ..., "test6x7b") to demonstrate the computational limitations of the latter approaches.

Unless otherwise specified, we used $w_1 = 7$ and $w_2 = 1$ as the objective weights for the number of apertures and beam-on-time, respectively.

Name	**m**	**n**	**Name**	**m**	**n**	**Name**	**m**	**n**	**Name**	**m**	**n**	**Name**	**m**	**n**
c1b1	15	14	c2b1	18	20	c3b1	22	17	c4b1	19	22	c5b1	15	16
c1b2	11	15	c2b2	17	19	c3b2	15	19	c4b2	13	24	c5b2	13	17
c1b3	15	15	c2b3	18	18	c3b3	20	17	c4b3	18	23	c5b3	14	16
c1b4	15	15	c2b4	18	18	c3b4	19	17	c4b4	17	23	c5b4	14	16
c1b5	11	15	c2b5	17	18	c3b5	15	19	c4b5	12	24	c5b5	12	17

Table 1: Dimensions of Clinical Problem Instances

3.2 Implementation Issues

We have implemented our decomposition algorithm using CPLEX 11.0 running on a Windows XP PC with a 3.4 GHz CPU and 2 GB RAM. We use callback functions of CPLEX to generate a single branch-and-bound tree in which we solve the subproblems corresponding to each integer solution found in the tree, and add cuts to tighten the master problem as necessary. This implementation turned out to be consistently faster than one which re-solves the master problem each time a cutting plane is added to the model. Furthermore, in our base algorithm, we use the subsequence inequalities (31) and (32) described in Section 2.4.2. We also use Engel's heuristic [11], which executes in well under one CPU second for each instance and generates a solution having minimum beam-on-time, to (i) obtain an initial

upper bound and (ii) compute the upper bounds R_ℓ ($\ell = 1, \ldots, L$).

3.3 Comparison with Langer et al.'s Model

Our first experiment compares our base algorithm that minimizes the total treatment time to that of Langer et al. [24] and to the modification of their model as described in Appendix A of this paper. We choose randomly generated test instances of various dimensions in order to identify the problem sizes that can be solved by each algorithm, as well as four of the smallest clinical instances to compare the effectiveness of the algorithms on clinical instances. We imposed a one-hour time limit past which we halted the execution of an algorithm. For these experiments we disabled the use of Engel's heuristic as an initial heuristic to test the ability of these models to efficiently find good-quality upper bounds.

Table 2 summarizes the results of these three algorithms in terms of the execution time, the best upper and lower bounds found within the time limit, and the optimality gap (calculated as the difference between the upper and lower bound as a percentage of the upper bound). Our decomposition algorithm can solve all 15 instances in this data set within a few seconds, whereas only six instances can be solved to optimality within an hour by either integer programming formulation. We conclude that, even though the integer programming formulation given in [24] can solve small instances to optimality, it cannot be used to solve clinical problem instances to optimality within practical computation time limits.

3.4 Random Problem Instances

For our next experiment, we first solved each of the 20×20 random problem instances in our data set to optimality for the problems of (i) minimizing total treatment time ("Total Time"), (ii) minimizing the number of apertures while constraining the beam-on-time to be minimal ("Lexicographic"), and (iii) minimizing the number of apertures ("# Apertures"). We also implemented three heuristic algorithms proposed by Siochi [38], Engel [11], and Xia and Verhey [41], which we executed on the same data set. (The results we present from Siochi [38] refer to the Variable Depth Recursion (VDR) algorithm without tongue-and-groove constraints, using the parameters recommended in the paper. We will discuss the

				Two Stage		Langer				Modified Langer			
Name	**m**	**n**	**L**	**CPU**	**Optimal**	**CPU**	**UB**	**LB**	**GAP**	**CPU**	**UB**	**LB**	**GAP**
test3x3	3	3	8	0.1	29	1.6	29	29.00	0.0%	0.9	29	29.00	0.0%
test3x4	3	4	8	0.1	37	5.2	37	37.00	0.0%	1.6	37	37.00	0.0%
test4x4	4	4	8	0.1	36	30.4	36	36.00	0.0%	10.7	36	36.00	0.0%
test5x5a	5	5	10	0.2	45	2069.6	45	45.00	0.0%	86.4	45	45.00	0.0%
test5x5b	5	5	15	0.2	50	198.2	50	50.00	0.0%	92.6	50	50.00	0.0%
test5x6a	5	6	10	0.2	55	3600	61	33.53	45.0%	3600	55	40.95	25.5%
test5x6b	5	6	18	0.4	71	3600	84	51.58	38.6%	3600	77	58.67	23.8%
test6x6a	6	6	13	0.3	55	3600	55	45.63	17.0%	3600	55	48.00	12.7%
test6x6b	6	6	13	0.3	52	3600	57	43.82	23.1%	3600	57	50.00	12.3%
test6x7a	7	6	10	0.2	45	690.0	45	45.00	0.0%	435.1	45	45.00	0.0%
test6x7b	6	7	15	0.4	74	3600	94	35.69	62.0%	3600	80	47.88	40.1%
c1b1	15	14	20	1.3	111	3600	336	48.58	85.5%	3600	273	42.00	84.6%
c1b2	11	15	20	0.8	104	3600	280	38.26	86.3%	3600	132	39.55	70.0%
c1b5	11	15	20	3.1	104	3600	280	46.20	83.5%	3600	140	49.29	64.8%
c5b4	14	16	20	2.5	124	3600	360	34.00	90.6%	3600	360	39.11	89.1%

Table 2: Comparison of Our Base Algorithm with Langer et al.'s Model

effect of including tongue-and-groove considerations in the algorithm below.)

Figure 2 summarizes the total treatment times associated with the solutions generated by the six algorithms we tested. Each algorithm is represented by a curve that depicts quality of the solutions obtained by the corresponding algorithm. For each value T of total treatment time on the horizontal axis, each curve plots the number of problem instances for which the corresponding algorithm was able to find a solution having total treatment time no more than T. For instance, Figure 2 shows that Siochi's heuristic found a solution with a total treatment time of at most 175 time units in 5% of the problem instances, while an optimal solution (represented by "Total Time") has the same quality level in 97% of the problem instances. We observe that all three exact algorithms find solutions having similar treatment times. Solution qualities generated by the Engel and Siochi heuristics are similar, with the Siochi heuristic being slightly better. A comparison of the heuristic solutions with optimal solutions reveals that average optimality gaps for Siochi, Engel, and Xia-Verhey heuristics are 10.1%, 12.0%, and 51.5%, respectively.

