Improved leaf sequencing reduces segments or monitor units needed to deliver IMRT using multileaf collimators

Mark Langer,[a)] Van Thai, and Lech Papiez
Department of Radiation Oncology, Indiana University Medical School, Indianapolis, Indiana 46202

(Received 21 February 2001; accepted for publication 1 October 2001)

Leaf sequencing algorithms may use an unnecessary number of monitor units or segments to generate intensity maps for delivery of intensity modulated radiotherapy (IMRT) using multiple static fields. An integer algorithm was devised to generate a sequence with the fewest possible segments when the minimum number of monitor units are used. Special hardware related restrictions on leaf motion can be incorporated. The algorithm was tested using a benchmark map from the literature and clinical examples. Results were compared to sequences given by the routine of Bortfeld that minimizes monitor units by treating each row independently, and the areal or reducing routines that use fewer segments at the price of more monitor units. The Bortfeld algorithm used on average 58% more segments than provided by the integer algorithm with bidirectional motion and 32% more segments than did an integer algorithm admitting only unidirectional sequences. The areal algorithm used 48% more monitor units and the reducing algorithm used 23% more monitor units than did the bidirectional integer algorithm, while the areal and reducing algorithms used 23% more segments than did the integer algorithm. Improved leaf sequencing algorithms can allow more efficient delivery of static field IMRT. The integer algorithm demonstrates the efficiencies possible with an improved routine and opens a new avenue for development. © *2001 American Association of Physicists in Medicine.* [DOI: 10.1118/1.1420392]

Key words: intensity modulated radiotherapy, integer programming, leaf sequencing

I. INTRODUCTION

The intensity profile of a radiation beam can be modified by using a multileaf collimator to block different portions of the beam for different lengths of time. Different leaf sequences may be used to satisfy the rules for generating an intensity map.[1] The sequences describe the setup of leaf positions as a function of the beam monitor units. When the leaves are moved in discrete steps with the beam off from one setup of positions to the next, two important measures of sequence efficiency have been identified. One is the total number of monitor units needed to generate the map and the other is the number of segments, or setups of the leaves. It is desirable to produce a leaf sequence in which both the monitor units and number of segments are low. More monitor units unfavorably affect treatment delivery by increasing the component of machine leakage and lengthening each treatment session. A lengthened treatment session worsens machine throughput and leads to inaccuracies in patient positioning, while machine leakage is a source of discrepancy between the planned and delivered dose distribution. More segments also lengthen the treatment session because of the time needed to switch the beam on and off and to move the leaves. One report stated that the number of segments is a more important factor for lengthening session time than the cumulative number of monitor units.[2]

An efficient sequence of leaf movements should take the fewest possible segments to generate a map with the number of monitor units that it uses. It should also use the minimum number of monitor units for the number of segments it takes. If either condition is violated, the sequence can be improved by either keeping the number of monitor units constant and reducing the number of segments, or else keeping constant the number of segments and reducing the number of monitor units. An extensive literature records the substantial number of leaf sequencing algorithms that have been developed in industry and academia.[3–7] The number of monitor units and number of segments required by the algorithms have been compared against each other using random maps and clinical examples. However, their performance measures have not been gauged against an absolute standard. The minimum number of segments needed to generate a sequence using the minimum number of monitor units has not been found. If this number were available, it could serve as a test of proposed algorithms and motivate the construction of improved techniques. Substantial excesses in number of monitor units or number of segments would argue for further research in algorithm design and the replacement of current techniques with improved methods.

This report finds the minimum number of segments needed to generate a leaf sequence when the minimum number of monitor units are expended. The sequences corresponding to the minimum are shown and the solutions were verified by hand for smaller problems and by automated routine for larger ones. The results were obtained by constructing an integer program to give a provably optimal solution. The solutions are compared to those given by algorithms that represent two well-described approaches to leaf sequencing solutions. One algorithm, credited to Bortfeld, uses the fewest possible monitor units but may use more segments than this condition requires.[5] The other approach, represented by the areal and reducing algorithms, aims to reduce the number of segments at the price of an increased number of monitor units.[6] This study asks whether it is possible to reduce the number of segments without increasing the number of monitor units beyond the minimum required.

Comparisons were made using the general form of the published algorithms. Machine specific restrictions on leaf position were not introduced to avoid the uncertainties or biases that can arise in adding new conditions onto a basic algorithm description. The tests compare the algorithms under the most general conditions, but relative performances may differ when conditions specific to a particular manufacturer's systems are considered. Nevertheless, one series[6] has been interpreted as showing that the addition of a constraint on leaf overtravel had little effect upon the relative performance of different algorithms as measured by number of segments or fluence.[8]

The consequences of introducing additional constraints were examined using the integer algorithm. Using the integer algorithm, the effects of two leaf constraints on the minimum number of segments needed to complete a sequence with the minimum number of monitor units were individually found. One constraint forbade overtravel of leaf ends along adjacent rows (sometimes called a collision constraint). A second constraint eliminated reversal between segments in the states of any two bixels, one opened and one closed, that lie adjacent along a column.[7] This condition, referred to as a tongue and groove constraint, aims to combine segment elements when possible so as to avoid underdosing along the common border of two bixels, each found open at least once during a segment in which the adjacent bixel is closed by a leaf side. The tongue and groove construction of a leaf side disturbs the fluence pattern and the planned and delivered doses better match when the interference of a leaf side is not present. The studies measure the costs, in the number of segments and the number of monitor units, of introducing the leaf movement constraints.

