OPTIMAL COMPACTION OF MULTIPLE TWO COMPONENT CHANNELS UNDER RIVER ROUTING

F. L. Heng

Andrea S. LaPaugh

CS-TR-068-86

December 1986

Optimal Compaction of Multiple Two Component Channels under River Routing†

F.L. Heng

Andrea S. LaPaugh

Department of Computer Science
Princeton University
Princeton, New Jersey 08544

Abstract

We develop an $O(kn^3)$ time algorithm to establish the tradeoff graph of minimum total separation (i.e. width) versus spread (i.e. length) given k parallel river routing channels, each bounded by a single component at the top and bottom, and a total of n nets. This solves the two dimensional compaction problem for a special case of slicing structured layout: a single hierarchical level structure with single layer interconnections between adjacent components. It serves as a first cut toward solving the two dimensional layout compaction problem in a slicing structure with one layer interconnection.

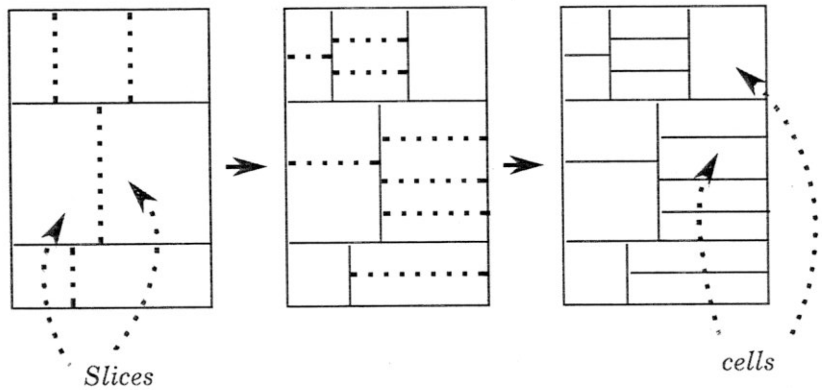
† This work was supported by DARPA Contract N00014-82-K-0549

Introduction:

Compacting VLSI layouts to produce minimum area designs is a very important issue in the VLSI design automation community. There are two major reasons: First, more compact layout allows maximal functionality; second, the yield of fabricated chips is inversely proportional to their areas. Producing an optimally compacted layout, *compaction,* is known to be a very hard problem [3, 11]. Most efficient compaction algorithms are one dimensional [e.g. 6, 7] : they work on the width and length of a layout independently. Some indirect interaction between the two dimensions can be obtained by iterating between horizontal and vertical compaction. However, two dimensional compaction is more desirable because it allows direct interactions between the two dimensions, potentially giving significantly better layouts. Kedem and Watanabe proposed a graph-optimization technique for two dimensional compaction that produces satisfactory layouts [2]. However it takes a tremendous amount of running time and is only suitable for small circuits.

The two dimensional compaction problem has been considered by others in a more tractable setting known as *slicing structure* [4, 9, 11, 12]. A *slicing structure* is obtained by hierarchically partitioning a rectangle into smaller rectangles, called *slices,* using parallel lines. Each of these slices is then divided into smaller slices using parallel lines that are orthogonal to the previous set of parallel lines. Slices resulting from the partitioning of a slice are called *children* of that slice. This process can be repeated to any depth. Each slice in which no further partitioning is applied is called a *cell.*(Figure 1) The compaction problem for a slicing structure is to find the shape function of the bounding rectangle given the shape function of each cell. A *shape function* is a stepwise monotonically decreasing tradeoff graph between horizontal and vertical dimensions of a slice (Figure 2). The minimum of any cost measure which is monotonically increasing in both dimensions can be found on this graph. (e.g. area = horizontal dimension * vertical dimension). More general shape functions are discussed in [9].

If no interconnection is considered, the shape function of a given slice can be obtained by composing the shape functions of its children. The shape function of the bounding rectangle can be computed in low order polynomial time [9, 11]. Unfortunately, with simple one layer interconnections the problem becomes NP-Complete [10].



Slicing structure

Figure 1

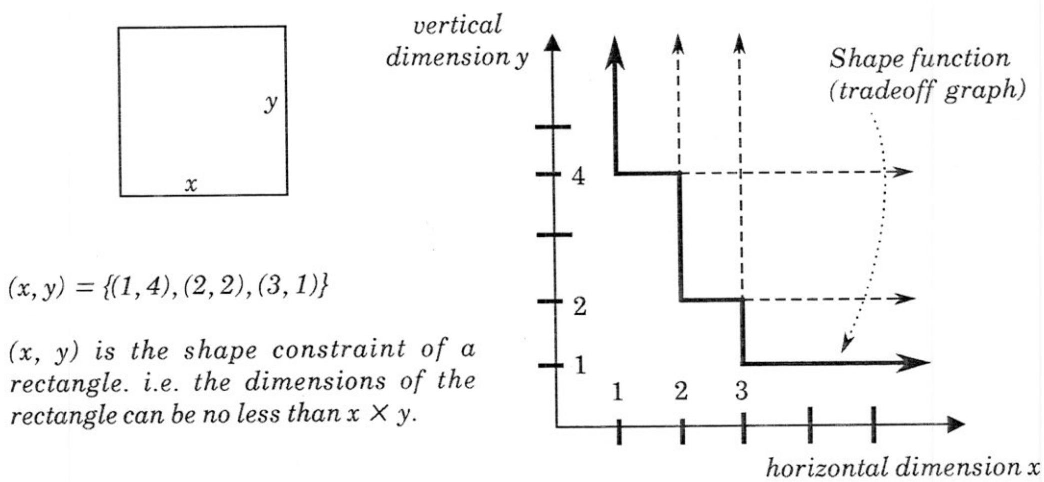
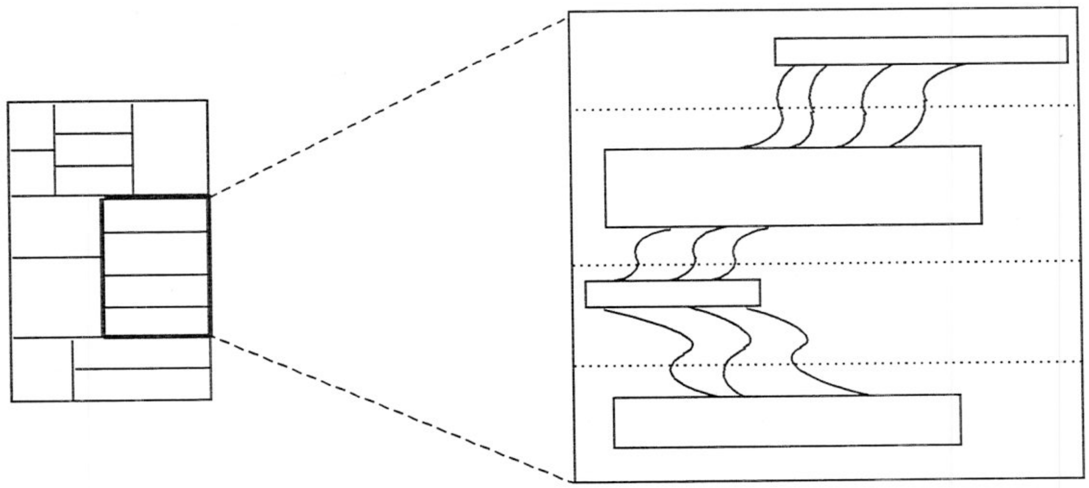


Figure 2

We look at an optimization problem with one layer interconnections at one hierarchical level in a slicing structure with fixed-shaped cells. We want to compute the tradeoff graph of a slice given the fixed shapes of its children and interconnection specifications between adjacent children. This is essentially a stack of rectangles with one layer interconnections between them. (Figure 3) We develop a polynomial time algorithm to establish the tradeoff graph. We intend this algorithm to be used as subroutine for a more efficient two dimensional compaction scheme.