Figure 3 compares the algorithms with respect to the number of apertures used in their respective solutions. We note that our algorithm that minimizes total treatment time ("Total

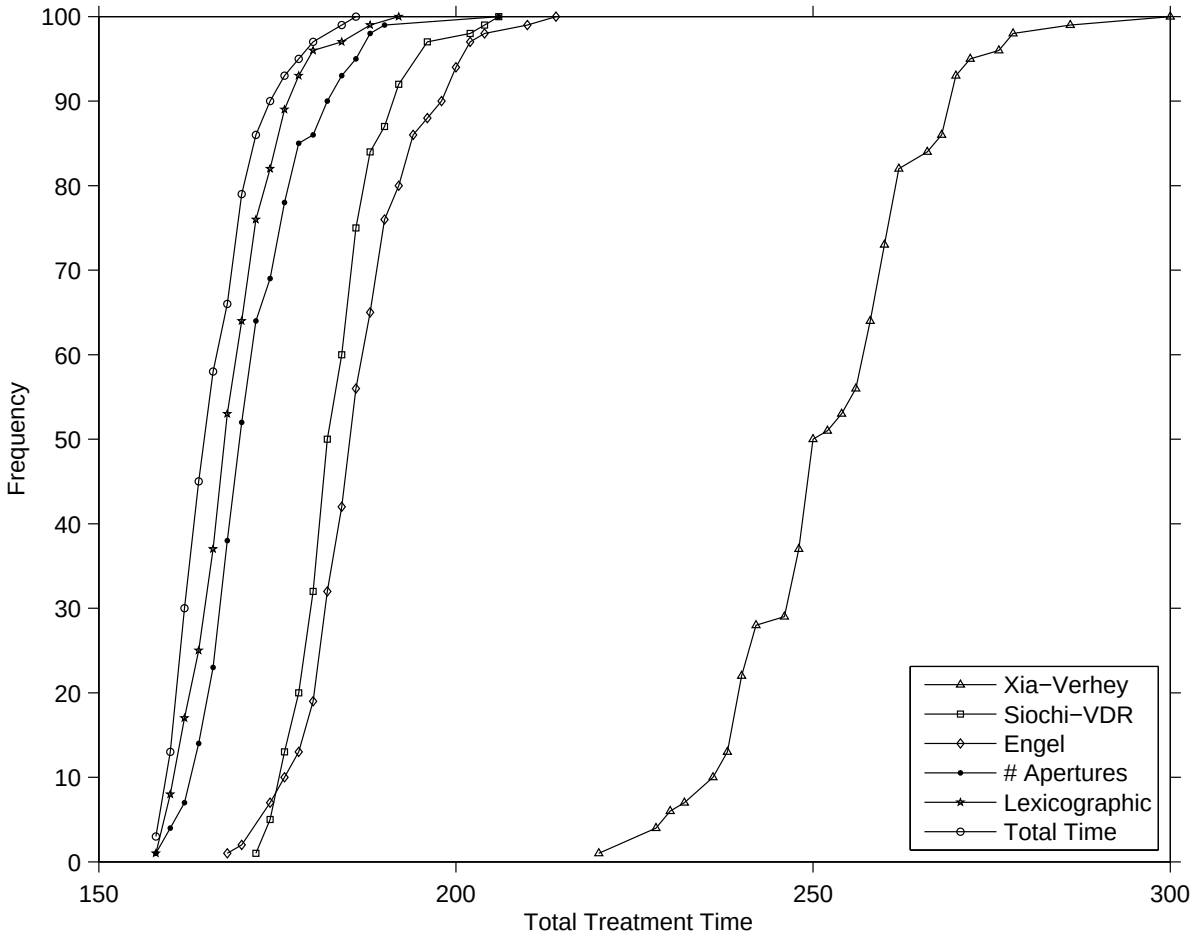


Figure 2: Comparison of Total Treatment Times on Random Data

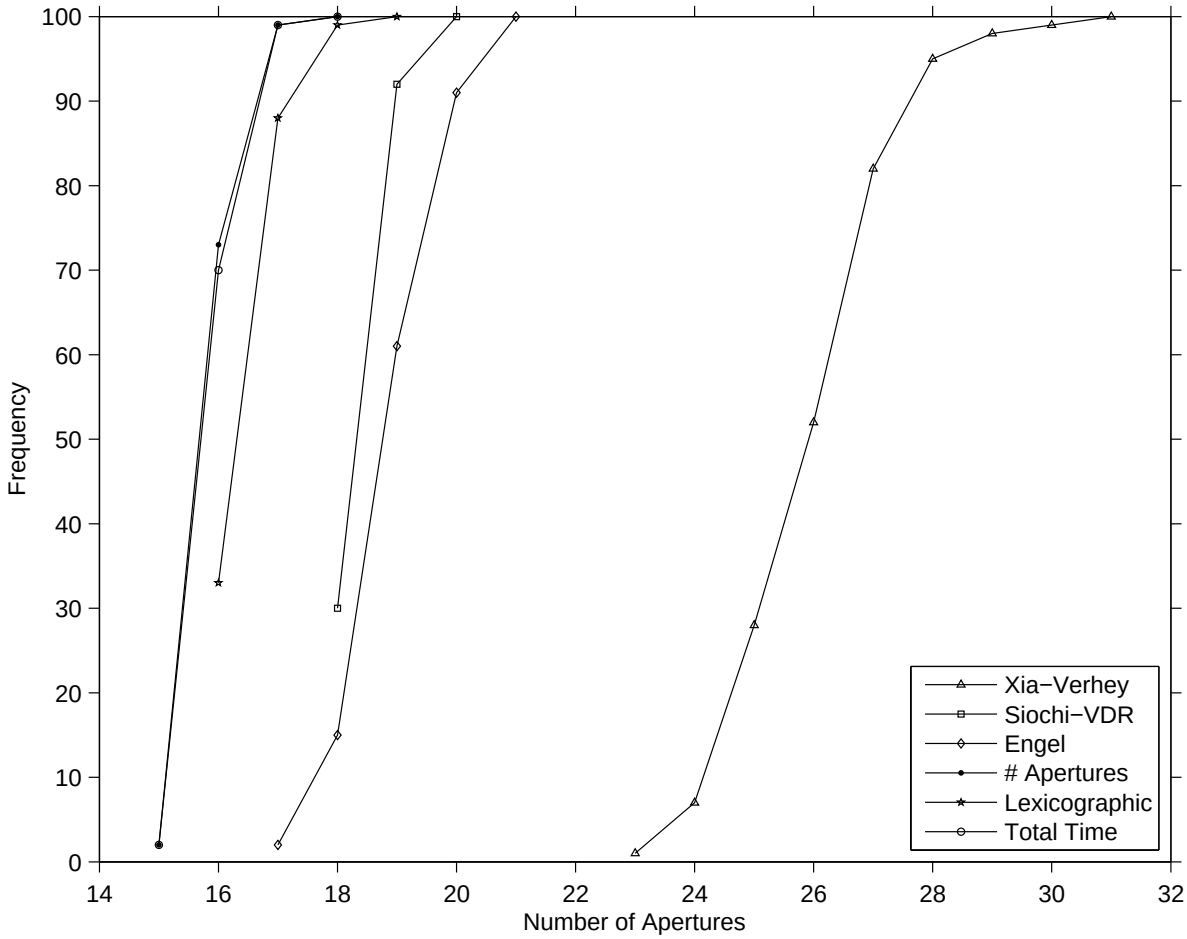


Figure 3: Comparison of Number of Apertures on Random Data

Time") finds a solution that also minimizes the number of apertures for most problem instances. As expected, lexicographic minimization of the two objective functions results in an increased number of apertures. For this objective the "# Apertures" algorithm finds optimal solutions. Average optimality gaps for the heuristics of Siochi, Engel, and Xia-Verhey are 15.6%, 18.9%, and 62.3%, respectively.