II. METHODS

Evaluations were made using intensity maps derived from clinical examples and an arbitrary map found in the published literature to serve as a benchmark.[9] Intensity maps were related to the beam times assigned to each leaf position according to established rules for the construction of leaf sequencing algorithms.[6,10,8,1,9,7] The rules take the fluence of a beam element to be proportional to its cumulative exposure measured in monitor units. The minimum number of segments needed to generate a leaf sequence with the minimum number of monitor units was found using an integer program. The results were compared to the number of monitor units and the number of segments expended by other solution approaches. The accumulated number of monitor units and number of segments for all the leaf sequences were cross-checked by a single computer program. Leaf sequences were explicitly exhibited for the smaller problems to allow manual inspection. Cumulative totals of the number of segments and the number of monitor units over all beam positions were prepared for each individual case, and summed over the several cases studied.

A. Generation of a sequence using the minimum number of segments under the condition that the number of monitor units be at its minimum

The intensity map for the beam is decomposed into an integer matrix whose values assign intensities to the corresponding elements of the beam. The beam is divided into rectangular elements by the rows along which each leaf pair moves, and by the columns that mark out, at constant intervals, the possible leaf positions along a row. Two binary (0/1) variables index the state of every beam element at each monitor unit. One binary variable indicates whether the beam element is covered by the right leaf during delivery of the labeled monitor unit, and the other variable indicates whether the element is covered by the left leaf. Specifically, the binary variable p_{ij}^t takes the value 1 if the beam element falling in row *i*, column *j* is covered by a right leaf when the *t*-th monitor unit is delivered but takes the value zero otherwise. Similarly, the binary variable l_{ij}^t takes the value 1 if the left leaf covers the element. These two variables determine the value of a third binary variable, d_{ij}^t, that takes the value 1 when a monitor unit is delivered through the element at period *t* because it is covered by neither leaf. The three variables are related by the condition that the leaves in any row cannot override each other, meaning that a beam element cannot be covered at the same time by both a left leaf and a right leaf. The relation among the three variables is given by

$$p_{ij}^t + l_{ij}^t = 1 - d_{ij}^t \tag{1}$$

where $p_{ij}^t, l_{ij}^t, d_{ij}^t \in \{0,1\}$.

The equation in binary variables can hold only if p_{ij}^t and l_{ij}^t are not both equal to 1, meaning that the two leaves cannot simultaneously cover beam element *ij*. If one variable takes the value 1, then d_{ij}^t must take the value 0, meaning that an element is covered by some leaf so no radiation can pass through. If p_{ij}^t and l_{ij}^t are both zero, the element is uncovered and d_{ij}^t is set to 1, indicating transmittal of a unit intensity during time period *t*.

The most general description of a leaf is that it has no holes. If a leaf covers a beam element, then every element between it and the side of the collimator to which the leaf is connected is also covered. This rule is established by the following two inequalities, with the columns numbered from left to right:

$$p_{ij}^t \leq p_{ij+1}^t, \tag{2}$$

$$l_{ij+1}^t \leq l_{ij}^t, \tag{3}$$

which must hold for all pairs of adjacent columns.

Finally, the sum d_{ij}^t must equal the desired intensity, or

$$\sum_{t=1}^{T} d_{ij}^t = I_{ij}, \tag{4}$$

where I_{ij} is the intensity assigned to beam element *ij* and T is an upper bound on the number of monitor units that can be required. The upper bound is formed as $\max_i \Sigma_j I_{ij}$, found by summing the elements of the intensity map in each row, and choosing the largest of the sums.

The total number of monitor units expended is tallied by introducing a new binary variable z^t that takes the value 1 if at least one beam element remains exposed when the t-th monitor unit in the sequence is delivered. The role of z^t is established by the following inequality:

$$\sum_i^I \sum_j^J d_{ij}^t \leqslant z^t IJ \tag{5}$$

where I and J are the number of rows and columns, respectively, in the matrix of beam elements. The objective of the integer program is to minimize the total number of monitor units needed to complete the intensity map, as expressed by

$$\min \sum_t^T z^t = Z. \tag{6}$$

Relations (1)–(5) together with objective (6) describe an integer program, which can be solved using well-described methods to give the minimum number of monitor units required to generate the intensity map when no restrictions on the number of segments is placed. Once the value for the minimum number of monitor units, $\tilde{Z}$, is found the program can be expanded to find the minimum number of segments needed to complete the map using the minimum number of monitor units. First, the objective (6) is replaced by the constraint (6′):

