Inf Process Lett. Author manuscript; available in PMC 2008 November 25.

Published in final edited form as:
Inf Process Lett. 2008 January 31; 106(5): 188–194. doi:10.1016/j.ipl.2007.11.008.

Efficient Intensity Map Splitting Algorithms for Intensity-Modulated Radiation Therapy*

Xiaodong Wu
Department of Electrical and Computer Engineering, Department of Radiation Oncology, The University of Iowa, Iowa City, IA 52242, USA, xiaodong-wu@uiowa.edu

Abstract

In this paper, we study several interesting *intensity map splitting (IMSp)* problems that arise in Intensity-Modulated Radiation Therapy (IMRT), a state-of-the-art radiation therapy technique for cancer treatments. In current clinical practice, a multi-leaf collimator (MLC) with a *maximum leaf spread* is used to deliver the prescribed intensity maps (IMs). However, the maximum leaf spread of an MLC may require that a large intensity map be split into several abutting sub-IMs each being delivered separately, which results in prolonged treatment time. Few IM splitting techniques reported in the literature have addressed the issue of treatment delivery efficiency for large IMs. We develop a unified approach for solving the IMSp problems while minimizing the total beam-on time in various settings. Our basic idea is to formulate the IMSp problem as computing a *k*-link shortest path in a directed acyclic graph. We carefully characterize the intrinsic structures of the graph, yielding efficient algorithms for the IMSp problems.

Keywords

Intensity map splitting; *k*-link shortest paths; Algorithms; IMRT; Computational Medicine

1 Introduction

The *intensity map splitting* (IMSp) problems that we study in this paper arise in *Intensity-Modulated Radiation Therapy* (IMRT) [24], a state-of-the-art radiation therapy technique for cancer treatments. IMRT aims to deliver a highly conformal radiation dose to a target tumor while sparing the surrounding normal tissues. The quality of IMRT crucially depends on the ability to accurately and efficiently deliver the prescribed dose distributions of radiation, commonly called *intensity maps* (IMs). An intensity map is specified by a set of nonnegative integers on a 2-D grid (see Figure 1(a)). The number in a grid cell indicates the amount (in unit) of radiation to be delivered to the body region corresponding to that cell.

One of the most advanced tools today for delivering IMs is the *multileaf collimator* (MLC) [24]. An MLC has multiple pairs of tungsten leaves of the same rectangular shape and size (see Figure 1(b)). The two opposite leaves of each pair are aligned to each other. The leaves can

*This research was supported in part by an NIH-NIBIB research grant R01-EB004640, and in part by a seed grant award from the American Cancer Society through an Institutional Research Grant to the Holden Comprehensive Cancer Center, the University of Iowa, Iowa City, Iowa, USA. A preliminary version of this paper was presented at the 11th International Computing and Combinatorics Conference (COCOON), August 2005.

move left and right to form (say) an y-monotone rectilinear region (i.e., monotone to the y-axis), called an *MLC-aperture*. The cross-section of a cylindrical radiation beam (generated by a radiotherapy machine) is shaped by this MLC-aperture to deliver certain units of radiation to (a portion of) an IM. We associate that MLC-aperture with an integer representing the amount of radiation delivered by it. The mechanical constraints of the MLC limit what kinds of aperture shapes are allowed to be used [24]. One such constraint is called the *no-interleaf collision*: The distance between two opposite leaves of the neighboring pairs must be ≥ a given separation value δ (e.g., $\delta = 1cm$). For example, the Elekta MLC is subject to the no-interleaf collision constraint, while the Varian MLC allows interleaf collision. Another common constraint is called the *maximum leaf spread*: The two leaves of the same pair have an opening ≤ a given threshold Δ (e.g., $\Delta = 14.5cm$). Geometrically, the maximum leaf spread means the rectilinear y-monotone polygon corresponding to each MLC-aperture has a maximum horizontal "width" ≤ Δ.

One of the most popular IMRT approaches for delivering IMs using an MLC is the "step-and-shoot" technique [7,24,25]. Mathematically, the "step-and-shoot" delivery planning can be viewed as the following segmentation problem: Given an intensity map A defined on a 2-D $m \times n$ grid, decompose A into the form of $A = \sum_k^{\kappa} \alpha_k S_k$, where S_k is a special 0-1 matrix specifying an MLC-aperture, α_k is the amount of radiation delivered through S_k (called the *height* of S_k), and κ is the number of MLC-apertures used to deliver A (see Figure 2). (The reader is referred to [25,2,9,5] for more details on the step-and-shoot IMRT technique.) Intuitively, one may view the SLS problem as playing the following game: An IM is a 3-D "mountain" made of unit cubes (Figure 1(a)). Delivering an MLC-aperture can be viewed as creating a plateau whose height is the amount of radiation delivered. Then, one likes to "build" this mountain by stacking up the minimum number of plateaus with minimum total height (Figure 2). It is important to note that each plateau, like the 3-D IM mountain, is not one whole "rigid" object, i.e., it is made of unit cubes as well.