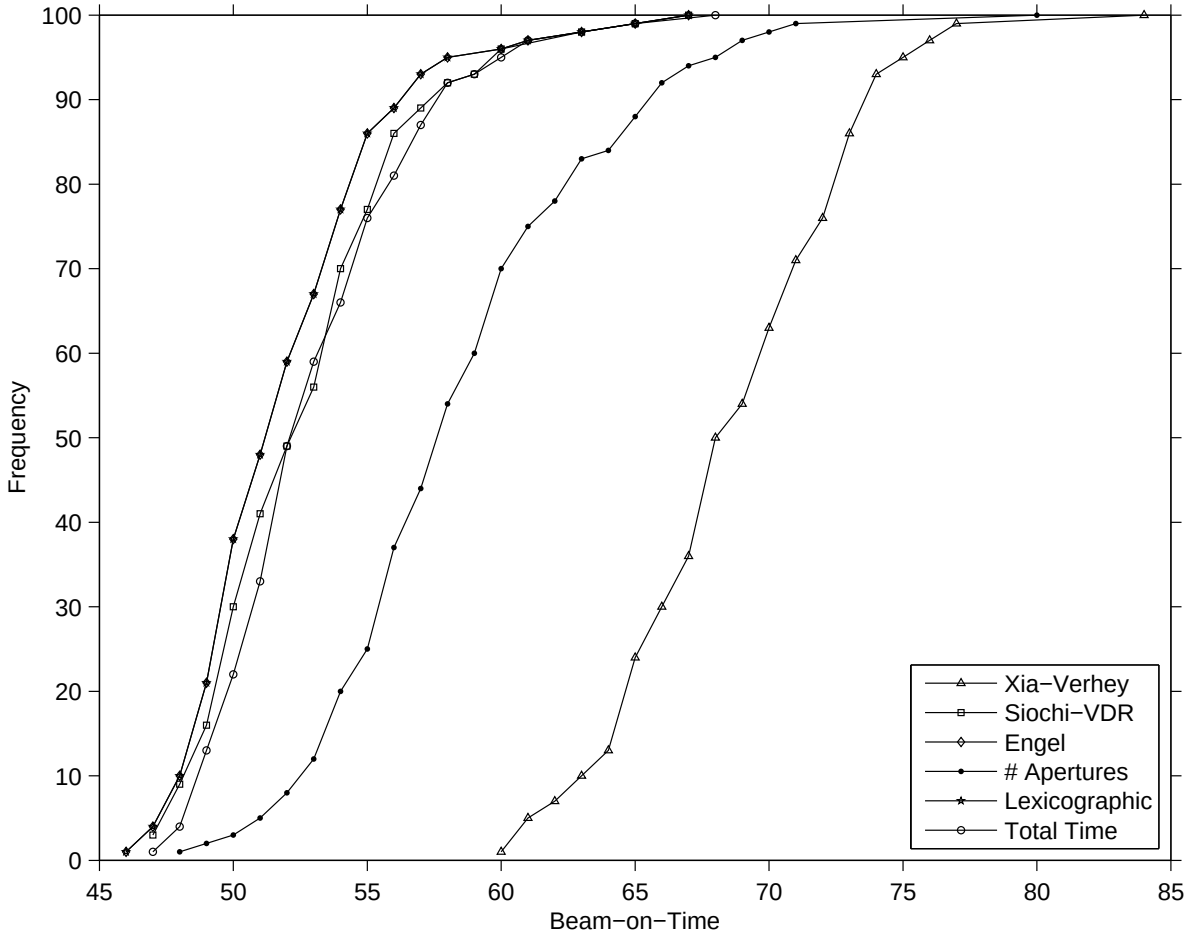


Figure 4: Comparison of Beam-on-Time Values on Random Data

We analyze the beam-on-time values of the solutions generated by each algorithm in Figure 4. Since both Engel's heuristic and our "Lexicographic" algorithm find optimal solutions having minimum beam-on-time, their curves overlap. We observe that the Siochi heuristic and our "Total Time" algorithm tend to generate solutions having small beam-on-time values, but the solutions generated by our "# Apertures" algorithm, and by the

Xia-Verhey heuristic have higher beam-on-time values. We calculated the average optimality gaps for the latter two algorithms as 12.6% and 32.1%, respectively.