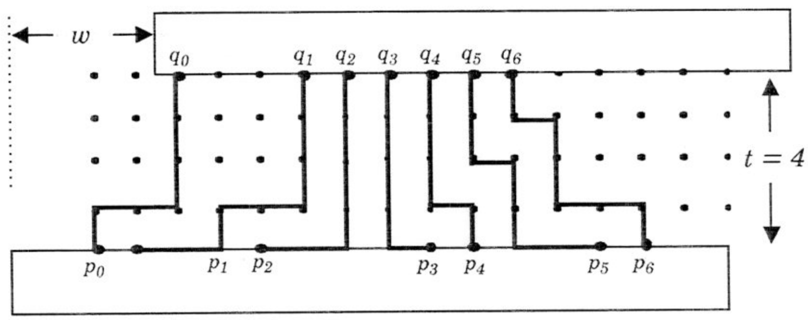


Two dimensional compaction in one hierarchical level

Figure 3

Background:

The *single channel river routing* problem has been studied extensively in the literature, see in particular [1, 5, 8, 10]. The problem is stated as follows: Given two rectangles, the top rectangle has m terminals $q_0, q_1, \ldots, q_{m-1}$ on its lower boundary; the bottom rectangle has m terminals $p_0, p_1, \ldots, p_{m-1}$ on its upper boundary. Each of the q_i's and p_i's denotes the position of a terminal with respect to the left boundary of the corresponding rectangle. Terminal q_i is to be routed to terminal p_i by wire w_i in a *single-layer-rectilinear* wiring model. In this model, wires are laid out along the grid lines of an integral grid and are separated by at least one grid unit. Each connecting pair p_i and q_i is called a *net*. Terminals on the same side of the rectangles are separated by integral grid units. The routing space between the rectangles is called the *channel*. The horizontal grid lines are called *tracks*. Tracks are numbered 0, 1, 2, ..., t from bottom to top. $t \geq 0$ is called the *separation* of that channel. For $t > 0$, wires are connected to track t vertically. An *offset* of a channel is the position of the left boundary of the top rectangle with respect to the left boundary of the bottom rectangle (Figure 4). A separation and offset pair *(t, w)* is said to be a *feasible* pair if there is a realizable routing with separation t at offset w. The *offset range* problem is, given a separation t, find the range of offsets w such that (t, w) is feasible. A



An instance of river routing

Figure 4

feasible set problem is to find all feasible (t, w) pairs. A necessary and sufficient condition for (t, w) to be feasible is given in [5, 8]:

$$(t, w) \text{ is feasible if and only if } L(t) \leq w \leq R(t)$$

Where,

$$L(t) = max\{p_{i-t} + t - q_i \,/\, t \leq i < m\}$$
$$L(m) = -\infty$$
$$R(t) = min\{p_{i+t} - t - q_i / 0 \leq i < m - t\}$$
$$R(m) = +\infty$$

Intuitively, when L(t) ≤ R(t), L(t) is the leftmost position of the top rectangle where (t, w) remains feasible, and R(t) is the rightmost position of the top rectangle where (t, w) remains feasible. Call L(t) the *left constraint* and R(t) the *right constraint* of the channel.

The following is true [8]:

$$L(0) \geq L(1) \geq L(2) \geq \ldots \geq L(m-1) \geq L(m)$$
$$R(0) \leq R(1) \leq R(2) \leq \ldots \leq R(m-1) \leq R(m)$$

To prove this, observe that in our rectilinear unit grid wiring model,

$$p_{i+1} \geq p_i + 1$$
$$\Rightarrow p_{i-t} + t - q_i \geq p_{i-(t+1)} + (t+1) - q_i \qquad \text{for } (t+1) \leq i < m$$
$$\Rightarrow L(t) \geq L(t+1)$$

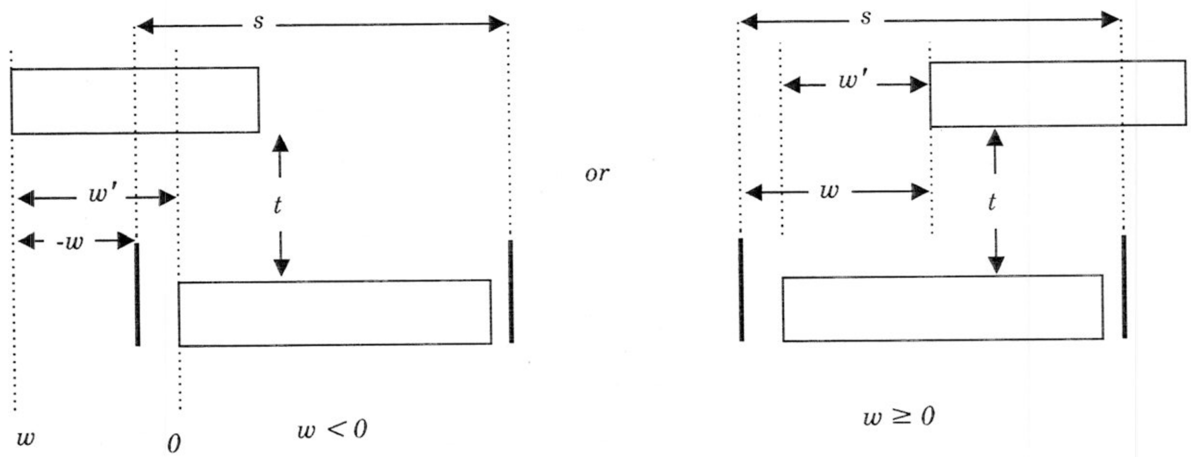
Similarly,

$$R(t) \leq R(t+1)$$

Notice that if w is feasible for t, then w is feasible for t+1. Since

$$L(t+1) \leq L(t) \leq w \leq R(t) \leq R(t+1)$$

Consider a variation of the feasible set problem in which the bottom rectangle is confined in a given horizontal span called the *spread*. In this case, the feasible set problem is to find, for a given separation t, the range of the top component with respect to the left boundary of the spread for which the separation is realizable. We do not restrict w to be positive, i.e. the top rectangle is not constrained by the spread. We say (s, t, w) is feasible if and only if (t, w') is feasible: see Figure 5. The *legal set*