There are two major measures for the quality of the step-and-shoot delivery: (1) the *beam-on time* which is given by $\sum_{k=1}^{\kappa} \alpha_k$, and (2) the number κ of the MLC-apertures used in the delivery plan. The beam-on time is the actual time that the patient is exposed under the radiation beams. Minimizing beam-on time is crucial to reduce the patient's risk under irradiation and to reduce the delivery error caused by tumor motion [2]. On the other hand, minimizing the number of MLC-apertures (hence, reducing the treatment time of each IM) is also important because it not only lowers the treatment cost for each patient but also enables hospitals to treat more patients [5].

In current clinical radiation therapy, large intensity maps frequently occur [10,8,20]. Due to the maximum leaf spread constraint of the MLC design, a large IM needs to be split into several abutting sub-IMs each being delivered separately using the step-and-shoot delivery technique. This results in prolonged treatment times (longer beam-on times and more MLC-apertures). Although the step-and-shoot delivery has received a great deal of attention from several research communities, such as medical physics [3,7,18,19,22,23,25], computer science [4,5,12,13], and operations research [1,2,9], few IM splitting techniques reported in the literature has addressed the issue of treatment delivery efficiency for large IMs. To our best knowledge, Kamath *et al.* [14] first gave a quadratic time algorithm to split a large IM into *at most three* sub-IMs (thus restricting the maximum size of a large IM) while minimizing the total beam-on time. They used straight lines to do the splitting, yielding abutting sub-IMs. One drawback associated with that field splitting method is the field mismatching problem that occurs in the field junction region due to the uncertainties in patient setup and organ motion [20,15]. If the borders of two abutting sub-IMs do not precisely align each other, it may result in *hotspots* or *codespots*. To alleviate the field mismatching problem, Kamath *et al.* recently studied the field splitting problem in which the overlapping of the resulting sub-IMs was allowed [15]. However, their algorithm again works only for the case that the width of the IM is $\leq 3\Delta$. Chen and Wang [6] developed an $O(mn + m\Delta^{d-2})$ time algorithm for optimally splitting an IM of size $m \times n$ allowing overlapping of the sub-IMs, where d is the number of resulting sub-IMs. In their algorithm, they optimized the total minimum beam-on time of the resulting sub-IMs and assumed that the MLC allows leaf collision. More recently, Wu *et al.* [21] gave a linear time algorithm for field splitting allowing overlapping, while minimizing the total complexity of the resulting sub-IMs.

In this paper, we study the following **intensity map splitting (IMSp)** problem: Given an $m \times n$ IM A and an integral maximum leaf spread $\Delta > 0$, split A into a set of abutting sub-IMs $\{A_i, A_2, \ldots, A_{\lceil \frac{n}{\Delta} \rceil}\}$ each A_i with size of $m \times n_i$ $(i=1,2,\ldots,\kappa=\lceil \frac{n}{\Delta} \rceil)$, such that: (1) $ni \leq \Delta$,(2) $\sum_{i=1}^{\kappa} n_i = n$; and (3) the total beam-on time for delivering those sub-IMs is minimized. Note that no overlapping of the resulting sub-IMs is allowed in this IMSp problem and $\kappa = \lceil \frac{n}{\Delta} \rceil$ is the minimum number of sub-IMs needed for delivering A subject to the maximum leaf spread constraint. We call such a set of κ sub-IMs a *κ-splitting* of A. More sub-IMs may be used to achieve less total beam-on time to delivery IM A. But, that could significantly increase the number of MLC-apertures (i.e., the total treatment time), which is undesirable. It is also worth of noting that no more than 3 sub-IMs seem required in current medical practice (i.e., $\kappa \leq 3$), but we believe that the IMSp problem is of its own interest and may find other applications.

We develop a unified approach for solving the IMSp problem in various settings. In our solution, the IMSp problem is formulated as computing a κ-link shortest path in a directed acyclic graph (DAG) [17] transformed from the given IM. By judiciously characterizing the intrinsic structures, we compute the κ-link shortest path without explicitly constructing the graph and integrate the computation of the beam-on times for sub-IMs into our κ-link shortest path computation, which yields an improvement of the running time by at least an order of magnitude. Our main results in this paper are summarized as follows.

- an optimal $O(n)$ time algorithm for solving the IMSp problem with $m = 1$ (i.e., using only *one* MLC leaf pair for the IM delivery).
- an $O(mn\Delta)$ time algorithm for solving the IMSp problem in which the interleaf collisions are allowed.
- an $O(m^2 n\Delta)$ time algorithm for solving the IMSp problem subject to the no-interleaf collision constraint.

2 Reformulation of the IMSp problem

Given an instance of the IMSp problem, an $m \times n$ IM A and an integer $\Delta > 0$ (the maximum leaf spread), we define a weighted directed acyclic graph $G = (V, E)$ for A, as follows.