$$\sum_t^T z^t \leqslant \tilde{Z}. \tag{6′}$$

Next, a binary variable, g^t, is used to tally the number of segments. It takes the value 1 if any element switches from covered to uncovered or from uncovered to covered between the delivery of one monitor unit and the next. Its role is established by creating a pair of binary variables c_{ij}^t and u_{ij}^t to track whether element *ij* switches to the covered or uncovered position, respectively, between monitor unit t and $t+1$. The variables take the value 1 if a change is made, according to the following relation:

$$-c_{ij}^t \leqslant d_{ij}^{t+1} - d_{ij}^t \leqslant u_{ij}^t \tag{7}$$

where $u_{ij}^t, c_{ij}^t \in \{0,1\}$.

A summary variable, s_{ij}^t, indicates whether a switch to either the covered or uncovered position occurs in beam element *ij*. It bears the value 0 only if the state of the element does not change, according to

$$u_{ij}^t + c_{ij}^t = s_{ij}^t \tag{8}$$

where $s_{ij}^t \in \{0,1\}$.

If any of the beam elements changes to either the covered or uncovered position, the global variable g^t must assume the value 1 according to

$$\sum_i \sum_j s_{ij}^t \leqslant IJg^t \tag{9}$$

where $g^t \in \{0,1\}$.

The objective of the sequencing problem given the constraint of not exceeding the minimum required number of monitor units is to complete the intensity map with the fewest possible segments:

$$\min \sum_t^{\tilde{z}} g^t = g. \tag{10}$$

The problem of minimizing objective (10) subject to conditions (1)–(5), (6′), and (7)–(9) describes an expanded mixed integer program which can again be solved using regular methods such as branch and bound. Solutions in this study are obtained using a commercial package [CPLEX 6.5 Users Manual, Incline Village, NV (1999)] that implements the technique of branch and bound with reference to a standard text.[11] Inequalities are preprocessed to reduce the number of constraints or variables in the problem set. The branch and bound method solves a succession of linear programs in which values are fixed on some of the integer variables (the branch) and the remaining integer variables are free to take on fractional values.[12] The number of free variables is successively reduced until a solution is found that is worse than one in which none of the integer variables have fractional values (the bound). All ways of completing the assignment of integer values that contain the previously fixed values on the subset of integer variables can then be discarded. A different sequence of fixed values on integer variables is formed and the process continued until all possible combinations of integer variables are implicitly enumerated. Although the number of possible combinations of values for the integer variables can be very large, implicit enumeration allows large scale problems to be solved.[13]

B. Special constraints

Special conditions can be added to describe additional restrictions on leaf movement. One condition of practical interest is that the leaves move in a single direction. Without loss of generality, the allowed direction can be specified as toward the right. The right leaf cannot cover an element that it had not covered before and the left leaf cannot uncover an element that it previously exposed. The following two conditions enforce these rules:

$$p_{ij}^t - p_{ij}^{t+1} \geqslant 0, \tag{11}$$

$$l_{ij}^{t+1} - l_{ij}^t \geqslant 0. \tag{12}$$

Other conditions that describe manufacturer or user specific constraints can be included within the integer algorithm. Two were considered in this study. One, sometimes called a collision constraint, forbids overtravel of leaves along adjacent rows. Specifically, a right leaf cannot travel further to

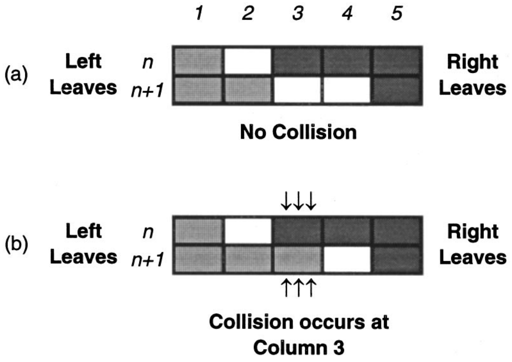


FIG. 1. Illustration of collision constraint: (a) constraint satisfied, (b) constraint violated. Arrows show the extension of the left leaf beyond the end of the right leaf on an adjacent row. In this case, condition (13) is not satisfied, for $i=n,\ j=3$

the left than the end of a left leaf along an adjacent row (Fig. 1). The collision constraint is enforced by adding the following two inequalities to the integer program:

$$l_{i+1j}^{t}+p_{ij}^{t}\leqslant 1, \tag{13}$$

$$l_{i-1j}^{t}+p_{ij}^{t}\leqslant 1. \tag{14}$$

The second condition has been proposed to reduce the tongue and groove effect, and is called here a tongue and groove constraint.[7] The condition forbids two bixels that lie in the open and closed states, respectively, in one segment and are adjacent along a column to lie in the closed and opened states, respectively, in some other segment. The state of bixel ij at time period t is described by the variable d_{ij}^{t}. The tongue and groove condition is described by the following inequality pair:

$$-1\leqslant d_{i+1j}^{t}+d_{ij}^{t'}-d_{ij}^{t}-d_{i+1j}^{t'}\leqslant 1 \quad (t\neq t'). \tag{15}$$