Feasible arrangement

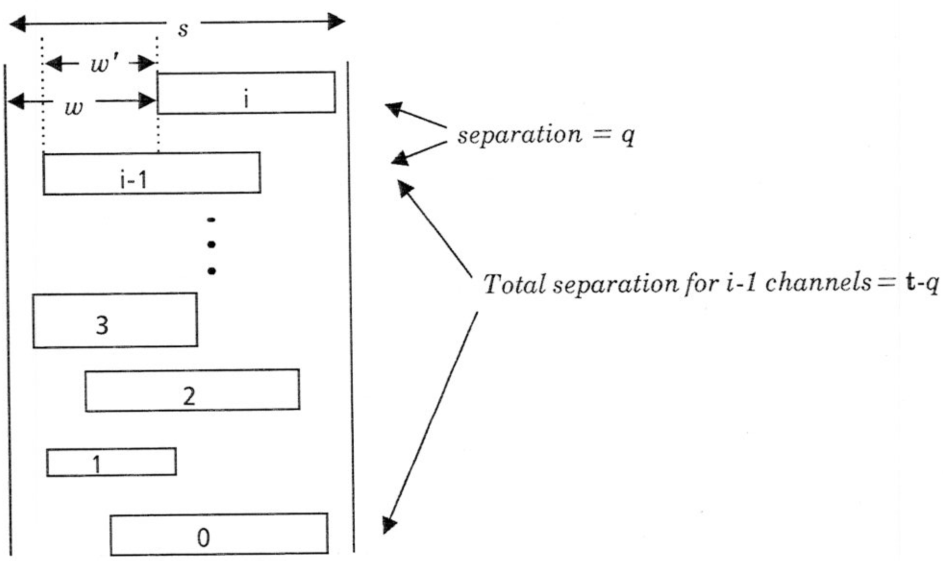
Figure 5

problem is to find all feasible points for which the top rectangle also lies within the spread.

Multiple channel single component:

We now extend the notions of feasibility and legality to multiple channels. Consider a stack of k+1 *components* which are rectangles. Each component contains terminals on both of its upper and lower boundaries, except the bottom and top components. The bottom component only contains terminals on its upper boundary and the top component only contains terminals on its lower boundary. Terminals on the upper boundary of the i^{th} component are to be routed to terminals on the lower boundary of the $(i+1)^{st}$ component. There are k channels in total and the sum of all k separations of each of the k channels is called the *total separation*.

For a given stack of $(i+1)$ components, a 3-tuple $(s, t, w)_i$ is said to be *legal* if the components can be positioned such that all of the $(i+1)$ components lie within the given spread s, each of the i channels is feasible, the top component is at position w with respect to the left boundary of the spread and the total separation is t. The 3-tuple is said to be *feasible* if the components can be positioned with total separation t such that the stack of i components is legal, and the top channel is feasible, i.e. the top component is not constrained by the spread. We call a particular set of positions for all the $(i+1)$ components such that $(s, t, w)_i$ is legal a *configuration*. (Figure 6)



Assume each channel is feasible and let tj be separation of j-th channel, then **t** $= \Sigma\, tj$

A configuration of a stack of $i+1$ components
Figure 6

Let,

m_i : number of nets in i^{th} channel.
$length_i$: length of i^{th} component.
$L_i(t)$: left constraint of i^{th} channel at separation t.

$R_i(t)$: right constraint of i^{th} channel at separation t.

T_i^* : the smallest separation such that

$$L_i(T_i^*) \leq R_i(T_i^*)$$

$$\text{i.e. } L_i(T_i^* - 1) > R_i(T_i^* - 1)$$

$T_0^* = 0$

We now define legality of a stack of $i+1$ components in terms of legality of the stack of i components and the i^{th} channel.

Definition:

For a given spread s, $(s, t, w)_i$ is legal if and only if $0 \leq w \leq s - length_i$ and there exists (q, w') such that

(1) (q, w') is feasible for the i^{th} channel.

(2) $(s, t\text{-}q, w\text{-}w')_{i\text{-}1}$ is legal.

For completeness,

$(s, t, w)_0$ is legal for $t \geq 0$ and $0 \leq w \leq s - length_0$,

$(t, w)_0$ is feasible for the same t and w.

We would like to find the left and right boundary functions for feasibility and legality to characterize the legal set of a stack of $i+1$ components.

Definitions:

$t_i^*(s)$ = minimum t such that $l_i(s, t) \leq r_i(s, t)$.

$l_0(s, t) = 0$, for $t \geq 0$, undefined otherwise.

$r_0(s, t) = s - length_0$, for $t \geq 0$, undefined otherwise.

$l_i'(s, t) = \min \{ l_{i-1}(s, t\text{-}q) + L_i(q) \mid T_i^* \leq q \leq t - t_{i-1}^*(s)\}$

= undefined if $t < t_{i-1}^*(s) + T_i^*$

$r_i'(s, t) = \max \{ r_{i-1}(s, t\text{-}q) + R_i(q) \mid T_i^* \leq q \leq t - t_{i-1}^*(s)\}$

= undefined if $t < t_{i-1}^*(s) + T_i^*$

$l_i(s, t) = \max \{0, l_i'(s, t)\}$, when $l_i'(s, t)$ is defined.

= undefined otherwise

$r_i(s, t) = \min \{ s - length_i, r_i'(s, t)\}$, when $r_i'(s, t)$ is defined.

= undefined otherwise

If there is an i such that $s\text{-}length_i < 0$, then the i^{th} component cannot be fitted in spread s, and the legal set is empty. Without loss of generality, we assume

$s \geq length_i$ for all i. We show that $l_i'(s, t)$, $r_i'(s, t)$, $l_i(s, t)$ and $r_i(s, t)$ are well defined, and that they indeed characterize feasibility and legality of an i-channel stack. Moreover we show $t_i^*(s)$ is the *minimum total separation* for a stack of $i+1$ components, i.e. for a given spread s in a stack of $i+1$ components there is no legal configuration for $t < t_i^*(s)$. First, we show that $l_i(s, t)$ and $r_i(s, t)$ are finite and $t_i^*(s)$ exists.

Lemma 1:

$l_i(s,t)$ and $r_i(s,t)$ are defined and finite for all $t \geq t_{i-1}^*(s) + T_i^*$, in addition

$$l_i(s, t) \geq l_i(s, t+1)$$
$$r_i(s, t) \leq r_i(s, t+1) \qquad \text{for all } t \geq t_{i-1}^*(s) + T_i^*$$

and $\quad t_i^*(s) \geq t_{i-1}^*(s) + T_i^*$ exists.

Proof:

By induction on i.

Basis : $l_0(s,t)$ and $r_0(s,t)$ are defined and finite for $t \geq 0$.

$$0 = l_0(s, 0) = l_0(s, 1) = \ldots$$
$$s - length_0 = r_0(s, 0) = r_0(s, 1) = \ldots \qquad \text{for } t \geq 0.$$

and $\quad t_0^*(s) = 0$

Induction step:

Assume $l_{i-1}(s,t)$ and $r_{i-1}(s,t)$ are defined and finite for all $t \geq t_{i-2}^*(s) + T_{i-1}^*$

$$l_{i-1}(s, t) \geq l_{i-1}(s, t+1)$$
$$r_{i-1}(s, t) \leq r_{i-1}(s, t+1) \qquad \text{for all } t \geq t_{i-2}^*(s) + T_{i-1}^*$$

and $\quad t_{i-1}^*(s) \geq t_{i-2}^*(s) + T_{i-1}^*$ exists.

In the following we assume all $t \geq t_{i-1}^*(s) + T_i^*$ or mention otherwise. We show $l_i'(s, t)$ is defined and cannot be $+\infty$, this implies $l_i(s, t)$ is defined and finite, since $l_i(s, t) = \max\{0, l_i'(s, t)\}$.