The vertices of G are defined as $V = \{s, t\} \cup \{v_j : 0 \leq j < n\}$. Each column $A[*, j] = \{A[0, j], A[1, j], \ldots, A[m-1, j]\}$ $(j = 0, 1, \ldots, n-1)$ of A corresponds to exact one vertex v_j in G. For every $j \in \{0, 1, \ldots, n-2\}$, vertex v_j has a directed edge in E to each vertex in $\{v_k : j < k \leq \min\{j + \Delta, n-1\}\}$ (i.e., v_j connects to its following Δ vertices). Note that every edge $e = (v_j, v_{j'})$ $\in E$ $(j < j')$ is associated with a sub-IM of A, denoted by $A[*, j+1 .. j']$, which consists of all rows of A from column $j+1$ to column j'. Let $T_{bot}(A')$ denote the minimum beam-on time for delivering the IM A'. The weight $w(e)$ of the edge $e = (v_j, v_{j'})$ is $T_{bot}(A[*, j+1 .. j'])$, which can be computed using algorithms in [9,11]. From vertex s, we introduce a directed edge $(s, v_j) \in E$ to every vertex v_j for $0 \leq j < \Delta$ and the weight of the edge equals to $T_{bot}(A[*, 0 .. j])$. Meanwhile, each vertex v_j with $n - \Delta = j < n - 1$ has a directed edge to vertex t whose weight is the minimum beam-on time for delivering the sub-IM $A[*, j+1 .. n-1]$.

Obviously, $G = (V, E)$ thus constructed from A is a weighted directed acyclic graph (DAG). We next show that a κ-link shortest path ($\kappa=\lceil \frac{n}{\Delta} \rceil$) from s to t in G specifies an optimal κ-splitting of A. Note that each κ-link s-to-t path p in G is of the form of $s \to v_{j0} \to v_{j1} \to \ldots \to v_{j\kappa-2} \to t$; further, p defines a set of κ abutting sub-IMs $\mathcal{A}=\{A[*,0..j_0],A[*,j_0+1..j_1],\ldots,A[*,j_{\kappa-2}+1..n-1]\}$ used for delivering the IM A. The total minimum beam-on time of $\mathcal{A}$ (i.e., $T_{bot}(\mathcal{A})=\sum_{M\in\mathcal{A}} T_{bot}(M)$) equals to the total edge weight of p. Hence, we have the following lemma.

Lemma 1

A κ-link shortest path from s to t in G specifies an optimal κ-splitting $\mathcal{A}$ of the given IM A.

The DAG G has $O(n)$ vertices and $O(\Delta \cdot n)$ edges. It takes $O(\kappa \cdot \Delta\, n) = O(n^2)$ time to compute in G a κ-link shortest path [17], after G is constructed from IM A. The total time complexity for the construction of G depends on the computations of edge weights in G, while the weight of each edge equals the minimum beam-on time of the associated sub-IM of A. Thus, the construction of G involves in computing the minimum beam-on times [9,11] for $O(\Delta \cdot n)$ sub-IMs of A, which may become the bottleneck of our IMSp algorithm. Let $T(G)$ denote the total time complexity for constructing G from A. The IMSp problem is solvable in $O(n^2 + T(G))$ time.

The κ-link s-to-t path is a straightforward model for solving the IMSp problem. Next, we further exploit the intrinsic structures of the IMSp problem to simplify G and give a new dynamic programming approach.

Let $\mu = n \bmod \Delta$ and if $\mu = 0$, set $\mu = \Delta$. Then, we partition the column index set $\{0, 1,\ldots, n-1\}$ of A into the following consecutive segments: $C_k = \{j : k \cdot \Delta + \mu - 1 \le j < \min\{n, (k+1) \cdot \Delta\}\}$ and $U_k = \{j : k \cdot \Delta \le j < k \cdot \Delta + \mu - 1\}$ for $k = 0, 1,\ldots, \kappa - 2$; and $C_{\kappa-1} = \{n-1\}$ and $U_{\kappa-1} = \{j : (\kappa-1) \cdot \Delta \le j < n-1\}$ (note that U_k 's are empty when $\mu = 1$). Figure 2(b) illustrates the partition for the sample IM in Figure 2(a) with $\Delta = 5$. Let $\mathcal{A}=\{A[*,0..j_0],A[*,j_0+1..j_1],\ldots,A[*,j_{\kappa-2}+1..n-1]\}$ be a feasible κ-splitting of A, simply denoted by a set $\mathcal{A}=\{j_0,j_1,\ldots,j_{\kappa-1}\}$ with $j_{\kappa-1} = n - 1$. The following lemmas establish the connection between a feasible κ-splitting $\mathcal{A}$ of A and the segments $\{C_k : 0 \le k < \kappa\}$.

Lemma 2

For any feasible κ-splitting $\mathcal{A}=\{j_0,j_1,\ldots,j_{\kappa-1}\}$ of A, $\mathcal{A} \cap \left(\cup_{k=0}^{\kappa-1} U_k\right)=\varnothing$.

Proof—If U_k's are empty, obviously, $\mathcal{A} \cap \left(\cup_{k=0}^{\kappa-1} U_k\right)=\varnothing$. Otherwise, we prove this lemma by contradiction.