Implementation of the special conditions was facilitated by expressing each integer problem as a feasibility program. The minimum number of monitor units required once the collision constraint is added was found by fixing the monitor unit sum $\Sigma_t^T z^t$ in condition $(6')$ at the lowest level consistent with a feasible solution. The starting level was the number of monitor units obtained when no collision constraint was added. This level was increased one unit at a time until a feasible solution consistent with the constraints was reached. The minimum number of segments was obtained once the number of monitors units was fixed at its minimum. A series of integer programs were constructed by breaking down the intensity map into individual rows and solving the single row problems under the condition that the total number of monitor units not exceed the minimum possible for the entire map. The largest number of segments among the solutions to all these single row problems becomes a lower bound on the original problem containing all the rows together. Using this lower bound, the original problem containing the entire intensity map was solved. The total number of segments given by $\Sigma_t^{\tilde{Z}} g^t$ in Eq. (10) was first set equal to this lower bound, and then increased one segment at a time until a feasible solution was reached. The development of a lower bound reduces the number of integer programs which must be handled. At each step, all possible combinations of values for the binary variables $\{g^t\}$ whose sum equaled the current fixed bound was examined. If no feasible solution was found for these combinations, the total number of segments was increased one at a time until feasibility was reached. The method of creating more programs each with fewer constraints was used to sequence the maps for one of the clinical cases under the special condition of a leaf collision condition.

The process of sequencing the integer program into progressively smaller subprograms was extended to implement the tongue and groove condition. The minimum number of monitor units required to sequence a map without imposition of the tongue and groove constraint, $\tilde{Z}$, was found as before. This number was then partitioned into all possible sets of integer values that will sum to its total. Each set of integer values, known as a partition, corresponds to one possible way of assigning monitor units to individual leaf segments without exceeding the allowed total, $\tilde{Z}$. The number of monitor units assigned to each segment corresponds to the difference between values of the monitor unit indices, t', t'', over successive instances when a new segment is formed, which is denoted by the condition $g^t=1$ in (9). The feasibility of each of these possible assignments of monitor units to segments was then examined, beginning with partitions containing the smallest number of segments and proceeding iteratively to sets with the next highest number. Sets tied for the number of elements could be examined in any order. The feasibility of each tentative assignment of monitor units to segments was found by determining through a method analogous to branch and bound whether a set of binary values $\{d_{ij}^t\}$ existed that would satisfy the conditions described by inequalities (1)–(3) and (15) that define the allowed structure of the leaf positions, and which will deliver the intensity map defined by Eq. (4) when the segments are given their assigned number of monitor units. The limited number of leaf positions allowed by the tongue and groove condition makes it possible to examine all feasible constructions of leaf setups using the method of branch and bound. If no feasible solution is found, partitions of the allowed number of monitor units into sets containing the next highest number of elements that can be assigned to individual segments are examined. The process continues until the smallest number of segments that can satisfy the leaf sequencing requirements without using more than the allowed number of monitor units is found. If all partitions of $\tilde{Z}$ are exhausted without finding a feasible solution, then the allowed number of monitor units is incremented by one and the process continued. The described method will terminate in a finite number of steps because the values assigned to the intensity map limit the total number of monitor units that can be assigned to segments that expose at least one bixel.

The leaf sequencing algorithms were applied to an arbitrary intensity map in the literature that served as a benchmark,[9] and to clinical examples of intensity modulated treatments of prostate cancer. The arbitrary example [Fig. 2(a)] consisted of four rows and six columns, with between 0 and 5 monitor units assigned to each element. The clinical cases consisted of treatment volumes divided into transverse planes with 1.0 cm interslice intervals, treated using up to nine beam directions selected from a set of eighteen or thirty-six beams spaced at equal angular intervals. Archieved cases or published contours were used.[14] The simulated examples employed 1.0 cm leaf widths and 1.0 cm increments of leaf movement, with intensity values discretized into fifteen units. A total of 19 intensity maps were generated, including 18 derived from prostate patient contours (available as archieved material or taken from the literature). The remaining arbitrary map was taken from the literature and used as a benchmark.