$l_i'(s, t) = \min\{ l_{i-1}(s, t-q) + L_i(q) \mid T_i^* \leq q \leq t - t_{i-1}^*(s) \}$ is defined. Because for $q = T_i^*$, $t-T_i^* \geq t_{i-1}^*(s) \geq t_{i-2}^*(s) + T_{i-1}^*$, $l_{i-1}(s, t-T_i^*)$ is defined and finite, and $L_i(T_i^*)$ is not $+\infty$. Therefore $l_i'(s, t)$ is defined and cannot be $+\infty$, so $l_i(s, t)$ is defined and finite. Similarly, $r_i(s, t)$ is defined and finite. In addition,

$$
\begin{aligned}
l_i'(s, t) &= l_{i-1}(s, t-q') + L_i(q') && \text{for some } q' \in [T_i^*, t - t_{i-1}^*(s)] \\
&\geq l_{i-1}(s, t+1-q') + L_i(q') && \text{inductive hypothesis} \\
&\geq l_i'(s, t+1)
\end{aligned}
$$

therefore, $l_i'(s, t) \geq l_i'(s, t+1)$

If $l_i'(s, t) < 0$, then $l_i(s, t) = 0$, and if $l_i'(s, t) \geq 0$, then $l_i(s, t) = l_i'(s, t)$. Therefore $l_i(s, t) \geq l_i(s, t+1)$ for $t \geq t_{i-1}^*(s) + T_i^*$. Similarly, $r_i(s, t) \leq r_i(s, t+1)$ for $t \geq t_{i-1}^*(s) + T_i^*$.

To show that $t_i^*(s)$ exists, we find a t such that $l_i(s, t) \leq r_i(s, t)$. Since $l_i(s, t)$ is decreasing and $r_i(s, t)$ is increasing, there is a smallest $t \geq t_{i-1}^*(s) + T_i^*$ such that $l_i(s, t) \leq r_i(s, t)$. This minimum t is the $t_i^*(s)$ we are seeking and $t_i^*(s) \geq t_{i-1}^*(s) + T_i^*$.

$L_i(m_i) = -\infty$ and $R_i(m_i) = +\infty$, since any offset is feasible for separation m_i.

Therefore,

$l_i'(s, t_{i-1}^*(s) + m_i) \leq l_{i-1}(s, t_{i-1}^*(s)) + L_i(m_i) < 0$, because $l_{i-1}(s, t_{i-1}^*(s))$ is finite
$r_i'(s, t_{i-1}^*(s)+m_i) \geq r_{i-1}(s, t_{i-1}^*(s))+R_i(m_i) > \text{s-length}_i$, because $r_{i-1}(s, t_{i-1}^*(s))$ is finite

Hence,

$l_i(s,t_{i-1}^*(s) + m_i) = 0$, and $r_i(s, t_{i-1}^*(s)+m_i) = \text{s - length}_i \geq 0$, therefore
$l_i(s,t_{i-1}^*(s) + m_i) \leq r_i(s, t_{i-1}^*(s)+m_i)$,

but,

$l_i(s, t) \geq l_i(s,t+1)$
$r_i(s, t) \leq r_i(s,t+1)$

there is $t \in [\, t_{i-1}^*(s) + T_i^*,\ \ t_{i-1}^*(s) + m_i \,]$ such that $l_i(s, t) \leq r_i(s, t)$, and if both $l_i(s, t-1)$ and $r_i(s, t-1)$ are defined $l_i(s, t-1) > r_i(s, t-1)$. If $t = t_{i-1}^*(s) + T_i^*$, then $l_i(s, t-1)$ and $r_i(s, t-1)$ are undefined. In each case $t_i^*(s) = t$. Q.E.D.

Given Lemma 1, we know $l_i(s, t)$ is defined if and only if $t \geq t_{i-1}^*(s) + T_i^*$. For the rest of this paper $l_i(s, t)$ is assumed to be defined, i.e. $t \geq t_{i-1}^*(s) + T_i^*$, whenever it is used, unless mentioned otherwise. The same assumption is applied for $r_i(s, t)$. Notice that $l_i(s, t) \leq r_i(s, t)$ if and only if $t \geq t_i^*(s)$.

The following lemma shows that, in fact $l_i(s, t)$ and $r_i(s, t)$ characterize exactly the legal set of an i-channel stack.

Lemma 2:

$(s, t, w)_i$ is legal if and only if $l_i(s, t) \leq w \leq r_i(s, t)$.

Proof:

Basis :

$(s, t, w)_0$ is legal $\Leftrightarrow l_0(s, t) = 0 \leq w \leq s - length_0 = r_0(s, t)$

Induction step:

Assume $(s, t, w)_{i-1}$ is legal $\Leftrightarrow l_{i-1}(s, t) \leq w \leq r_{i-1}(s, t)$. The second condition implies $t \geq t_{i-1}^*(s)$.

[$\Rightarrow$] Assume $(s, t, w)_i$ is legal. Then $0 \leq w \leq s - length_i$, and there is (q, w') such that

(1) $(q, w')_i$ is feasible, i.e. $L_i(q) \leq w' \leq R_i(q)$

(2) $(s, t\text{-}q, w\text{-}w')_{i-1}$ is legal, i.e. $l_i(s, t\text{-}q) \leq w - w' \leq r_i(s, t)$. By induction hypothesis.

But, for some $q', q'' \in [T_i^*, t - t_{i-1}^*(s)]$

$$l_i'(s, t) = l_{i-1}(s, t\text{-}q') + L_i(q') \leq l_{i-1}(s, t\text{-}q) + L_i(q) \leq l_{i-1}(s, t\text{-}q) + w' \leq w\text{-}w' + w' = w$$
$$r_i'(s, t) = r_{i-1}(s, t\text{-}q'') + R_i(q'') \geq r_{i-1}(s, t\text{-}q) + R_i(q) \geq r_{i-1}(s, t\text{-}q) + w' \geq w\text{-}w' + w' = w$$

and $0 \leq w \leq s - length_i$, therefore $l_i(s, t) \leq w \leq r_i(s, t)$.

To prove the sufficient condition for legality we need the following:

Claim:

If there is $q \in [T_i^*, t\text{-}t_{i-1}^*(s)]$ such that $l_{i-1}(s, t\text{-}q) + L_i(q) \leq w \leq r_{i-1}(s, t\text{-}q) + R_i(q)$,

then

there is $w' \in [L_i(q), R_i(q)]$ such that $l_{i-1}(s, t\text{-}q) \leq w - w' \leq r_{i-1}(s, t\text{-}q)$.

i.e. we can find w' such that $(q, w')_i$ is feasible and $(s, t\text{-}q, w\text{-}w')_{i-1}$ is legal. (Figure 7)

Notice that $l_{i-1}(s, t\text{-}q) + L_i(q) \leq r_{i-1}(s, t\text{-}q) + L_i(q) \leq r_{i-1}(s, t\text{-}q) + R_i(q)$, since $q \in [T_i^*, t\text{-}t_{i-1}^*(s)]$, i.e. $t\text{-}q \geq t_{i-1}^*(s)$.

If $l_{i-1}(s, t\text{-}q) + L_i(q) \leq w \leq r_{i-1}(s, t\text{-}q) + L_i(q)$, choose $w' = L_i(q)$.