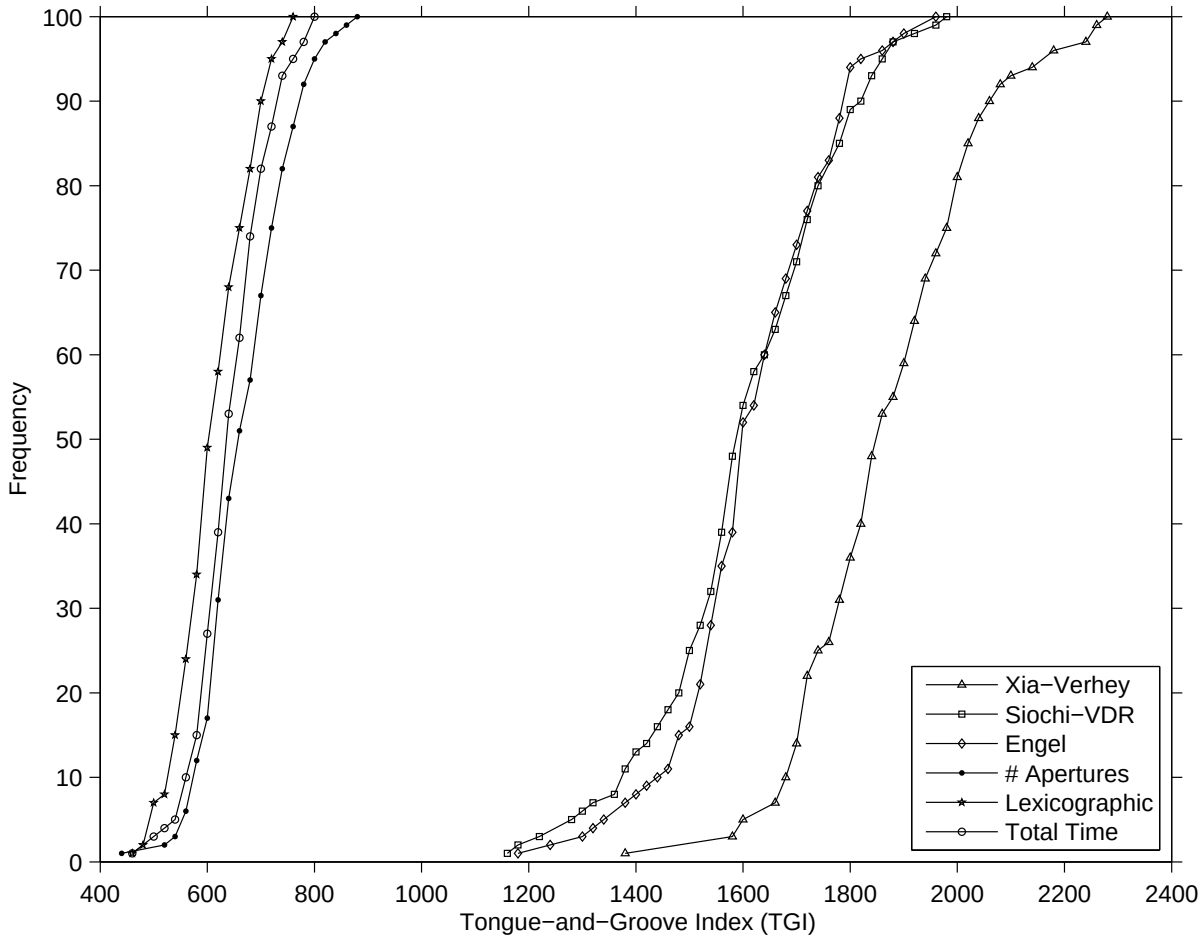


Figure 5: Comparison of TGI Values on Random Data

The final measure of solution quality that we consider is TGI, which is a measure of the tongue-and-groove effect given by (33). Figure 5 reveals that the solutions obtained by all three variants of our decomposition algorithm have significantly lower TGI values than the heuristic procedures. This result implies that, even though our TGI-reduction algorithm described in Section 2.5 does not guarantee a minimum TGI, it is highly effective in finding solutions with TGI values superior to the other heuristic approaches. In order to estimate optimality gaps for the heuristics we compare heuristic solutions with the solutions generated by our "Lexicographic" algorithm, which provides the best TGI among all methods

mentioned above. We note that average gaps for Siochi, Engel and Xia-Verhey heuristics are 162.1%, 164.4%, and 205.4%, respectively. We also note that these heuristics do not attempt to minimize TGI, and it might be possible to modify them to obtain solutions with lower TGI values. It is interesting to note that a variant of Siochi's algorithm [38] is capable of completely eliminating TGI at the expense of creating additional apertures. This variant is reported to increase the number of apertures by 10% to 30% relative to the variant that does not remove TGI [38].

Finally, the Engel and Xia-Verhey heuristics took less than one second of CPU time in all instances we tested. The average CPU time for Siochi's heuristic, "Total Time" algorithm, "# Apertures" algorithm, and "Lexicographic" algorithm were 31.5, 963.1, 414.8, and 421.4 seconds, respectively. We note that all variants of our two-stage algorithm showed a "heavy-tail" behavior, where about 80% of the problem instances were solved to optimality in less than the average solution time. For instance, using the "# Apertures" algorithm, we were able to solve 40 instances within one minute, 58 within two minutes, 81 within 414.8 seconds (the average solution time for this algorithm), 90 within 15 minutes, and all but three instances were solved within an hour. The remaining three instances were solved within three hours.

3.5 Clinical Problem Instances

Recall from Section 1 that in clinical practice, we can deliver each fluence map using a decomposition into either row-convex or column-convex apertures, where the latter requires rotation of the MLC head. Our final set of experiments compares the algorithms on clinical problem instances in our data set, allowing for MLC head rotation.

We first show the results of applying our decomposition algorithm to decompose each of the 25 clinical fluence maps into row-convex apertures, and column-convex apertures, where the latter is achieved by applying our algorithm to the transpose of each fluence map. Table 3 reports the performance of our algorithm when the objective function is set to minimize total treatment time, and displays the number of apertures ("nAper"), beam-on-time ("BOT"), total treatment time ("Time"), tongue-and-groove index ("TGI"), and CPU

time used ("CPU") for the algorithm.

Our algorithm finds an optimal solution to several instances within a few seconds while four instances take more than 10 minutes of CPU time to be solved to optimality. Comparing the solutions obtained for row-convex and column-convex decompositions, we observe that rotating the MLC head is most beneficial (in terms of treatment time) for instances in which the number of rows is much smaller than the number of columns. These benefits are most apparent on instances c4b2 and c4b5, where rotating the MLC head can result in more than 50% reduction in total treatment time. We also note that several problem instances require much less computational time to solve for a column-convex decomposition compared to a row-convex decomposition.

	Row-Convex					**Column-Convex**				
Name	**nAper**	**BOT**	**Time**	**TGI**	**CPU**	**nAper**	**BOT**	**Time**	**TGI**	**CPU**
c1b1	10	41	111	102	1.1	11	38	115	50	5.5
c1b2	10	34	104	80	0.8	8	23	79	14	0.7
c1b3	11	31	108	97	11.4	9	28	91	59	1.0
c1b4	11	33	110	74	37.0	11	37	114	146	7.0
c1b5	10	34	104	133	4.3	8	32	88	49	1.2
c2b1	14	34	132	134	26.5	12	30	114	187	11.5
c2b2	13	41	132	159	20.1	11	33	110	192	8.0
c2b3	13	49	140	245	14.7	11	28	105	151	3.1
c2b4	14	51	149	316	87.3	12	34	118	148	8.3
c2b5	13	41	132	217	395.6	10	27	97	120	2.0
c3b1	13	41	132	323	310.0	14	40	138	254	23.0
c3b2	14	46	144	320	4759.8	8	23	79	86	1.1
c3b3	13	49	140	533	10373.9	12	40	124	360	18.6
c3b4	12	44	128	481	524.9	12	40	124	327	428.2
c3b5	13	34	125	133	3.3	9	27	90	75	2.6
c4b1	16	40	152	216	34.9	12	46	130	244	10.6
c4b2	16	69	181	450	20901.0	9	27	90	149	15.8
c4b3	14	41	139	130	44.7	10	32	102	129	3.3
c4b4	14	44	142	246	164.3	10	27	97	163	8.0
c4b5	17	76	195	470	14511.4	9	24	87	48	4.0
c5b1	10	26	96	68	0.5	10	35	105	41	0.5
c5b2	12	41	125	59	14.3	8	25	81	27	0.6
c5b3	10	34	104	155	3.1	9	23	86	42	1.0
c5b4	12	40	124	105	2.2	10	32	102	87	4.3
c5b5	12	46	130	151	51.9	8	31	87	17	0.8

Table 3: Effect of Rotating the MLC Head

Motivated by this observation, we modify our algorithm to directly solve for the best orientation by using obtained upper and lower bounds to quickly prove whether rotating the MLC head is beneficial. Assume that we have lower and upper bounds for the row-convex and column-convex problems, and suppose that the lower bound of the row-convex problem is greater than the upper bound of the column-convex problem. In this case, we can conclude that an optimal solution minimizing total treatment time for the given fluence map must be a column-convex decomposition. We use this argument to solve one of the problems, and then use the bound information to avoid having to solve the other one to optimality. We pick the first problem to solve by selecting one having the least initial lower bound, breaking ties if applicable by choosing the problem for which $n < m$, since the subproblems tend to solve faster for smaller values of n. Table 4 shows the nAper, BOT, TGI, and CPU metrics obtained from our algorithm enhanced with the above bounding scheme, corresponding to the "Total Time," "# Apertures," and "Lexicographic" objectives. Observe that all 25 instances, under any metric, terminate in under 15 minutes of CPU time with a solution that is optimal with respect to the corresponding objective, and all instances are solved to optimality within a minute using the "Lexicographic" algorithm.