Leaf sequences for the intensity maps were generated using three algorithms for comparison to the integer program results. The first was the algorithm of Bortfeld that treats each row independently.[5] It provides a solution which moves the leaves in a single direction and consistently yields the fewest monitor units in empirical tests. The second is an areal algorithm based on the work of Xia and Verhey[6] as described in Boyer and Yu.[9] The third was a formulation of the areal algorithm identified as the reducing algorithm in Xia and Verhey.[6] The areal and reducing algorithms do not treat each row independently. They permit the leaves to move in either direction at every step and reduce the number of segments at the cost of more monitor units. The areal algorithm uses monitor units at each step in the sequence equal to the largest power of two that can be accommodated in the residual intensity map.[9] Ties among elements to be grouped into a segment were resolved for these tests, as suggested by the illustrations of Boyer and Yu,[9] by choosing elements located most closely to the upper left-hand corner of the collimator. The reducing algorithm rounds off, rather than truncates, the largest power to which two can be raised without exceeding the maximum intensity left in the map, and then delivers one power of two less than this value.[6] The stated rationale for using rounded off powers of two is to drive the algorithm to deliver about half the maximum intensity level in the map.[6] The performance of the areal and reducing algorithms was improved by combining segments that appeared in the generated sequence with identical leaf patterns. By combining the segments, the total number of monitor units was not changed but the total number of segments was made smaller. The results of these three algorithms were compared to that given by the integer algorithm which minimizes the number of segments under the condition that the minimum number of monitor units are used. The integer algorithm was run in two modes, either one that is restricted to unidirectional movement [using relations (11) and (12)] or one that allows bidirectional leaf motion. The consequence of adding to the integer algorithm a leaf collision or a tongue and groove condition was also determined.

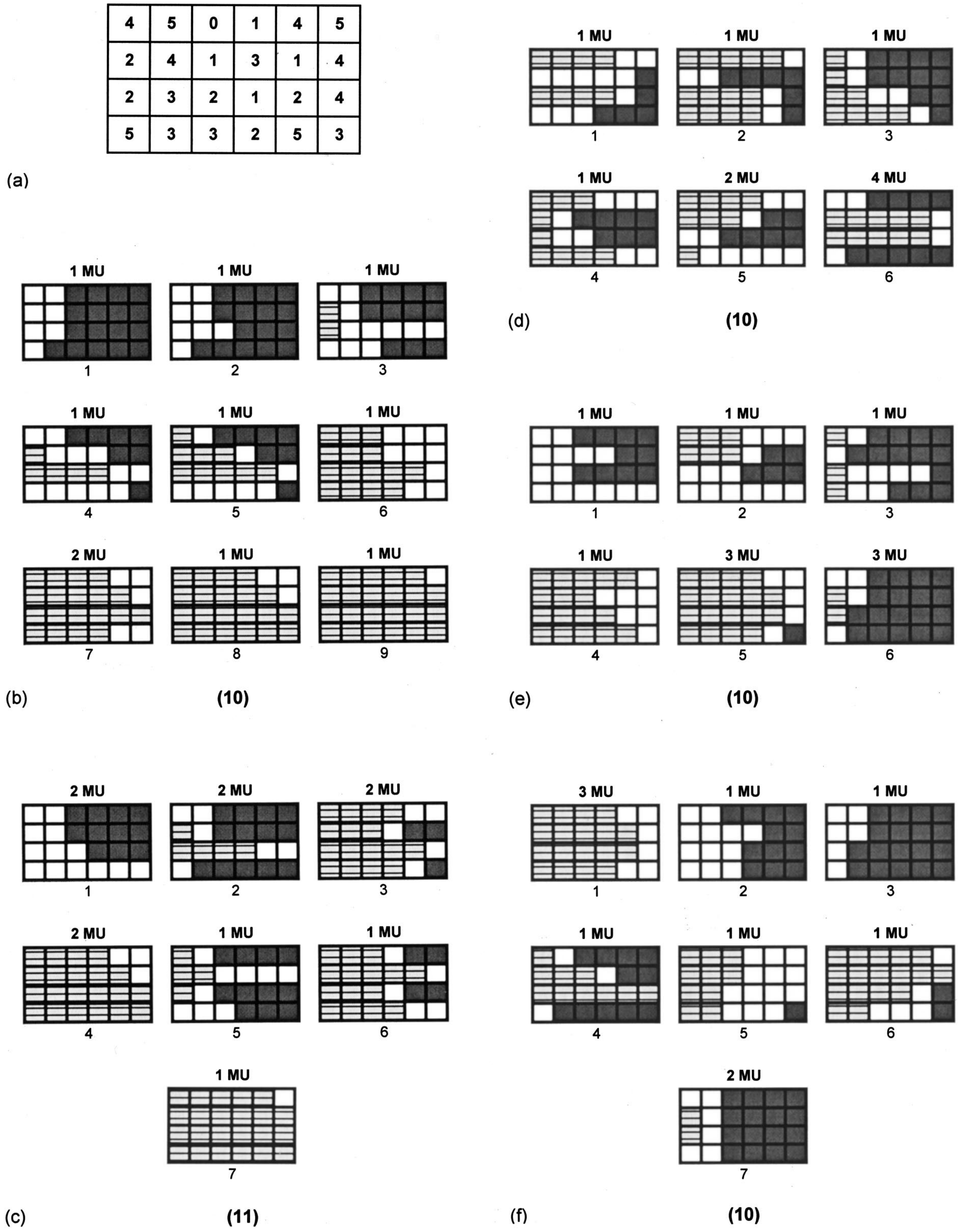
III. RESULTS

The sequences generated for the arbitrary map by the Bortfeld algorithm, the reducing algorithm, and the bidirectional integer algorithm are shown in Figs. 2(b)–2(d). The areal algorithm (not shown) returned more monitor units and segments than did the reducing algorithm for this example, as seen in Table I. A manual check reveals that the segments weighted by their monitor units sum to the desired intensity maps. Inspection of Fig. 2 shows that the integer algorithm produces the desired intensity maps using fewer segments than Bortfeld, and using fewer monitor units than the reducing algorithm.