Then

$L_i(q) \leq w' = L_i(q) \leq R_i(q)$, since $q \geq T_i^*$

and

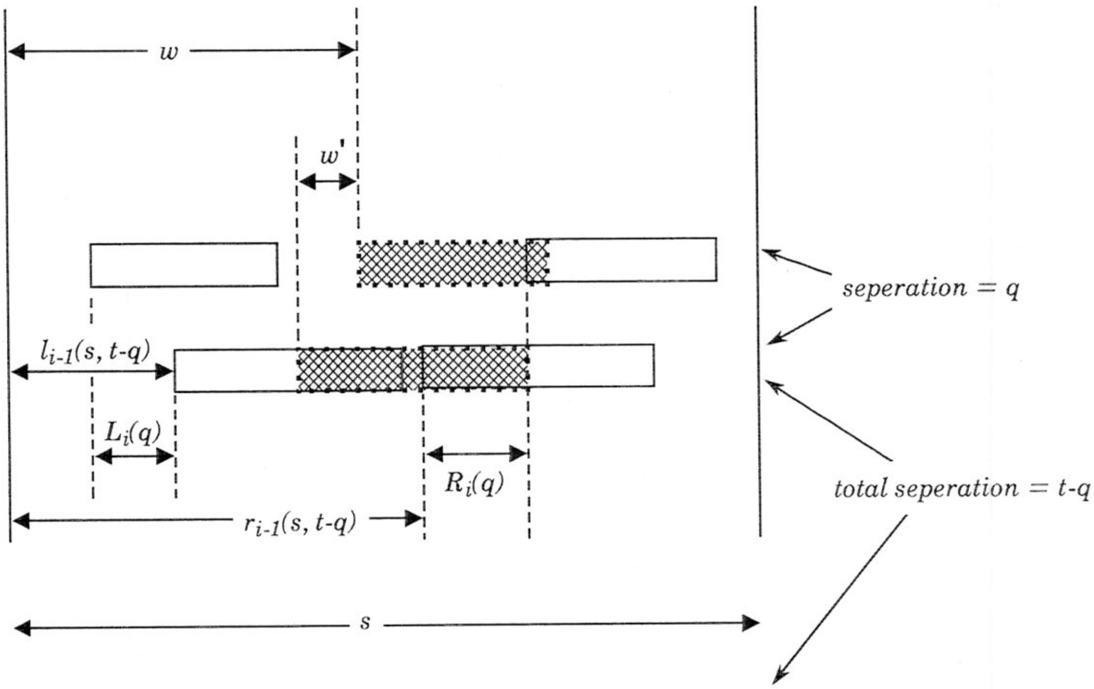


Figure 7

$l_{i-1}(s, t\text{-}q) \le w - w' \le r_{i-1}(s, t\text{-}q)$.

If $r_{i-1}(s, t\text{-}q) + L_i(q) \le w \le r_{i-1}(s, t\text{-}q) + R_i(q)$, choose $w' = w - r_{i-1}(s, t\text{-}q)$,

then

$L_i(q) \le w' \le R_i(q)$,

and

$l_{i-1}(s, t\text{-}q) \le r_{i-1}(s, t\text{-}q) = w - w' \le r_{i-1}(s, t\text{-}q)$, since $t\text{-}q \ge t_{i-1}^*(s)$. END CLAIM.

[⇐] Assume $l_i(s, t) \le w \le r_i(s, t)$, this implies $t \ge t_i^*(s)$ and $0 \le w \le s - length_i$. Then $l_i'(s,t) \le l_i(s, t) \le w \le r_i(s, t) \le r_i'(s, t)$.

i.e. $l_i'(s,t) = l_{i-1}(s, t\text{-}q') + L_i(q') \le w \le r_{i-1}(s, t\text{-}q'') + R_i(q'') = r_i'(s, t)$, for some $q', q'' \in [T_i^*, t\text{-}t_{i-1}^*(s)]$

Notice that $r_{i-1}(s, t\text{-}q') + R_i(q') \le r_i'(s, t) = r_{i-1}(s, t\text{-}q'') + R_i(q'')$. There are two cases.

Case 1: $w \le r_{i-1}(s, t\text{-}q') + R_i(q')$. Since $0 \le w \le s - length_i$, then by above claim, $(s, t, w)_i$ is legal.

Case 2: $r_{i-1}(s, t\text{-}q') + R_i(q') \le w \le r_{i-1}(s, t\text{-}q'') + R_i(q'')$.

(i) if $t\text{-}q' \le t\text{-}q''$, i.e. $q' \ge q''$, then

$$\begin{aligned} w &\ge r_{i-1}(s, t\text{-}q') + R_i(q') \\ &\ge r_{i-1}(s, t\text{-}q') + R_i(q''), \text{ since } R_i(q') \ge R_i(q'') \\ &\ge l_{i-1}(s, t\text{-}q') + R_i(q''), \text{ since } t\text{-}q' \ge t_{i-1}^*(s) \\ &\ge l_{i-1}(s, t\text{-}q'') + R_i(q''), \text{ since } l_{i-1}(s, t\text{-}q') \ge l_{i-1}(s, t\text{-}q'') \\ &\ge l_{i-1}(s, t\text{-}q'') + L_i(q'') \end{aligned}$$

Since $0 \le w \le s\text{-length}_i$, by the above claim, $(s, t, w)_i$ is legal.

(ii) if $t\text{-}q' > t\text{-}q''$, i.e. $q' < q''$, then

$$\begin{aligned} w &\ge r_{i-1}(s, t\text{-}q') + R_i(q') \\ &\ge r_{i-1}(s, t\text{-}q') + L_i(q'), \\ &\ge r_{i-1}(s, t\text{-}q') + L_i(q''), \text{ since } L_i(q') \ge L_i(q'') \\ &\ge r_{i-1}(s, t\text{-}q'') + L_i(q''), \text{ since } r_{i-1}(s, t\text{-}q') \ge r_{i-1}(s, t\text{-}q'') \\ &\ge l_{i-1}(s, t\text{-}q'') + L_i(q''), \text{ since } t\text{-}q'' \ge t_{i-1}^*(s) \end{aligned}$$

Since $0 \le w \le s\text{-length}_i$, by the above claim, $(s, t, w)_i$ is legal. Q.E.D.

Lemma 1 and Lemma 2 imply $t_i^*(s)$ is indeed the minimum total separation. And by Lemma 2, the legal set problem for a given spread s is equivalent to computing $l_k(s, t)$ and $r_k(s, t)$. In the actual computation we need only to compute $l_i(s, t)$ and $r_i(s, t)$ for $t \le \Sigma m_j$, since the largest separation required to river route a channel with m_i nets is m_i. With separation m_i at channel i, we can place the i^{th} component at an arbitrary position within the spread. Therefore for $t = \Sigma m_j$, allocate m_j tracks for the jth channel, and $l_i(s, t) = 0$ and $r_i(s, t) = s - \text{length}_i$. And for $t > \Sigma m_j$, $l_i(s, t) = l_i(s, \Sigma m_j)$. Examining the definition of $l_i'(s, t)$, we see that to compute $l_i'(s, t)$, we are in fact making an off diagonal sweep of the $(l_{i-1}(s, *) \times L_i(*))$ matrix in which the $\alpha\beta$-th entry is $l_{i-1}(s, \alpha) + L_i(\beta)$.