Recall that the "BOT" column in "Lexicographic" reports the minimum achievable beam-on-time, and the "nAper" column under the objective "# Apertures" reports the minimum number of apertures needed to decompose each instance. Perhaps surprisingly, in comparing these values with the results of "Total Time," we observe that there exists a solution that minimizes both the number of shapes and the beam-on-time simultaneously in 19 of the 25 instances.

Finally, we analyze performance of the three heuristics on clinical data, where we execute each heuristic on each problem instance and its transpose (corresponding to row-convex and column-convex decompositions), and pick the solution yielding the smallest treatment time. Table 5 shows the number of apertures, beam-on-time, TGI metrics for each solution as well as the CPU time spent by each heuristic. Comparison with the "Total Time" columns in Table 4 reveals that even though the heuristics consistently generated high-quality solutions, the Siochi and Engel heuristics were able to find an optimal solution in only five problem instances, and Xia-Verhey heuristic could not find an optimal solution to any instance.

	Total Time				# Apertures				Lexicographic			
Name	**nAper**	**BOT**	**TGI**	**CPU**	**nAper**	**BOT**	**TGI**	**CPU**	**nAper**	**BOT**	**TGI**	**CPU**
c1b1	10	41	102	4.7	10	41	102	2.3	11	38	50	4.8
c1b2	8	23	14	1.1	8	23	14	1.1	8	23	14	1.1
c1b3	9	28	59	3.0	9	28	59	4.5	9	28	59	4.5
c1b4	11	33	74	41.2	11	37	128	27.1	11	33	74	12.2
c1b5	8	32	49	2.1	8	34	56	1.3	9	26	9	1.9
c2b1	12	30	187	15.6	12	30	187	14.9	12	30	187	14.4
c2b2	11	33	192	10.8	11	38	161	6.9	11	33	146	7.8
c2b3	11	28	149	8.9	11	28	113	9.9	11	28	197	10.8
c2b4	12	34	148	16.8	12	34	148	16.8	12	34	148	17.1
c2b5	10	27	120	6.1	10	31	155	6.2	10	27	120	6.2
c3b1	13	41	323	315.0	12	51	521	62.1	13	41	325	31.4
c3b2	8	23	86	4.4	8	26	87	4.5	8	23	62	5.6
c3b3	12	40	360	27.4	12	40	360	894.7	12	40	365	20.1
c3b4	12	40	327	442.2	12	46	284	548.8	13	38	928	55.1
c3b5	9	27	75	5.6	9	27	75	5.4	9	27	75	5.7
c4b1	12	46	244	16.8	12	46	227	10.6	12	46	227	11.3
c4b2	9	27	149	45.5	9	32	150	56.2	9	27	135	35.0
c4b3	10	32	129	15.7	10	34	108	14.9	10	32	129	15.6
c4b4	10	27	163	32.0	10	28	112	32.6	11	26	72	29.9
c4b5	9	24	48	27.8	9	24	48	27.7	9	24	48	27.0
c5b1	10	26	68	1.2	10	26	68	1.2	10	26	68	1.2
c5b2	8	25	27	1.1	8	25	27	1.0	9	23	8	1.1
c5b3	9	23	42	3.6	9	24	45	3.2	9	23	83	3.1
c5b4	10	32	87	5.8	10	41	101	2.7	10	32	87	2.8
c5b5	8	31	17	1.4	8	33	16	1.2	8	31	71	1.1

Table 4: Computational Results for Our Base Algorithm

	Siochi				Engel				Xia-Verhey			
Name	**nAper**	**BOT**	**TGI**	**CPU**	**nAper**	**BOT**	**TGI**	**CPU**	**nAper**	**BOT**	**TGI**	**CPU**
c1b1	11	38	245	14.0	12	38	261	< 1	13	40	219	< 1
c1b2	8	23	109	3.0	8	23	127	< 1	10	32	133	< 1
c1b3	9	28	213	4.0	10	28	192	< 1	12	34	198	< 1
c1b4	12	34	306	9.5	11	37	398	< 1	14	42	355	< 1
c1b5	9	26	103	3.6	9	26	175	< 1	12	35	124	< 1
c2b1	12	30	652	11.2	12	30	738	< 1	15	45	635	< 1
c2b2	12	33	395	17.8	12	33	464	< 1	15	45	460	< 1
c2b3	12	28	625	34.8	12	28	429	< 1	15	43	459	< 1
c2b4	12	34	628	43.2	12	34	723	< 1	18	56	417	< 1
c2b5	11	27	463	15.3	11	27	465	< 1	14	41	375	< 1
c3b1	14	43	828	36.3	15	40	1054	< 1	17	55	765	< 1
c3b2	9	23	143	11.1	9	23	127	< 1	12	36	289	< 1
c3b3	14	40	1316	40.8	14	40	869	< 1	19	60	1038	< 1
c3b4	13	48	678	33.3	14	38	765	< 1	17	55	553	< 1
c3b5	9	28	263	7.0	9	27	325	< 1	13	45	261	< 1
c4b1	13	46	617	29.4	14	46	625	< 1	18	62	531	< 1
c4b2	10	29	295	73.8	10	27	466	< 1	14	44	350	< 1
c4b3	11	32	339	19.4	11	32	365	< 1	14	48	428	< 1
c4b4	11	26	489	13.5	11	26	540	< 1	15	46	424	< 1
c4b5	9	24	236	89.8	9	24	328	< 1	15	44	328	< 1
c5b1	11	26	188	4.6	12	26	176	< 1	12	38	185	< 1
c5b2	9	23	129	6.9	9	23	100	< 1	10	33	145	< 1
c5b3	9	26	201	5.1	10	23	293	< 1	12	32	189	< 1
c5b4	11	32	218	11.2	11	32	322	< 1	13	46	243	< 1
c5b5	8	32	217	7.2	9	31	211	< 1	11	35	138	< 1

Table 5: Comparison of Heuristic Algorithms on Clinical Data

4 Conclusions and Future Research

In this paper we have described an exact decomposition algorithm for solving a leaf sequencing problem arising in IMRT treatment planning. Our algorithm is based on an integer programming model for finding a multiset of intensity values to be assigned to apertures, and a backtracking algorithm that forms apertures by finding compatible leaf positions for each row. Computational results show that the vast majority of randomly generated and clinical problem instances in our data set can be solved to optimality within a few minutes. Our algorithm is flexible enough to handle a class of related problems with minor modifications, and is capable of quantifying the effect of rotating the MLC head. Furthermore, our computational results show that it is capable of obtaining significant reductions in the tongue-and-groove effect compared to several heuristics in clinical use. As such, not only can this algorithm reasonably be used in real clinical settings, but also the bounds obtained from our algorithm can serve as benchmark criteria to compare the performance of heuristic methods. Our benchmarks on the heuristic methods of Siochi [38], Engel [11], and Xia and Verhey [41] reveal that while these methods are capable of consistently identifying good solutions, they rarely find an optimal solution. On average they generate solutions that use at least 11.8% more apertures than an optimal solution when tested on clinical data, and 15.6% when tested on randomly generated problem instances.