The number of monitor units and the number of segments required by the sequencing algorithms were tabulated for all the intensity maps generated from the examples studied. Table II, columns 2–6 display the monitor units required by each of the algorithms for the intensity maps of one of the prostate case examples. Table III, columns 2–6 show the number of segments required by the algorithms for the same case. A summary of the segment and monitor unit numbers totaled over the entire treatment for the examples studied is shown in Table I. The minimum number of monitor units is achieved by the Bortfeld method, as verified by the minimizing integer algorithm. However, Table I shows that on average the Bortfeld algorithm used 32% more segments than needed if the least number of monitor units are to be used and the leaves restricted to unidirectional movement, and used 58% more segments than needed if bidirectional movement were allowed. The areal algorithm which gives bidirectional treatment reduced the number of segments from that given by Bortfeld but used approximately 48% more monitor units. The reducing algorithm also used fewer segments than did Bortfeld but required about 23% more monitor units than the minimum. The data aggregated in Table I show that this increase in monitor units was not necessary. The integer algorithm reduced the number of segments even further than did the areal and reducing algorithms, while using no more than the minimum number of monitor units. The additional monitor units used by the areal algorithm and reducing algorithms to lower the number of segments is consequently shown to be excessive.

The number of segments used by the areal or reducing algorithm was 23% higher than required by the integer algorithm (Table I). The results show excess use of both monitor units and segments by the areal and reducing algorithms. Combining the results, the areal algorithm expended 48% more monitor units and the reducing algorithm expended 23% more monitor units than did the integer algorithm, while the number of segments used by the areal or reducing algorithms was 23% higher than given by the integer algorithm.

The effect on the integer algorithm of adding a leaf collision or tongue and groove constraint is illustrated for the arbitrary map in Figs. 2(e) and 2(f). The effect of adding these constraints for all the maps in an entire clinical case is shown, respectively, in Tables II and III (columns 7 and 8) for the number of monitor units and number of segments. The addition of the tongue and groove conditions had little effect on the total number of segments or number of monitor units used, although different segments might appear in the sequence. The total number of segments increased by less



4	5	0	1	4	5
2	4	1	3	1	4
2	3	2	1	2	4
5	3	3	2	5	3

FIG. 2. Generation of an arbitrary intensity map by different leaf sequencing algorithms. (a) Arbitrary intensity map, with entries corresponding to Ref. 9. Leaves move along rows. Columns separate the positions at which the leaves can stop. (b)–(d) Leaf sequences provided by three algorithms that generate the map. (b) Bortfeld, (c) Reducing. (d) Integer (bidirectional mode). (e) Integer (bidirectional) with leaf collision constraint. (f) Integer with tongue and groove constraint. Individual segments are numbered sequentially at the bottom. The number of monitor units for each segment is shown above its figure. The total number of monitor units appears in parentheses below each identified sequence of segments.

TABLE I. The number of monitor units and segments for different leaf sequencing algorithms–aggregate results. Results are summed for the arbitrary map and two clinical examples. Treatment delivery maps generated for contours of prostate case I are from Ref. 14 and prostate case II. (MU are units proportional to monitor units Seg are Segments.)

	Unidirectional				Bidirectional					
	Bortfeld		Integer (U)		Areal		Reducing		Integer (B)	
	MU	Seg	Mu	Seg	Mu	Seg	MU	Seg	MU	Seg
Cases										
Arbitrary map	10	9	10	7	15	7	11	7	10	6
Prostate I	83	51	83	40	113	44	88	43	83	38
Prostate II	128	98	128	73	201	73	172	73	128	56

than 10% when the tongue and groove condition was added. Calculation times when the tongue and groove constraint was added amounted to 30 s for 90% of maps, with a range of up to about one half hour. The addition of a constraint on leaf collision across adjacent rows increased the total number of monitor units expended by 26%, and raised the number of segments used by 23%.