Let,

$$M_i = \sum_{j=1}^{i} m_j, \qquad n = M_k$$

The number of entries in the $(l_{i-1}(s, *) \times L_i(*))$ matrix is $M_{i-1} * m_i$. Similarly, to compute $r_i'(s, t)$ we make an off diagonal sweep of the $(r_{i-1}(s, *) \times R_i(*))$ matrix whose $\alpha\beta$-th entry is $r_{i-1}(s, \alpha) + R_i(\beta)$. Therefore the complexity of computing $l_k(s, t)$ and $r_k(s, t)$ is

$$C * \sum_{i=1}^{k} M_{i-1} * m_i \leq C * \sum_{i=1}^{k} n * m_i \leq C * n^2 = O(n^2), \text{ for some constant } C$$

In fact the actual number of entries we are required to consider in the $(l_{i-1}(s, *) \text{ X } L_i(*))$ matrix is less than $m_i * M_{i-1}$, because some of the matrix entries are undefined. To be more precise, $l_{i-1}(s, \alpha)$ is only defined for $\alpha \geq T_{i-1}^* + t_{i-2}^*(s)$ and $L_i(\beta)$ is only defined for $\beta \geq T_i^*$. Therefore, only $(M_{i-1} - (T_{i-1}^* + t_{i-2}^*(s)) + 1) * (m_i - T_i^* + 1)$ entries are defined. The same observation also applies to the $(r_{i-1}(s, *) \text{ X } R_i(*))$ matrix.

Above, we have shown how to calculate legal sets for a particular spread. To establish the tradeoff graph of minimum separation versus spread, it is necessary to establish the relation between them for all spreads. We introduce the notion of *critical spreads*. Critical spreads at level i are spreads at which the minimum total separation may change at level i-1 or level i. The following two lemmas characterize exactly how the increase in spread affects the minimum total separation and the corresponding left and right boundary functions of legality. Lemma 3 reveals that if the minimum total separation remains constant over a small increment of spread, the left boundary function of legality remains unchanged and the right boundary function of legality increases by the same amount that the spread increases. Lemma 4 is a direct consequence of Lemma 3 and shows the existence of critical spreads. The proof of Lemma 4 is constructive and provides an algorithm to construct the critical spreads.

Lemma 3:

Given $\Delta \geq 0$, if $t_{i-1}^*(s + \Delta) = t_{i-1}^*(s)$, and for all $t \geq t_{i-2}(s) + T_{i-1}^*$, $l_{i-1}(s+\Delta, t) = l_{i-1}(s, t)$, and $r_{i-1}(s + \Delta, t) = r_{i-1}(s, t) + \Delta$, then for all $t \geq t_{i-1}(s) + T_i^*$

$$l_i(s + \Delta, t) = l_i(s, t)$$
$$r_i(s + \Delta, t) = r_i(s, t) + \Delta$$

Proof:

For $t \geq t_{i-1}(s) + T_i^*$

$$l_i'(s, t) = \min \{ l_{i-1}(s, t\text{-}q) + L_i(q) \mid T_i^* \leq q \leq t - t_{i-1}^*(s) \}$$
$$l_i'(s + \Delta, t) = \min \{ l_{i-1}(s + \Delta, t\text{-}q) + L_i(q) \mid T_i^* \leq q \leq t - t_{i-1}^*(s + \Delta) \}$$
$$= \min \{ l_{i-1}(s, t\text{-}q) + L_i(q) \mid T_i^* \leq q \leq t - t_{i-1}^*(s) \}$$
$$= l_i'(s, t)$$

But

$l_i(s, t) = \max\{0, l_i'(s, t)\}$, therefore $l_i(s + \Delta, t) = l_i(s, t)$.

$$r_i'(s, t) = \max\{r_{i-1}(s, t-q) + R_i(q) \mid T_i^* \le q \le t - t_{i-1}^*(s)\}$$

$$\begin{aligned} r_i'(s + \Delta, t) &= \max\{r_{i-1}(s + \Delta, t-q) + R_i(q) \mid T_i^* \le q \le t - t_{i-1}^*(s + \Delta)\} \\ &= \max\{r_{i-1}(s, t) + \Delta + R_i(q) \mid T_i^* \le q \le t - t_{i-1}^*(s)\} \\ &= \max\{r_{i-1}(s, t) + R_i(q) \mid T_i^* \le q \le t - t_{i-1}^*(s)\} + \Delta \\ &= r_i'(s, t) + \Delta \end{aligned}$$

But,

$r_i(s, t) = \min\{s - length_i, r_i'(s, t)\}$, and

$$\begin{aligned} r_i(s + \Delta, t) &= \min\{s + \Delta - length_i, r_i'(s + \Delta, t)\} \\ &= \min\{s + \Delta - length_i, r_i'(s, t) + \Delta\} \\ &= \min\{s - length_i, r_i'(s, t)\} + \Delta \\ &= r_i(s, t) + \Delta \end{aligned}$$

Q.E.D.

Lemma 4:

Given s and s' such that $s < s'$, and for all Δ such that $0 \le \Delta < s' - s$, and all $t \ge t_{i-2}(s) + T_{i-1}^*$:

$$t_{i-1}^*(s + \Delta) = t_{i-1}^*(s),$$
$$l_{i-1}(s + \Delta, t) = l_{i-1}(s, t),$$
$$r_{i-1}(s + \Delta, t) = r_{i-1}(s, t) + \Delta,$$

then there exists a sequence of spreads

$$s = S_0 < S_1 < \ldots < S_m = s'$$

such that for each $j = 0, 1, \ldots, m-1$ and all δ such that $0 \le \delta < S_{j+1} - S_j$, and all $t \ge t_{i-1}(s) + T_i^*$

$$t_i^*(S_0) > t_i^*(S_1) > t_i^*(S_2) > \ldots > t_i^*(S_{m-1}) \ge t_i^*(S_m),$$
$$t_i^*(S_j) = t_i^*(S_j + \delta),$$
$$l_i(S_j + \delta, t) = l_i(S_j, t),$$
$$r_i(S_j + \delta, t) = r_i(S_j, t) + \delta.$$

Proof:

We prove this by constructing such a sequence.

Compute $l_i(s, t)$, $r_i(s, t)$ and $t_i^*(s)$ for all t. If $t_i^*(s) = T_i^* + t_{i-1}^*(s)$ the sequence consists of two elements namely $S_0 = s$ and $S_1 = s'$. If $t_i^*(s) > T_i^* + t_{i-1}^*(s)$, i.e. $l_i(s, t_i^*(s) - 1)$ and $r_i(s, t_i^*(s) - 1)$ are defined (Lemma 1), then choose

$S_0 = s,$
$S_1 = S_0 + l_i(S_0, t_i^*(S_0)\text{-}1) - r_i(S_0, t_i^*(S_0)\text{-}1).$

We repeat this process by choosing $S_{j+1} = S_j + l_i(S_j, t_i^*(S_j)\text{-}1) - r_i(S_j, t_i^*(S_j)\text{-}1)$. We stop when either $t_i^*(S) = T_i^* + t_{i-1}^*(S_{j+1})$, or $S_{j+1} \geq s'$. In either case we choose $S_{j+1} = s'$. In addition, for $0 \leq \delta < S_{j+1} - S_j$, by hypothesis

$$\begin{aligned}
t_{i-1}^*(S_j+\delta) &= t_{i-1}^*(s+(S_j+\delta\text{-}s)) = t_{i-1}^*(s) = t_{i-1}^*(s+(S_j\text{-}s)) = t_{i-1}^*(S_j) \\
l_{i-1}(S_j+\delta, t) &= l_{i-1}(s+(S_j+\delta\text{-}s), t) = l_{i-1}(s, t) = l_{i-1}(s+(S_j\text{-}s), t) = l_{i-1}(S_j, t) \\
r_{i-1}(S_j+\delta, t) &= r_{i-1}(s+(S_j+\delta\text{-}s), t) = r_{i-1}(s,t)+S_j+\delta\text{-}s = r_{i-1}(s,t)+(S_j\text{-}s)+\delta \\
&= r_i(s+S_j\text{-}s, t) + \delta = r_{i-1}(S_j, t) + \delta
\end{aligned}$$