The algorithm we have described assumes that leaves corresponding to different rows can be positioned independently, and exploits this assumption to decompose the problem by rows. Therefore, it is not directly applicable for problems in which there are other restrictions on aperture shapes that can be delivered by the available machinery, such as interdigitation or connectedness constraints. We are planning to conduct further research in order to generalize our algorithm to account for such additional technological constraints on the aperture shapes.

References

[1] N. Agazaryan and T. D. Solberg. Segmental and dynamic intensity-modulated radiotherapy delivery techniques for micro-multileaf collimator. *Medical Physics*, 30(7):1758–1765, 2003.

[2] R. K. Ahuja and H. W. Hamacher. A network flow algorithm to minimize beam-on-time for unconstrained multileaf collimator problems in cancer radiation therapy. *Networks*, 45(1):36–41, 2005.

[3] D. Baatar. *Matrix Decomposition with Time and Cardinality Objectives: Theory, Algorithms, and Application to Multileaf Collimator Sequencing*. PhD thesis, Department of Mathematics, Technische Universität Kaiserslautern, Kaiserslautern, Germany, 2005.

[4] D. Baatar, N. Boland, S. Brand, and P. J. Stuckey. Minimum cardinality matrix decomposition into consecutive-ones matrices: CP and IP approaches. In *Integration of AI and OR Techniques in Constraint Programming for Combinatorial Optimization Problems*, 2007.

[5] D. Baatar, H. W. Hamacher, M. Ehrgott, and G. J. Woeginger. Decomposition of integer matrices and multileaf collimator sequencing. *Discrete Applied Mathematics*, 152(1):6–34, 2005.

[6] N. Boland, H. W. Hamacher, and F. Lenzen. Minimizing beam-on time in cancer radiation treatment using multileaf collimators. *Networks*, 43(4):226–240, 2004.

[7] T. R. Bortfeld, D. L. Kahler, T. J. Waldron, and A. L. Boyer. X-ray field compensation with multileaf collimators. *International Journal of Radiation Oncology Biology Physics*, 28(3):723–730, 1994.

[8] J. Dai and Y. Zhu. Minimizing the number of segments in a delivery sequence for intensity-modulated radiation therapy with a multileaf collimator. *Medical Physics*, 28(10):2113–2120, 2001.

[9] J. Deng, T. Pawlicki, Y. Chen, J. Li, S. B. Jiang, and C. M. Ma. The MLC tongue-and-groove effect on IMRT dose distributions. *Physics in Medicine and Biology*, 46(4):1039–1060, 2001.

[10] M. Ehrgott, H. W. Hamacher, and M. Nußbaum. Decomposition of matrices and static multileaf collimators: A survey. In C. J. S. Alves, P. M. Pardalos, and L. N. Vicente, editors, *Optimization in Medicine*, 2008.

[11] K. Engel. A new algorithm for optimal multileaf collimator field segmentation. *Discrete Applied Mathematics*, 152(1):35–51, 2005.

[12] A. T. Ernst, V. H. Mak, and L. A. Mason. An exact method for the minimum cardinality problem in the planning of IMRT. *INFORMS Journal on Computing (to appear)*, 2009.

[13] M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman & Co., New York, New York, 1979.

[14] H. W. Hamacher and K.-H. Küfer. Inverse radiation therapy planning – a multiple objective optimization approach. *Discrete Applied Mathematics*, 118(1):145–161, 2002.

[15] T. Kalinowski. The algorithmic complexity of the minimization of the number of segments in multileaf collimator field segmentation. Technical report, Fachbereich Mathematik, Universität Rostock, Rostock, Germany, August 2004.

[16] T. Kalinowski. A duality based algorithm for multileaf collimator field segmentation with interleaf collision constraint. *Discrete Applied Mathematics*, 152(1):52–88, 2005.

[17] T. Kalinowski. Reducing the number of monitor units in multileaf collimator field segmentation. *Physics in Medicine and Biology*, 50(6):1147–1161, 2005.

[18] S. Kamath, S. Sahni, J. Li, J. Palta, and S. Ranka. Leaf sequencing algorithms for segmented multileaf collimation. *Physics in Medicine and Biology*, 48(3):307–324, 2003.

[19] S. Kamath, S. Sahni, J. Palta, and S. Ranka. Algorithms for optimal sequencing of dynamic multileaf collimators. *Physics in Medicine and Biology*, 49(1):33–54, 2004.

[20] S. Kamath, S. Sahni, J. Palta, S. Ranka, and J. Li. Optimal leaf sequencing with elimination of tongue-and-groove underdosage. *Physics in Medicine and Biology*, 49(3):N7–N19, 2004.

[21] S. Kamath, S. Sahni, S. Ranka, J. Li, and J. Palta. A comparison of step-and-shoot leaf sequencing algorithms that eliminate tongue-and-groove effects. *Physics in Medicine and Biology*, 49(14):3137–3143, 2004.

[22] S. Kamath, S. Sahni, S. Ranka, J. Li, and J. Palta. Optimal field splitting for large intensity-modulated fields. *Medical Physics*, 31(12):3314–3323, 2004.

[23] K.-H. Küfer, M. Monz, A. Scherrer, H. L. Trinkaus, T. Bortfeld, and C. Thieke. Intensity Modulated Radiotherapy - a large scale multi-criteria programming problem. *OR Spektrum*, 25:223–249, 2003.

[24] M. Langer, V. Thai, and L. Papiez. Improved leaf sequencing reduces segments or monitor units needed to deliver IMRT using multileaf collimators. *Medical Physics*, 28(12):2450–2458, 2001.

[25] E. K. Lee, T. Fox, and I. Crocker. Optimization of radiosurgery treatment planning via mixed integer programming. *Medical Physics*, 27(5):995–1004, 2000.

[26] E. K. Lee, T. Fox, and I. Crocker. Integer programming applied to intensity-modulated radiation treatment planning. *Annals of Operations Research*, 119:165–181, 2003.