Assume that there exists a feasible κ-splitting $\mathcal{A}$ and $q \in \mathcal{A}$ such that $q \in U_r$ $(0 \le r < \kappa)$. Thus, $q \ge r \cdot \Delta$. Hence, the sub-IM $A[*, 0 .. q]$ needs to be split into at least $(r+1)$ sub-IMs subject to the maximum leaf spread constraint, while $A[*, q+1 .. n-1]$ needs at least $(\kappa - r)$ sub-IMs. Therefore, the total number of sub-IMs needed is at least $\kappa + 1$, which is a contradiction.

Based on Lemma 2, any $j \in \mathcal{A}$ is in $\cup_{k=0}^{\kappa-1} C_k$. Let us investigate the distribution of $\mathcal{A}=\{j_0,j_1,\ldots,j_{\kappa-1}\}$ in the segments C_k's. When $\kappa = 2$, note that $j_1 \in \mathcal{A}$ equals to $n - 1$ (i.e., $j_1 \in C_1 = \{n-1\}$), it is then clear that $j_0 \in C_0$. Next, we consider $\kappa > 2$ and claim that for each $k = 0, 1,\ldots, \kappa - 1$, $j_k \in C_k$. We assume otherwise, that is, for any $k = 0, 1,\ldots, \kappa - 1$, $|C_k \cap \mathcal{A}|$ either equals to 0 or ≥ 2. Notice that if there exists C_r that $|C_r \cap \mathcal{A}| \ge 2$, then there must have an C_q such that $C_q \cap \mathcal{A}=\varnothing$ since $|\mathcal{A}|=\kappa$. Thus, $|C_q \cap \mathcal{A}|=0$. Then, there is a sub-IM $A[*, j' .. j'']$

in $\mathcal{A}$ such that $j' < q \cdot \Delta + \mu - 1$ and $j'' \geq (q + 1) \cdot \Delta$, which indicates that $j'' - j' > \Delta$. Hence, $\mathcal{A}$ is not a feasible κ-splitting of A, a contradiction. Thus, the following lemma holds.

Lemma 3

In any feasible κ-splitting $\mathcal{A}=\{j_0,j_1,\ldots,j_{\kappa-1}\}$ of A, $j_k \in C_k$ for $k = 0, 1,\ldots, \kappa - 1$.

Lemma 3 leads to a simple dynamic programming approach for solving the IMSp problem, as follows. Let $S(v_j)$ denote the minimum total beam-on time for delivering the sub-IM $A[*, 0 .. j]$ subject to the maximum leaf spread constraint. Note that if $j \in C_k$ ($k = 1, 2,\ldots, \kappa - 2$), then any j' with $j - \Delta \leq j' < k \cdot \Delta$ is in C_{k-1}. Thus, we have

$$S(v_j)=\begin{cases} w(s,v_j) & \text{if } j \in C_0 \\ \min\{S(v_{j'})+w(v_{j'+1},v_j): j-\Delta \leq j'<k\cdot\Delta\} & \text{if } j \in C_k (1 \leq k<\kappa) \end{cases} \tag{1}$$

Hence, $S(v_{n-1})$ is the minimum total beam-on time for delivering A, which defines an optimal κ-splitting $\mathcal{A}^*$. Based on Lemma 3, we can remove all the vertices v_j 's with $j \in \cup_{i=0}^{\kappa-1} U_i$ from G, thus simplifying the construction of G. Figure 2(b) shows an example graph G thus constructed from the IM in Figure 2(a) with $\Delta = 5$. We still use G to denote the resulting graph. The running time of this dynamic programming scheme is clearly $O(\Delta \cdot n)$ in the worst case after constructing the graph G, which takes $T(G)$ time.

Lemma 4

The IMSp problem can be solved in $O(\Delta \cdot n + T(G))$ time.

Recall that the construction of G involves in computing the minimum beam-on times for $O(\Delta \cdot n)$ sub-IMs of A, which may become the bottleneck of our IMSp algorithm. In the following sections, we integrate the computation of the beam-on time [9,11] into our dynamic programming scheme to further improve our algorithm for the IMSp solving problem in various settings.

3 The IMSp problem with *m* = 1

This section presents our optimal $O(n)$ time algorithm for computing an optimal κ-splitting of a given IM A with *only* one row of intensities (i.e., $m = 1$). For example, this case occurs when using one pair of MLC leaves for delivering the intensity map. Thus, we assume that the input IM A is a vector, i.e., $A = (A[0], A[1],\ldots, A[n - 1])$. Kamath *et al.* [14] gave an $O(n^2)$ algorithm for this case while $n \leq 3\Delta$. We consider in this section an arbitrary n and Δ.