Therefore, by Lemma 3, $l_i(S_j+\delta, t) = l_i(S_j, t)$ and $r_i(S_j+\delta, t) = r_i(S_j, t)$. In fact, for $S_{j+1} < s'$ the above argument also holds for $\delta = S_{j+1} - S_j$. Moreover, for $0 \leq \delta < S_{j+1} - S_j$ and $S_{j+1} \leq s'$

$$\begin{aligned}
l_i(S_j + \delta, t_i^*(S_j)) &= l_i(S_j, t_i^*(S_j)) \leq r_i(S_j, t_i^*(S_j)) \leq r_i(S_j, t_i^*(S_j)) + \delta \\
&= r_i(S_j + \delta, t_i^*(S_j))
\end{aligned}$$

and when $t_i^*(S_j) - 1 \geq T_i^* + t_{i-1}^*(S_j)$

$$\begin{aligned}
r_i(S_j + \delta, t_i^*(S_j)\text{-}1) &= r_i(S_j, t_i^*(S_j)\text{-}1) + \delta < r_i(S_j, t_i^*(S_j)\text{-}1) + S_{j+1} - S_j \\
&= l_i(S_j, t_i^*(S_j)\text{-}1) = l_i(S_j + \delta, t_i^*(S_j)\text{-}1)
\end{aligned}$$

i.e. $\quad l_i(S_j + \delta, t_i^*(S_j)) \leq r_i(S_j + \delta, t_i^*(S_j))$
and $\quad l_i(S_j + \delta, t_i^*(S_j)\text{-}1) > r_i(S_j + \delta, t_i^*(S_j)\text{-}1)$,
therefore $t_i^*(S_j + \delta) = t_i^*(S_j)$.

Also for $S_{j+1} < s'$

$$\begin{aligned}
l_i(S_{j+1}, t_i^*(S_j)\text{-}1) &= l_i(S_j + (S_{j+1} - S_j), t_i^*(S_j)\text{-}1) = l_i(S_j, t_i^*(S_j)\text{-}1) \\
r_i(S_{j+1}, t_i^*(S_j)\text{-}1) &= r_i(S_j + (S_{j+1} - S_j), t_i^*(S_j)\text{-}1) = r_i(S_j, t_i^*(S_j)\text{-}1) + (S_{j+1} - S_j) \\
&= l_i(S_j, t_i^*(S_j)\text{-}1)
\end{aligned}$$

therefore $t_i^*(S_{j+1}) \leq t_i^*(S_j)\text{-}1 < t_i^*(S_j)$.

Let m be the index such that $S_m = s'$. To show that $t_i^*(S_{m-1}) \geq t_i^*(S_m)$, observe that for any spread S and $S' > S$, $l_i(S, t) \geq l_i(S', t)$ and $r_i(S, t) \leq r_i(S', t)$, for all t such that $l_i(S, t)$ and $r_i(S, t)$ are defined. This means that for a larger spread, the top component has a larger range of legality, which is intuitively true. In fact this can be proved by simple induction which we leave to the reader. Since $S_{m-1} < S_m$,

$$l_i(S_m, t_i^*(S_{m-1})) \leq l_i(S_{m-1}, t_i^*(S_{m-1})) \leq r_i(S_{m-1}, t_i^*(S_{m-1})) \leq r_i(S_m, t_i^*(S_{m-1}))$$

i.e. $l_i(S_m, t_i^*(S_{m-1})) \leq r_i(S_m, t_i^*(S_{m-1}))$, therefore $t_i^*(S_{m-1}) \geq t_i^*(S_m)$. Q.E.D.

Formally we compute the sequence as follow:

SEQUENCE(s, s')
$S_0 = s$
Compute $l_i(S_0, t), r_i(S_0, t)$ *for all* t.
$j = 0$
While $S_j < s'$
    Compute $t_i^*(S_j)$
    If $t_i^*(S_j) = T_i^* + t_{i-1}^*(s)$ *then*
        $S_{j+1} = s'$
    Else
        $S_{j+1} = S_j + l_i(S_j, t_i^*(S_j) - 1) - r_i(S_j, t_i^*(S_j) - 1)$
    $j = j + 1$

The complexity of SEQUENCE is $O(n^2)$. It requires $O(n^2)$ time to compute $l_i(s, t)$ and $r_i(s, t)$. The time required for the loop is (number of S_j's) $*$ (time to compute $t_i^*(S_j)$). (Number of S_j's) is equal to (number of $t_i^*(S_j)$). But $t_i^*(S_j) \leq M_i \leq n$, and the sequence of $t_i^*(S_j)$ is strictly decreasing. Therefore (number of S_j's) $\leq M_i \leq n$. As in Lemma 1, $t_i^*(S_j)$ can be computed in $O(t_i^*(S_j))$ time. So SEQUENCE can be computed in $O(n^2)$ time. To be more precise, if information on $l_{i-1}(*, *)$ and $r_{i-1}(*, *)$ is kept, $l_i(s, t)$ and $r_i(s, t)$ can be computed in $O(M_{i-1} * m_i)$ time (see Lemma 2). If a pointer is kept on $t_i^*(S_j)$, the sequence of $t_i^*(S_j)$'s and S_j's can be computed in $O(M_i)$ time. Thus SEQUENCE can be implemented in time $O(M_{i-1} * m_i + M_i)$.

Theorem:

The tradeoff graph of minimum total separation versus spread can be computed in $O(kn^3)$ time.

Proof:

We show this by providing an algorithm to compute a sequence of spreads

$$\max\{length_i\} = S_0 < S_1 < S_2 < \ldots < S_N = \Sigma\, length_i$$

such that

$$t_k^*(S_0) \geq t_k^*(S_1) \geq \ldots \geq t_k^*(S_N) \text{ and } t_k^*(S_i) = t_k^*(S_i + \Delta), \text{ for } 0 \leq \Delta < S_{i+1} - S_i.$$

Notice that this sequence of spread covers the range of essential spreads, since for any spread $s > S_N$, $t_k^*(s) = \Sigma\, T_i^*$.

We construct sequences inductively until we have constructed $S_0, S_1, ..., S_N$.