[27] F. Lenzen. An integer programming approach to the multileaf collimator problem. Master's thesis, University of Kaiserslautern, Department of Mathematics, Kaiserslautern, Germany, June 2000.

[28] G. J. Lim and J. Choi. A two-stage integer programming approach for optimizing leaf sequence in IMRT. Technical report, IE Tech Report, IE0807-01, University of Houston, Houston, Texas, 2007.

[29] F. Preciado-Walters, R. Rardin, M. Langer, and V. Thai. A coupled column generation, mixed integer approach to optimal planning of intensity modulated radiation therapy for cancer. *Mathematical Programming*, 101(2):319–338, 2004.

[30] W. Que. Comparison of algorithms for multileaf collimator field segmentation. *Medical Physics*, 26(11):2390–2396, 1999.

[31] W. Que, J. Kung, and J. Dai. "Tongue-and-groove" effect in intensity modulated radiotherapy with static multileaf collimator fields. *Physics in Medicine and Biology*, 49(3):399–405, 2004.

[32] H. E. Romeijn, R. K. Ahuja, J. F. Dempsey, and A. Kumar. A column generation approach to radiation therapy treatment planning using aperture modulation. *SIAM Journal on Optimization*, 15(3):838–862, 2005.

[33] H. E. Romeijn, R. K. Ahuja, J. F. Dempsey, and A. Kumar. A new linear programming approach to radiation therapy treatment planning problems. *Operations Research*, 54(2):201–216, 2006.

[34] D. M. Shepard, M. A. Earl, X. A. Li, S. Naqvi, and C. Yu. Direct aperture optimization: a turnkey solution for step-and-shoot IMRT. *Medical Physics*, 29(6):1007–1018, 2002.

[35] D. M. Shepard, M. C. Ferris, G. H. Olivera, and T. R. Mackie. Optimizing the delivery of radiation therapy to cancer patients. *SIAM Review*, 41(4):721–744, 1999.

[36] R. A. C. Siochi. Minimizing static intensity modulation delivery time using an intensity solid paradigm. *International Journal of Radiation Oncology Biology Physics*, 43(3):671–680, 1999.

[37] R. A. C. Siochi. Modifications to the IMFAST leaf sequencing optimization algorithm. *Medical Physics*, 31(12):3267–3278, 2004.

[38] R. A. C. Siochi. Variable depth recursion algorithm for leaf sequencing. *Medical Physics*, 34(2):664–672, 2007.

[39] A. M. Stell, O. A. Zeidan, J. G. Li, and J. F. Dempsey. An extensive log-file analysis of step-and-shoot IMRT subfield delivery errors. *International Journal of Radiation Oncology Biology Physics*, 31(6):1593–1602, 2004.

[40] J. P. C. Van Santvoort and B. J. M. Heijmen. Dynamic multileaf collimation without 'tongue-and-groove' underdosage effects. *Physics in Medicine and Biology*, 41(10):2091–2105, 1996.

[41] P. Xia and L. J. Verhey. Multileaf collimator leaf sequencing algorithm for intensity modulated beams with multiple static segments. *Medical Physics*, 25(8):1424–1434, 1998.

Appendix A

In this appendix we discuss an integer programming approach to decomposing a fluence map into a number of apertures and corresponding intensities that is based on a model proposed by Langer et al. [24]. Given a maximum number of unit-intensity apertures, say T, this formulation determines the positions of the left and right leaves in each row of each of these apertures. We develop the model by separately studying four components:

- *Fluence map requirements.* Define, for each aperture $t = 1, \ldots, T$ and each bixel $(i, j) \in \{1, \ldots, m\} \times \{1, \ldots, n\}$, a binary variable d_{ij}^t that is equal to one if and only if bixel (i, j) is exposed, i.e., not covered by a left leaf or a right leaf. Since each aperture has unit intensity, the following constraints then ensure that the desired fluence map is delivered:
$$\sum_{t=1}^{T} d_{ij}^t = b_{ij} \quad \forall\ i = 1, \ldots, m,\ j = 1, \ldots, n. \tag{34}$$
- *Aperture deliverability constraints.* Define, for each aperture $t = 1, \ldots, T$ and each bixel $(i, j) \in \{1, \ldots, m\} \times \{1, \ldots, n\}$, binary variables p_{ij}^t and l_{ij}^t that are equal to one if and only if bixel (i, j) is covered by the right leaf or the left leaf in row i of aperture t, respectively. The following set of constraints then ensure that each of the T apertures

is deliverable:

$$\begin{aligned} p_{ij}^t + l_{ij}^t + d_{ij}^t &= 1 \qquad \forall\ i = 1, \ldots, m,\ j = 1, \ldots, n,\ t = 1, \ldots, T && (35) \\ p_{ij}^t &\leq p_{i,j+1}^t \quad \forall\ i = 1, \ldots, m,\ j = 1, \ldots, n-1,\ t = 1, \ldots, T && (36) \\ l_{ij}^t &\leq l_{i,j-1}^t \quad \forall\ i = 1, \ldots, m,\ j = 2, \ldots, n,\ t = 1, \ldots, T. && (37) \end{aligned}$$

In particular, constraints (35) state that each bixel is either covered by a right-hand leaf, covered by a left-hand leaf, or uncovered (where the d-variables are included only for convenience and can be substituted out of the formulation). Constraints (36) and (37) state that if any bixel (i, j) is covered by a right-hand leaf (resp. left-hand leaf), then bixel $(i, j + 1)$ (resp. $(i, j - 1)$) should be covered by a right-hand leaf (resp. left-hand leaf) as well.

- *Beam-on-time.* We associate a binary variable z^t with each aperture $t = 1, \ldots, T$ that is equal to one if there are uncovered bixels in aperture t and zero otherwise, so that the beam-on-time is simply given by

$$\sum_{t=1}^{T} z^t. \tag{38}$$

While Langer et al. [24] impose the following constraints to ensure that these variables have (at least) their desired value:

$$\sum_{i=1}^{m} \sum_{j=1}^{n} d_{ij}^t \leq (mn) z^t \quad \forall\ t = 1, \ldots, T, \tag{39}$$

we note that the following stronger formulation, which would actually not require enforcing the z-variables to be binary, can be obtained by disaggregating (39).

$$d_{ij}^t \leq z^t \quad \forall\ i = 1, \ldots, m,\ j = 1, \ldots, n,\ t = 1, \ldots, T. \tag{39$'$}$$

Note that this model allows z^t to be equal to one even if in aperture t no bixels are exposed, so that formally speaking (38) is an upper bound on the beam-on-time. The objective function will ensure that the z-variables take on their minimum possible value.