The minimum beam-on time $T_{bot}(B)$ of an IM B ($m = 1$) can be computed [9], as follows:

$$T_{bot}(B)=B[0]+\sum_{j=1}^{n-1}\max\{0,B[j]-B[j-1]\}.$$

Let $\mathcal{A}=\{j_0, j_1,\ldots, j_{\kappa-1}\}$ with $j_{\kappa-1} = n - 1$ be a feasible κ-splitting of the given IM A. Then,

$$T_{bot}(\mathcal{A})=\left(A[0]+\sum_{j=1}^{j_0}\max\{0,A[j]-A[j-1]\}\right) +\sum_{k=1}^{\kappa-1}\left(A[j_{k-1}+1]+\sum_{j=j_{k-1}+2}^{j_k}\max\{0,A[j]-A[j-1]\}\right)$$

Note that for any $0 < j < n$, $A[j] = \max\{0, A[j] - A[j - 1]\} + \min\{A[j - 1], A[j]\}$. Thus,

$$T_{bot}(\mathcal{A})=T_{bot}(A)+\sum_{k=0}^{\kappa-2}\min\{A[j_k+1],A[j_k]\}.$$

We call $\sum_{k=0}^{\kappa-2}\min\{A[j_k+1],A[j_k]\}$ the *bot*-increase of a feasible κ-splitting $\mathcal{A}$ of A. Hence, the following lemma, which generalizes the result in [14], follows.

Lemma 5

A feasible κ-splitting $A=\{j_0,j_1,\ldots,j_{\kappa-2},n-1\}$ that minimizes the bot-increase among all possible feasible κ-splitting of A gives an optimal solution to the IMSp problem.

Based on Lemmas 3 and 5, the following dynamic programming scheme is used to compute an optimal κ-splitting of A. Note that $C_{\kappa-1}$ has only one element v_{n-1}. For the convenience of computation, we assume that $A[n] = 0$. Let $S(j)$ be the minimum *bot*-increase among all possible feasible κ-splitting of $A[0 .. j]$. Then,

$$S(j)=\begin{cases}\min\{A[j],A[j+1]\} & \text{if } j\in C_0\\ \min\{S(j'):j-\Delta\le j'<k\cdot\Delta\}+\min\{A[j],A[j+1]\} & \text{if } j\in C_k(1\le k<\kappa)\end{cases} \quad (2)$$

Thus, for each $j \in C_k$ $(1 \le k < \kappa)$, we need to compute the minimum of $S(j')$'s for $j - \Delta \le j' < k \cdot \Delta$. A straightforward way takes $O(\Delta)$ time and the total running time is $O(\Delta \cdot n)$. However, we can do better to achieve an optimal linear time algorithm. Note that all $S(j')$'s for $j' \in C_{k-1}$ are static (i.e., do not change their values during the computation). We thus can speed up the computation of $\min\{S(j') : j - \Delta \le j' < k \cdot \Delta\}$ by using a simplified range-minima data structure [16] for $\{S(j') : j' \in C_{k-1}\}$. In our algorithm, we use an additional array L of size $O(\Delta)$ to keep the minimum of $\{S(i), S(i + 1),\ldots, S(k \cdot \Delta - 1)\}$ for each $i = [(k - 1) + \mu - 1], [(k - 1) + \mu - 1] + 1,\ldots, k \cdot \Delta - 1$ (i.e., $i \in C_{k-1}$). It is easy to see that array L can be computed in $O(\Delta)$. Hence, for each $j \in C_k$ $(1 \le k < \kappa)$, $\min\{S(j') : j - \Delta \le j' < k \cdot \Delta\} + \min\{A[j], A[j + 1]\}$ can be obtained in $O(1)$ time by using this array L. Therefore, $S(n - 1)$ can be computed in $O(n)$ time.

Theorem 1

The IMSp problem with $m = 1$ is solvable in an optimal $O(n)$ time.

4 The IMSp problem allowing the interleaf collisions

In this section, we give our $O(mn\Delta)$ time algorithm for solving the IMSp problem in which the interleaf collisions are allowed.

As shown in Section 2, the graph G constructed from the given IM A has $O(\Delta \cdot n)$ edges. For each edge in G, we need to compute the minimum bean-on time for its associated sub-IM of size $O(m \times \Delta)$. The minimum beam-on time $T_{bot}(B)$ of an $m' \times n'$ IM B can be computed as shown in [9]:

$$T_{bot}(B)=\max_{i\in\{0,1,\ldots,m'-1\}}\left\{B[i,0]+\sum_{j=1}^{n'-1}\max\{0,B[i,j]-B[i,j-1]\}\right\},$$

which takes $O(m'n')$ time. Thus, the time $T(G)$ for constructing the graph G is $O(mn\Delta^2)$. Hence, our algorithm in Section 2 for solving the IMSp problem allowing the interleaf collisions runs in $O(mn\Delta^2)$ time.