IV. DISCUSSION

Neither of the two well-described approaches for leaf sequencing produces a solution that makes efficient use of the numbers of segments or monitor units. The two approaches bracket the range of goals adopted in the literature for leaf sequencing routines. The Bortfeld algorithm expends the minimum number of monitor units but used on average about a third more segments than needed for unidirectional movement, and about 50% more segments than required if bidirectional movement is allowed. The areal and reducing algorithms, respectively, expended 48% and 23% more monitor units than the minimum required to deliver treatment, and yet used 23% more segments than needed if only the minimum number of monitor units had been used. It is, of course, possible to replace the restriction in Eq. $(6')$ that the minimum number of monitor units be used with the condition that the employed number of monitor units not exceed some absolute or proportional increase above the minimum. The possible tradeoff between the number of segments and monitor units when the number of segments is minimized for a fixed number of monitor units is the subject of ongoing investigation. The minimizing integer algorithm makes it possible to generate sequences that respect general rules for leaf movement while reaching an efficient frontier of monitor unit and segment usage.

The Bortfeld, areal, and reducing algorithms are well-described routines designed to apply to a general class of leaf sequencing problems. Other algorithms have been developed to take account of manufacturer specific restrictions on collimator movement. Two conditions were considered in this paper, leaf collision and tongue and groove constraints. Tests with the integer algorithm showed that inclusion of the collision constraint raised the required number of segments and number of monitor units by more than 20%. In contrast, incorporating the tongue and groove constraint into the algo-

TABLE II. Comparison of number of monitor units under different algorithms, prostate case I. Number of monitor units required by different leaf sequencing algorithms to generate intensity maps for a sample prostate case, listed by beam angle. Columns 2–6 compare the number of monitor units required by the integer algorithm, run in unidirectional (U) or bidirectional (B) mode, with the number of monitor units required by the Bortfeld, areal, and reducing algorithms when no tongue and groove or leaf collision constraint is present. The effect of imposing a leaf collision or tongue and groove constraint on the number of monitor units returned by the integer algorithm is shown in columns 7 and 8. Prostate case contours are from Ref. 14.

	Unidirectional		Bidirectional				
(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)
						Special Constraints	
Beam angles	Bortfeld	integer (U)	Areal	reducing	integer (B)	collision	Tongue and Groove
60	10	10	15	11	10	10	10
160	13	13	14	13	13	16	13
180	9	9	18	10	9	10	9
220	2	2	2	2	2	3	3
260	15	15	17	15	15	16	15
280	3	3	4	4	3	4	3
300	5	5	8	5	5	7	5
320	17	17	19	19	17	24	17
340	9	9	16	9	9	15	9
Total	8	83	113	88	83	105	84

TABLE III. Comparison of number of segments under different algorithms, prostate case I. Number of segments required by different leaf sequencing algorithms to generate intensity maps for a sample prostate case, listed by beam angle. Columns 2–6 compare the number of segments required by the integer algorithm, run in unidirectional (U) or bidirectional (B) mode, with the number of segments required by the Bortfeld, areal, and reducing algorithms when no tongue and groove or leaf collision constraint is present. The effect of imposing a leaf collision or tongue and groove constraint on the number of segments returned by the integer algorithm is shown in columns 7 and 8. Prostate case contours are from Ref. 14.

	Unidirectional		Bidirectional				
(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)
						Special Constraints	
Beam angles	Bortfeld	integer (U)	Areal	reducing	integer (B)	collision	Tongue and Groove
60	8	6	7	6	6	7	7
160	7	5	5	5	5	6	5
180	7	5	6	5	5	5	5
220	2	2	2	2	2	3	3
260	5	4	5	6	4	5	4
280	3	3	3	3	3	4	3
300	4	3	4	4	3	4	3
320	8	7	7	7	6	6	6
340	7	5	5	5	4	7	5
Total	51	40	44	43	38	47	41

rithm had little effect on the monitor units or segments used.

The integer algorithm can also be adjusted to reflect different goals for the sequence that might arise as the technology for delivering intensity modulated radiotherapy (IMRT) changes. Dramatic improvements in the speed of leaf setups can diminish the relative importance of keeping the number of segments low. On the other hand, the control of monitor units may assume greater significance as IMRT with ever finer leaf widths is applied to sites such as breast for which issues of scatter radiation to uninvolved tissues are important. It is possible to substitute as a goal a weighted combination of number of monitor units and number of segments by incorporating the variables that represent these quantities (viz. Z,g) into the objective function.

The shortfalls in the solution quality of available algorithms demonstrates a need for improved routines, and should encourage further work in this area. The precision and flexibility of the integer programming approach itself offers one avenue for finding improved solutions. A drawback is that solution speed is sensitive to the structure and scale of the problem to be solved. The integer algorithm solved the examples shown in this report. Other problems may not be amenable to solution. With problems of denser scale that would arise from finer leaf widths, smaller increments of leaf movement, or more gradations in the monitor units, it is possible that solution speed would deteriorate to make results unattainable in a practical time. As is true of most real world problems to which integer programs are applied, special tailoring of the algorithm to exploit characteristics of the problem structure may allow significant improvement in speed and render the approach practical for future needs in leaf sequencing design.

At present, the integer program routine offers a practical method to study the performance of leaf sequencing algorithms applied to problems that arise in clinical practice. Shortfalls from optimality produced by available algorithms can be identified, so that it becomes possible to measure the excess in the number of monitor units or number of segments that are used.