- *Number of apertures.* We associate a binary variable g_t with each aperture $t = 1, \ldots, T-1$ that is equal to one if aperture t is different from aperture $t+1$ and zero otherwise. The number of setups is then given by

$$\sum_{t=1}^{T} g_t. \tag{40}$$

(If any aperture is used more than once but separated by another one, we consider the second occurrence of the aperture to be a new setup. However, when minimizing total treatment time there will always exist an optimal solution in which identical apertures are delivered sequentially.) Now let c_{ij}^t and u_{ij}^t be auxiliary binary variables such that the former is equal to one if bixel (i,j) is exposed in aperture t but not in aperture $t+1$ and zero otherwise, and the latter is equal to one if bixel (i,j) is covered in aperture t but not in aperture $t+1$. This relationship is stated by

$$-c_{ij}^t \leq d_{ij}^{t+1} - d_{ij}^t \leq u_{ij}^t \quad \forall\ i = 1, \ldots, m,\ j = 1, \ldots, n,\ t = 1, \ldots, T-1. \tag{41}$$

Langer et al. [24] then use the following constraints to ensure that the variables g_t have (at least) their desired value:

$$\sum_{i=1}^{m} \sum_{j=1}^{n} \left(c_{ij}^t + u_{ij}^t\right) \leq (mn) g^t \quad \forall\ t = 1, \ldots, T-1. \tag{42}$$

However, note that again a stronger set of inequalities (that permit g to be equivalently relaxed as continuous variables) is obtained using disaggregation:

$$c_{ij}^t + u_{ij}^t \leq g^t \quad \forall\ i = 1, \ldots, m,\ j = 1, \ldots, n,\ t = 1, \ldots, T-1. \tag{42$'$}$$

Similar to the case of the beam-on-time, this model allows g^t to be equal to one even if apertures t and $t+1$ are identical, although our objective function will ensure that the g-variables are chosen sufficiently small.

Langer et al. [24] then study the problem of minimizing the number of setups (40) subject to the constraints (35)–(37), (39), (42), the constraint that the beam-on-time is minimal:

$$\sum_{t=1}^{T} z^t \leq \tilde{z} \tag{43}$$

and binary constraints on the variables, where we recommend determining $\tilde{z}$ via one of the polynomial-time procedures mentioned in Section 1. We note that an equivalent model is obtained by simply setting $T = \tilde{z}$, which reduces the problem dimension, and hence should be more efficient than adding a beam-on-time constraint.

We wish to minimize the total treatment time as measured by

$$w_1 \sum_{t=1}^{T} g_t + w_2 \sum_{t=1}^{T} z^t \tag{44}$$

subject to constraints (35)–(37), (39′), (42′), and binary constraints on the appropriate variables (and hence we do not impose (43)).

Appendix B

Proposition 1. C1-Partition *is strongly NP-complete.*

Proof. Let ξ be the subset of $\{1, \ldots, L\}$ such that $\ell \in \xi$ if and only if $x_\ell > 0$. Formally speaking, the problem size is given by $\log_2(L)$, n, and $|\xi|$ (since the zero entries of $\mathbf{x}$ need not be encoded).

Let $\mathcal{K}$ denote the set of all $O(n^2)$ n-dimensional binary vectors whose ones appear consecutively, where $\mathbf{v}_k$ is the binary vector corresponding to $k \in \mathcal{K}$. Consider a guessed solution that consists of $|\xi|$-dimensional nonnegative integer vectors $\mathbf{d}_k$, $\forall k \in \mathcal{K}$, where $d_{k\ell}$ denotes the number of times leaf position $\mathbf{v}_k$, $k \in \mathcal{K}$, is assigned to intensity $\ell \in \xi$. Since all $d_{k\ell} \leq L$ in some feasible solution, we restrict the guessed $\mathbf{d}$-vectors as such. The size of the guessed vectors is thus $O(n^2|\xi| \log_2(L))$. We can verify whether or not $\sum_{k\in\mathcal{K}} \sum_{\ell\in\xi} d_{k\ell}\mathbf{v}_k = \mathbf{b}$ in $O(n^2|\xi|)$ additions. Therefore, C1-Partition is in NP.

In order to show that C1-Partition is NP-complete, we reduce 3-Partition to it. 3-Partition is a strongly NP-complete problem and seeks whether a given multiset of integers can be partitioned into triplets having the same sum. Formally, it can be defined as follows (see Garey and Johnson [13]):

3-Partition
INSTANCE: A multiset A of 3ν positive integers $a_1, \ldots, a_{3\nu}$ and a positive integer B such

that $B/4 < a_i < B/2$ for $i = 1, \ldots, 3\nu$ and such that $\sum_{i=1}^{3\nu} a_i = \nu B$.
QUESTION: Can A be partitioned into ν disjoint multisets $A_1, \ldots, A_\nu$ such that $\sum_{j \in A_i} a_j = B$ for $i = 1, \ldots, \nu$?

Given an arbitrary instance of 3-PARTITION, we construct an instance for C1-PARTITION as follows. First, we define $\hat{x}$ to be an integer vector whose ℓ^{th} component, $\hat{x}_\ell$, is equal to the number of indices i for which $a_i = \ell$. Furthermore, we let $\mathbf{b}$ be a $(2\nu - 1)$-dimensional vector of the form $[B \;\; 0 \;\; B \;\; 0 \;\; \cdots \;\; 0 \;\; B]$. We construct a feasible solution to C1-PARTITION that employs only the odd-indexed unit vectors of $\mathcal{K}$. Denote these vectors as $\mathbf{e}_1, \mathbf{e}_3, \ldots, \mathbf{e}_{2\nu-1}$, and index their associated $\mathbf{d}$-vectors as $\mathbf{d}_1, \mathbf{d}_3, \ldots, \mathbf{d}_{2\nu-1}$.

Assume that the 3-PARTITION instance is a yes-instance, and hence there exist multisets $A_1, \ldots, A_\nu$ such that $\sum_{j \in A_i} a_j = B$. In this case, a feasible solution of the C1-PARTITION instance lets $d_{2j-1,\ell}$ be the number of elements of intensity ℓ in A_j, for each $j = 1, \ldots, \nu$, and assigns $d_{k\ell} = 0$ for all other k. Similarly, suppose that the C1-PARTITION instance is a yes-instance. Since all positive values in $\mathbf{b}$ are adjacent to 0, in any feasible solution to the instance of C1-PARTITION, we may only use leaf positions that expose a single odd-index bixel. Also, since $B/4 < a_i < B/2, \; \forall \; i = 1, \ldots, 3\nu$, vector $\mathbf{d}_k$ must be used to deliver exactly three intensity values, for $k = 1, 3, \ldots, 2\nu - 1$. Then a feasible solution of the 3-PARTITION instance is given as multisets $A_1, \ldots, A_\nu$ recovered from $\mathbf{d}_1, \mathbf{d}_3, \ldots, \mathbf{d}_{2\nu-1}$ as described above. Therefore, an arbitrary 3-PARTITION instance is a yes-instance if and only if the corresponding transformed C1-PARTITION instance is a yes-instance. Since 3-PARTITION is strongly NP-complete, and since the transformation provided is polynomial in terms of the size of the problem and the instance data, it follows that C1-PARTITION is also strongly NP-complete. □