However, by carefully characterizing the computation of the minimum beam-on time, we are able to improve the algorithm by at least $O(\Delta)$. The value of $(B[i,0]+\sum_{j=1}^{n'-1} \max\{0,B[i,j]-B[i,j-1]\})$ is the minimum beam-on time of the i-th row $B[i, *]$ of B. Thus, $T_{bot}(B) = \max_{i=0,1,\cdots,m'-1} T_{bot}(B[i, *])$. Then, consider a sub-IM $A[*, j'.. j]$ of A ($j > j' > 0$). The minimum beam-on time $T_{bot}(A[*, j'.. j]) = \max_{i=0,1,\cdots,m-1} T_{bot}(A[i, j'.. j])$. Note that,

$$T_{bot}(A[i,j'..j])=T_{bot}(A[i,0..j])-T_{bot}(A[i,0..j'-1])+\min\{A[i,j'-1],A[i,j']\}.$$

Hence, for every row i of A, we compute $T_{bot}(i, 0 .. j)$ for each $j \in \{0, 1,\ldots, n-1\}$, which totally takes $O(n)$ time. Then, $T_{bot}(A[i, j'.. j])$ can be computed in $O(1)$ time. Thus, $S(v_{n-1})$, the minimum total beam-on time for delivering A subject to the maximum leaf spread constraint, can be computed in $O(mn\Delta)$ time based on the recursive Equation (1).

Theorem 2

The IMSp problem allowing interleaf collisions is solvable in $O(mn\Delta)$ time for a given $m \times n$ IM A and the maximum leaf spread Δ.

5 The IMSp problem subject to the no-interleaf collision constraint

This section presents our $O(m^2n\Delta)$ time algorithm for solving the IMSp problem, in which an MLC subject to the no-interleaf collision constraint is used for delivering the given $m \times n$ IM A.

Recall that $S(v_q)$ denotes the minimum total beam-on time for delivering the sub-IM $A[*, 0 .. q]$ subject to the maximum leaf spread constraint. From Equation (1),

$$S(v_q)=\min\{S(v_j)+w(v_{j+1},v_q):q-\Delta\le j<k\cdot\Delta\},$$

if $q \in C_k$ ($k = 1, 2,\ldots, \kappa - 1$). For a given $q \in C_k$, instead of computing $w(v_{j+1}, v_q)$ for each possible j (i.e., $q - \Delta \le j < k \cdot \Delta$) first, we compute all $w(v_{j+1}, v_q)$'s in one shot by formulating it as a single-source longest path problem in a weighted directed acyclic graph. Then, clearly $S(v_q)$ can be computed in $O(\Delta)$ time.

The DAG $H_q = (V_q, E_q)$ for computing all $w(v_{j+1}, v_q)$'s is constructed in the following way. The vertices of H_q are defined as $V_q = \{s\} \cup \{c_j: q - \Delta \le j < k \cdot \Delta\} \cup \{u_{i,j} : 0 \le i < m, q - \Delta < j \le q\}$ (note that $q \in C_k$). The edge set E_q contains three different types of edges, E_s, E_b, and E_c, that is, $E_q = E_s \cup E_b \cup E_c$, where

$$\begin{aligned}&E_s=\{(s,u_{i,q}):0\le i<m\},\\&E_b=\{(u_{i,j},u_{i',j-1}):0\le i,i'<m,q-\Delta+1<j\le q\},\\&E_c=\{(u_{i,j},c_{j-1}):0\le i<m,q-\Delta\le j<k\cdot\Delta\}.\end{aligned}$$

Next, we define the *cost* $w'(e)$ for each edge $e \in E_q$.

$$w'(e)=\begin{cases}A[i,q] & \text{if } e=(s,u_{i,q})\in E_s,\\ \max\{0,A[i',j-1]-A[i',j]\}-\sum_{p=i}^{i'-1}A[p,j] & \text{if } e=(u_{i,j},u_{i',j-1})\in E_b \text{ and } i\le i',\\ \max\{0,A[i',j-1]-A[i',j]\}-\sum_{p=i'+1}^{i}A[p,j] & \text{if } e=(u_{i,j},u_{i',j-1})\in E_b \text{ and } i>i',\\ 0 & \text{if } e=(u_{i,j},c_{j-1})\in E_c.\end{cases} \tag{3}$$

Figure 5 shows an example graph H_q for the IM in Figure 2(a) with $q = 7$. Using similar techniques in [11], we are able to prove that the total cost of the longest path from vertex s to

c_j ($q - \Delta \le j < k \cdot \Delta$) equals to the minimum beam-on time of the sub-IM $A[*, j + 1 .. q]$ (i.e., $w(v_{j+1}, v_q)$). Note that H_q is a DAG with $O(m \cdot \Delta)$ and $O(m^2 \cdot \Delta)$ edges. Thus, it takes $O(m^2 \cdot \Delta)$ time for computing a shortest s-to-t path in H_q. Note that in the worst case, we need to compute $S(v_q)$ for every $q = 0, 1,\ldots, n - 1$. Hence, Theorem 3 follows.

Theorem 3

The IMSp problem subject to the no-interleaf collision constraint can be solved in $O(m^2 n\Delta)$ time.