Begin with component 0.
$s = \max\{length_i\}$, $s' = \Sigma\, length_i$.
For $0 \le \Delta < s' - s$, and $t \ge 0$,

$$t_0^*(s + \Delta) = 0 = t_0^*(s),$$
$$l_0(s+\Delta, t) = l_0(s, t) = 0,$$
$$r_0(s+\Delta, t) = s+\Delta\text{-}length_0 = r_0(s, t) + \Delta$$

By Lemma 4, we can construct a sequence of S_i^1's such that

$$s = S_0^1 < S_1^1 < S_2^1 < ... < S_m^1 = s'.$$
$$t_1^*(S_0^1) > t_1^*(S_1^1) > ... > t_1^*(S_{m-1}^1) \ge t_1^*(S_m^1),$$

and for $0 \le i < m$, and $0 \le \Delta < S_{i+1}^1 - S_i^1$

$$t_1^*(S_i^1) = t_1^*(S_i^1 + \Delta).$$
$$l_1(S_i^1 + \Delta, t) = l_1(S_i^1, t),$$
$$r_1(S_i^1 + \Delta, t) = r_1(S_i^1, t) + \Delta.$$

In general we get a set of sequences at channel i: each of these sequences is generated by applying Lemma 4 to a pair of consecutive elements of the sequence for channel i-1. The sequence for channel i is the concatenation in order of these sequences. The algorithm is described as follow:

FINAL_SEQUENCE:
Let $\Psi_j + 1$ *be the number of* S^j*'s.*
$S_0^0 = \max\{length_i\}, S_1^0 = \Sigma\, length_i$
$i = 1, \Psi_0 = 1$
While $i \le k$
  $j = 0, l = 0$
  While $j < \Psi_{i-1}$
    compute a sequence $S_j^{i-1} = S_l^i < S_{l+1}^i < S_{l+2}^i < ... < S_{l+a+1}^i = S_{j+1}^{i-1}$
    as in Lemma 3. This is done by calling SEQUENCE(S_j^{i-1}, S_{j+1}^{i-1}*)*
    $l = l + a + 1$
    $j = j + 1$
  End while j
  $\Psi_i = l$
  $i = i + 1$
End while i

FINAL_SEQUENCE produces the required sequence. For each S_i^k, $l_k(S_i^k, t)$, $r_k(S_i^k, t)$ and $t_k^*(S_i^k)$ are computed in the course of computing the S_i^k. Thus the complete tradeoff graph is obtained.

The complexity of FINAL_SEQUENCE depends on the complexity of SEQUENCE and the number of times it is called. The number of times SEQUENCE is called is equal to the total number of S^i's intervals $(i=0,1,\ldots,k\text{-}1)$. Let $|S^i|$ denotes number of S^i intervals. We show now that $|S^i| \le M_i + |S^{i-1}| = O(i * M_i)$.

After the sequence of S^i's is created, $t_i^*(S_j^i)$ is strictly decreasing except at some $S_a^i = S_l^{i-1}$, where, $t_i^*(S_{a-1}^i) \ge t_i^*(S_a^i)$. Therefore $|S^i| \le M_i + |S^{i-1}|$. But,

$$\begin{aligned} |S^1| &\le 1 * M_1 + 1 \\ |S^2| &\le M_2 + M_1 + 1 \le 2 * M_2 \\ |S^i| &\le M_i + (i\text{-}1)\, M_{i-1}, \text{ for } i > 1. \\ &\le M_i + (i\text{-}1)\, M_i \\ &\le i * M_i \end{aligned}$$

therefore $|S^i| = O(i * M_i)$, for $i > 1$.

The complexity of SEQUENCE$(S_j^{i-1}, S_{j+1}^{i-1})$ is $O(M_{i-1} * m_i + M_i)$, and the (number of S^{i-1} intervals) $= O((i\text{-}1) * M_{i-1})$. Therefore the time required to compute critical spreads for level i is $O((i\text{-}1) * (M_{i-1}^2 * m_i + M_{i-1} * M_i))$, and the complexity to establish the tradeoff graph is at most,

$$C * \sum_{i=1}^{k-1} i * (M_i^{\,2} * m_{i+1} + M_i * M_{i+1}) \le C * k * n^2 \sum_{i=1}^{k-1} (m_{i+1} + 1) \le C * (k * n^3 + k^2 * n^2)$$

Without lost of generality, $n \ge k$, i.e. there is at least one net in each channel. Therefore the complexity is $O(kn^3)$. Q.E.D.

Concluding remarks:

We have presented a polynomial time algorithm to solve the multiple channel single component river routing problem. It is a dynamic programming approach which allows incremental updating of the solution from previous solutions. This type of algorithm is especially desirable for interactive VLSI design systems. In an interactive design environment, designers would like to see the compacted layout right after a new component is added to the design. Our algorithm provides such incremental updating capability.

Another feature of the algorithm is that placement of components and routing can be done independently yet a legal placement ensures routability. This feature is also very desirable for interactive VLSI design systems. There can be as many as $O(n^2)$ wire segments for a design of n nets. Laying them out can be time consuming, and displaying them can clutter a design. Since the routability condition is insured, a router can be called at anytime during the design if needed. Whenever area is the primary concern, designers have no interest in knowing the appearance of the wires as long as routability is ensured.

A natural extension of our work is to consider rows containing more than one component. However Pinter has shown that the problem for multiple components per row is NP-Complete [10]. Thus the problem for slicing structured layout with one layer routing is also NP-Complete. Therefore, for the more general slicing structures, we wish to find approximate solutions which ensure routability and give good compactions. This problem is open.

We have considered only one layer interconnections between adjacent components. Another open problem is to find similar algorithms for more general routing models. Since more general routing is generally NP-Complete, we expect these algorithms to be approximation algorithms using measures of channel width such as density.

Acknowledgement:

We would like to thank Arvin Park for his comments and suggestions regarding the organization of this paper.

References:

[1] Danny Dolev, Kevin Karplus, Alan Siegel, Alex Strong, Jeffrey D. Ullman: "Optimal Wiring Between Rectangles", *Proceedings of 13th ACM Symposium on Theory of Computing, pp 312-317*

[2] Gershon Kedem, Hiroyuki Watanabe: "Graph-Optimization Techniques for IC Layout and Compaction", *IEEE Transaction on Computer-Aided Design, Vol. CAD-3, No. 1, January 1984, pp 12-19.*

[3] Andrea S. LaPaugh: "Algorithms for Integrated Circuit layout: an analytic approach", *Ph.D. dissertation, 1980, Massachusetts Institute of Technology.*

[4] Ulrich Lauther: "A min-cut placement algorithm for general cell assemblies based on a graph representation", *16th Design Automation Conference, 1979, pp 1-9.*

[5] Charles E. Leiserson, Ron Y. Pinter: "Optimal Placement For River Routing", *SIAM J. Comput., Vol 12, No 3, August 1983, pp 447-462.*

[6] F. Miller Maley: "Compaction with Automatic Jog Introduction", *1985 Chapel Hill Conference, pp 261-283.*

[7] José Monteiro da Mata: "A Methodology for VLSI Design and a Constraint-based Layout Language", *Ph.D. dissertation, October 1984, Princeton University.*

[8] A. Mirzaian: "Channel Routing in VLSI", *Proceedings of 16th ACM symposium on Theory of Computing, pp101-107.*

[9] Ralph H.J.M. Otten: "Efficient Floorplan Optimization", *International Conference on Computers and Design, pp 499-502.*

[10] Ron Y. Pinter: "The Impact of Layer Assignment Methods on Layout Algorithms for Integrated Circuits", *Ph.D. dissertation, 1982, Massachusetts Institute of Technology.*

[11] Larry Stockmeyer: "Optimal Orientations of Cells in Slicing Floorplan Designs", *Information and Control 57, pp 91-101, 1983.*

[12] Antoni A. Szepieniec, Ralph H.J.M. Otten: "The Genealogical Approach To the Layout Problem", *17th Design Automation Conference, 1980, pp 535-542.*