V. CONCLUSION

An integer algorithm was developed to test the performance of available leaf sequencing routines. The algorithm can accommodate device specific restrictions on leaf movement, such as a collision constraint or a tongue and groove condition. As implemented in this report, the algorithm returns a leaf sequence that minimizes the number of segments given the constraint that the minimum number of monitor units not be exceeded. The integer model can be generalized to include other goals that have been considered for leaf sequencing routines. It can minimize a weighted combination of the numbers of monitor units and segments, or minimize the number of segments for different settings of the allowed number of monitor units. The model can be extended to consider more complicated expressions of leaf movement that may track wear and tear or overall delivery time. This report provides a method to minimize one expression of delivery time, the sum across all segments of the larger of either the maximum time needed to move any leaf between segments or some basic setup time. The performance of two well-described approaches to the design of leaf sequencing algorithms was studied without imposing any special conditions on the leaf movement. The Bortfeld algorithm matched the minimum number of monitor units given by the integer algorithm, but used on average 30% more segments than the integer algorithm required. The alternative approach of the areal and reducing algorithms is to accept fewer segments in exchange for more monitor units. The tests showed that the areal or reducing algorithms used on average at least 23% more monitor units than the minimum, and yet required 23% more segments than needed if a solution using only the minimum number of monitor units had been used. The improved solution was obtained using the integer algorithm. The shortfalls of the other algorithms were found without imposing machine specific constraints, such as tongue and groove or collision conditions. Other studies have been interpreted as showing no change in the relative performance of sequencing algorithms in the presence of a collision constraint. Because the integer algorithm has been implemented to accept a collision or tongue and groove constraint, it is possible to use it to evaluate a wide variety of algorithms that accept these special conditions or which adopt other goals, such as minimization of leaf setup time. The integer program method introduced here offers an approach to better algorithm design, and is a practical tool for measuring solution quality.

[a] Electronic mail: mlanger@radonc.uh.iupui.edu

[1] S. Webb, ''Configuration options for intensity-modulated radiation therapy using multiple static fields shaped by a multileaf collimator,'' Phys. Med. Biol. **43**, 241–260 (1998).

[2] M.-A. Keller-Reichenbecher, T. Bortfeld, S. Levegrun, J. Stein, K. Preiser, and W. Schlegel, ''Intensity modulation with the 'step and shoot' technique using a commercial MLC: A planning study,'' Int. J. Radiat. Oncol., Biol., Phys. **45**, 1315–1324 (1999).

[3] P. M. Evans, V. N. Hansen, and W. Swindell, ''The optimum intensities for multiple static multileaf collimator field compensation,'' Med. Phys. **24**, 1147–1156 (1997).

[4] J. M. Galvin, X.-G. Chen, and R. M. Smith, ''Combining multileaf fields to modulate fluence distributions,'' Int. J. Radiat. Oncol., Biol., Phys. **27**, 607–705 (1993).

[5] T. R. Bortfeld, D. L. Kahler, T. J. Waldron, and A. L. Boyer, ''X-ray field compensation with multileaf collimators,'' Int. J. Radiat. Oncol., Biol., Phys. **28**, 723–730 (1994).

[6] P. Xia and L. J. Verhey, ''Multileaf collimator leaf sequencing algorithm for intensity modulated beams with multiple static segments,'' Med. Phys. **25**, 1424–1434 (1998).

[7] R. A. Siochi, ''Minimizing static intensity modulation delivery time using an intensity solid paradigm,'' Int. J. Radiat. Oncol., Biol., Phys. **43**, 671–680 (1999).

[8] W. Que, ''Comparison of algorithms for multileaf collimator field segmentation,'' Med. Phys. **26**, 2390–2396 (1999).

[9] A. L. Boyer and C. Y. Yu, ''Intensity-modulated radiation therapy with dynamic multileaf collimators,'' Semin Radiat. Oncol. **9**, 48–59 (1999).

[10] J.-R. Dai and Y.-M. Hu, ''Intensity-modulation radiotherapy using independent collimators: An algorithm study,'' Med. Phys. **26**, 2562–2570 (1999).

[11] G. L. Nemhauser and L. A. Wolsey, *Integer and Combinatorial Optimization* (Wiley, New York, 1988).

[12] A. H. Land and A. G. Doig, ''An automatic method of solving discrete programming problems,'' Econometrica **28**, 497–320 (1960).

[13] H. Crowder, E. L. Johnson, and M. Padberg, ''Solving large-scale zero-one linear programming problems,'' Oper. Res. **31**, 803–833 (1983).

[14] T. Bortfeld, A. Boyer, W. Schlegel, D. Kahler, and T. Waldron, ''Realization and verification of three-dimensional conformal radiotherapy with modulated fields,'' Int. J. Radiat. Oncol., Biol., Phys. **30**, 899–908 (1994).