References

1. Ahuja RK, Hamacher HW. A Network Flow Algorithm to Minimize Beam-on Time for Unconstrained Multileaf Collimator Problems in Cancer Radiation Therapy. Networks 2004;43(4):226–240.
2. Boland N, Hamacher HW, Lenzen F. Minimizing Beam-on Time in Cancer Radiation Treatment Using Multileaf Collimators. Networks 2004;43(4):226–240.
3. Bortfeld TR, Kahler DL, Waldron TJ, Boyer AL. X-ray Field Compensation with Multileaf Collimators. Int J Radiat Oncol Biol Phys 1994;28:723–730. [PubMed: 8113118]
4. Chen DZ, Hu XS, Luan S, Wu X, Yu CX. Optimal Terrain Construction Problems and Applications in Intensity-Modulated Radiation Therapy, Lecture Notes in Computer Science, Springer-Verlag. Proc 10th Annual European Symp on Algorithms (ESA) 2002;2461:270–283.
5. Chen, DZ.; Hu, XS.; Luan, S.; Wang, C.; Wu, X. Geometric Algorithms for Static Leaf Sequencing Problems in Radiation Therapy. Proc. of 19th ACM Symp. on Computational Geometry (SoCG); San Diego, CA. June 2003; p. 88-97.
6. Chen, DZ.; Wang, C. Lecture Notes in Computer Science, Springer-Verlag, Proc. 17th International Symposium on Algorithms and Computation (ISAAC). 4288. 2006. Field Splitting Problems in Intensity-Modulated Radiation Therapy; p. 701-711.
7. Dai J, Zhu Y. Minimizing the Number of Segments in a Delivery Sequence for Intensity-Modulated Radiation Therapy with Multileaf Collimator. Med Phys 2001;28(10):2113–2120. [PubMed: 11695773]
8. Dogan N, Leybovich LB, Sethi A, Emami B. Automatic Feathering of Split Fields for Step-and-Shoot Intensity Modulated Radiation Therapy. Phys Med Biol 2003;48:1133–1140. [PubMed: 12765327]
9. Engel, K. A New Algorithm for Optimal Multileaf Collimator Field Segmentation. [March 2003]. http://www.trinity.edu/aholder/HealthApp/oncology/papers/paperlist.html
10. Hong L, Kaled A, Chui C, LoSasso T, Hunt M, Spirou S, Yang J, Amols H, Ling C, Fuks Z, Leibel S. IMRT of Large Fields: Whole-Abdomen Irradiation. Int J Radiat Oncol Biol Phys 2002;54:278–289. [PubMed: 12183002]
11. Kalinowski, T. An algorithm for optimal multileaf collimator field segmentation with interleaf collision constraint. [February 2003]. http://www.trinity.edu/aholder/HealthApp/oncology/papers/paperlist.html
12. Kamath S, Sahni S, Li J, Palta J, Ranka S. Leaf sequencing algorithms for segmented multileaf collimation. Phys in Med and Biol 2003;48(3):307–324. [PubMed: 12608609]
13. Kamath S, Sahni S, Palta J, Ranka S, Li J. Optimal leaf sequencing with elimination of tongue-and-groove underdosage. Phys in Med and Biol 2004;49:N7–N19. [PubMed: 15012015]
14. Kamath S, Sahni S, Li J, Palta J, Ranka S. Optimal Field Splitting for Large Intensity-Modulated Fields. Medical Physics 2004;31(12):3314–3323. [PubMed: 15651614]
15. Kamath S, Sahni S, Li J, Ranka S, Palta J. Generalized field splitting algorithms for optimal IMRT delivery efficiency. Physics in Medicine and Biology 2007;52:1–14. [PubMed: 17183124]
16. Gabow, HN.; Bentley, J.; Tarjan, RE. Proc 16th Annual ACM Symp Theory of Computing. 1984. Scaling and Related Techniques for Geometric Problems; p. 135-143.
17. Manber, U. Introduction to Algorithms: A Creative Approach. Addison-Wesley; 1989.
18. Potter LD, Chang SX, Cullip TJ, Siochi AC. A Quality and Efficiency Analysis of the *IMFAST*™ Segmentation Algorithm in Head and Neck “Step & Shoot” IMRT Treatments. Med Phys 2002;29(3):275–283. [PubMed: 11929009]

19. Que W. Comparison of Algorithms for Multileaf Collimator Field Segmentation. Med Phys 1999;26:2390–2396. [PubMed: 10587222]
20. Wu Q, Arnfield M, Tong S, Wu Y, Mohan R. Dynamic Splitting of Large Intensity-Modulated Fields. Phys Med Biol 2000;45:1731–1740. [PubMed: 10943915]
21. Wu, X.; Dou, X.; Bayouth, J.; Buatti, J. Lecture Notes in Computer Science. 4835. Springer Verlag; New Algorithm for Field Splitting in Radiation Therapy. Proc. of the 18th International Symposium on Algorithms and Computation (ISAAC); Sendai, Japan. December 2007; p. 692-703.
22. Siochi RAC. Minimizing Static Intensity Modulation Delivery Time Using an Intensity Solid Paradigm. Int J Radiat Oncol Giol Phys 1999;43:671–680.
23. Svensson R, Kallman P, Brahme A. An Analytical Solution for the Dynamic Control of Multileaf Collimation. Phys in Med and Biol 1994;39:37–61. [PubMed: 7651999]
24. Webb, S. Intensity-Modulated Radiation Therapy; Institute of Cancer Research and Royal Marsden NHS Trust; Jan. 2001;
25. Xia P, Verhey LJ. MLC Leaf Sequencing Algorithm for Intensity Modulated Beams with Multiple Static Segments. Med Phys 1998;25:1424–1449. [PubMed: 9725129]

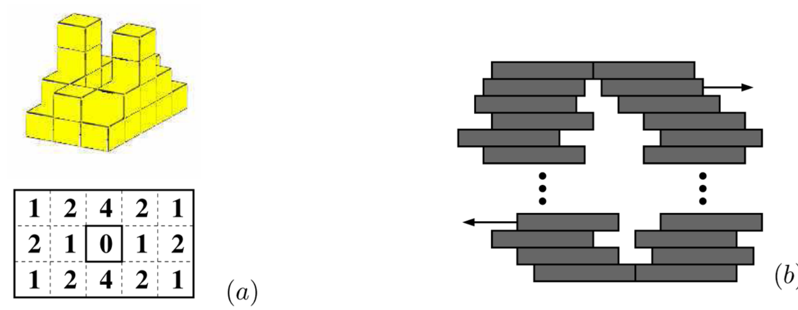
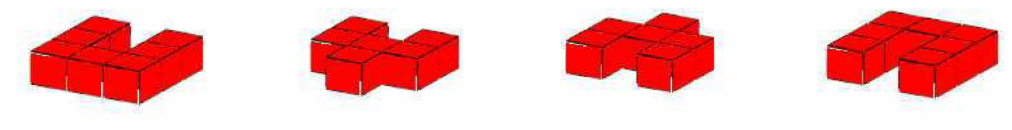


Figure 1.

(a) An intensity map *A* (bottom) and the corresponding 3-D IM mountain (top). (b) A multileaf collimator.



1	1	1	0	0
1	0	0	0	0
1	1	1	0	0

0	1	1	0	0
1	1	0	0	0
0	1	1	0	0

0	0	1	1	0
0	0	0	1	1
0	0	1	1	0

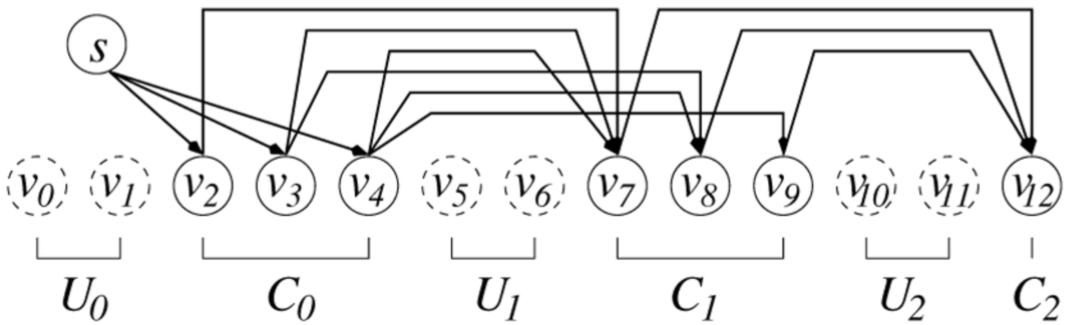
0	0	1	1	1
0	0	0	0	1
0	0	1	1	1

Figure 2.
Four MLC-apertures S_k's (bottom) and the corresponding plateaus (top) of a unit height for building the 3-D IM mountain in Figure 1(a).

NIH-PA Author Manuscript
NIH-PA Author Manuscript

0	3	6	5	2	7	8	1	6	9	5	0	0
2	4	1	1	6	6	9	12	12	16	8	6	2
1	1	3	12	9	16	16	7	8	12	8	3	1

(a)



(b)

Figure 3.
Illustrating the construction of the graph G. (a) An example IM A with size of 3 × 13 and the maximum leaf spread Δ = 5. (b) The graph G constructed from the IM A in (a).

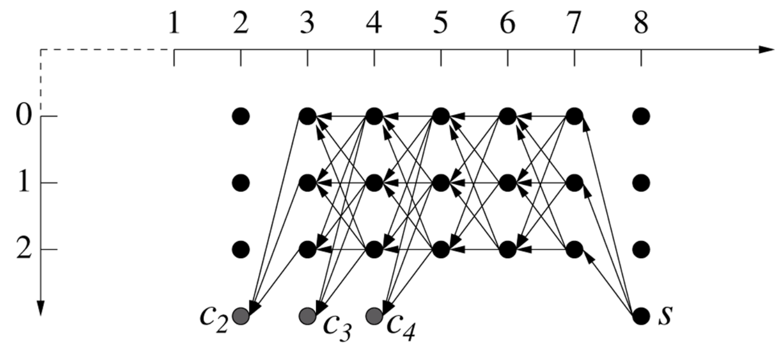


Figure 4.
Illustrating the construction of the graph H_q for computing $S(v_q)$ for the IM in Figure 2(a) with $\Delta = 5$ and $q = 7$.