Compressing Rectilinear Pictures and Minimizing Access Control Lists

DAVID A. APPLEGATE * GRUIA CALINESCU † DAVID S. JOHNSON ‡
HOWARD KARLOFF § KATRINA LIGETT ¶ JIA WANG ‖

July 5, 2006

Abstract

We consider a geometric model for the problem of minimizing access control lists (ACLs) in network routers, a model that also has applications to rectilinear picture compression and figure drawing in common graphics software packages. Here the goal is to create a colored rectilinear pattern within an initially white rectangular canvas, and the basic operation is to choose a subrectangle and paint it a single color, overwriting all previous colors in the rectangle. Rectangle Rule List (RRL) minimization is the problem of finding the shortest list of rules needed to create a given pattern. ACL minimization is a restricted version of this problem where the set of allowed rectangles must correspond to pairs of IP address prefixes. Motivated by the ACL application, we study the special cases of RRL and ACL minimization in which all rectangles must be strips that extend either the full length or the full height of the canvas (*strip-rules*). We provide several equivalent characterizations of the patterns achievable using strip-rules and present polynomial-time algorithms for optimally constructing such patterns when, as in the ACL application, the only colors are black and white (permit or deny). We also show that RRL minimization is NP-hard in general and provide $O(\min(n^{1/3}, \mathrm{OPT}^{1/2}))$ approximation algorithms for general RRL and ACL minimization by exploiting our results about strip-rule patterns.

* `david@research.att.com`. AT&T Labs – Research, Room C224, 180 Park Avenue, Florham Park, NJ 07932.

† `calinesc@iit.edu`. Computer Science Department, Illinois Institute of Technology, Stuart Building, Room 236, 10 West 31st Street, Chicago, IL 60616.

‡ `dsj@research.att.com`. AT&T Labs – Research, Room C239, 180 Park Avenue, Florham Park, NJ 07932.

§ `howard@research.att.com`. AT&T Labs – Research, Room C231, 180 Park Avenue, Florham Park, NJ 07932.

¶ `katrina+@cs.cmu.edu`. Department of Computer Science, Carnegie Mellon University, 5000 Forbes Avenue, Pittsburgh, PA 15213. Research supported in part by AT&T.

‖ `jiawang@research.att.com`. AT&T Labs – Research, Room A165, 180 Park Avenue, Florham Park, NJ 07932.

1 Background and Motivation

1.1 Rectilinear Pictures

Many of today's software packages that generate graphics, from `Xfig` to `Powerpoint`, share a common method for creating rectilinear patterns. Starting with a white rectangular canvas, the user repeatedly applies a "rectangle tool" with which one sweeps out a rectangular area and colors the interior with a specified color, overwriting any previous contents of that area. Indeed, most of the figures in this abstract were produced using such a tool. See for instance Figure 1, where the underlying canvas is a 4×4 grid, and a sequence of three rectangle operations is applied.

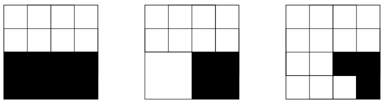

Figure 1: Creation of a figure using rectangle rule applications.

The sequence depicted is inefficient since two operations would have sufficed. Given the manual work involved in producing such figures, one might well want to determine efficient plans of action in advance. Call a pair (R, c), where R is a rectangle and c is a color, a *rectangle rule*, and a sequence of such rules $(R_1, c_1),\ (R_2, c_2), \ldots, (R_n, c_n)$ a *rectangle rule list* or RRL. For an RRL $\mathcal{R}$, let $P_{\mathcal{R}}$ denote the pattern it produces. Given a target pattern P, our goal is to find a minimum-length RRL $\mathcal{R}$ such that $P_{\mathcal{R}} = P$. Such an RRL also provides a scheme for picture compression that is potentially more effective than the classic scheme in which a black rectilinear figure is represented by a collection of black rectangles that covers it. Moreover, a restricted version of the problem has an important application to Internet management.

1.2 Access Control Lists

Access control lists (ACLs) are used in network router line cards to determine which arriving packets should be forwarded to their destination and which should be dropped. For instance, an Internet Service Provider (ISP) might want its access routers to forward only packets that came from or are destined to customers who have officially been assigned to that router. ACLs can also be used to implement firewalls and to provide a range of levels of quality of service [Cis01]. An ACL is made up of a sequence of *rules*. In an *extended ACL* a rule can be viewed as having five components: Source Range, Destination Range, Protocol, Port(s), and Action. The source and destination ranges are specified by binary strings s of length w or less (where w is the length of an IP address, currently 32 but expanding to 64 in IPV6). The string s matches all IP addresses that have it as a prefix; an empty string matches everything. Possible protocols include IP, TCP, UDP, and ICMP, with IP matching all protocols and the others matching only packets labeled by that particular protocol. Ports can either be an individual port number, a range of such numbers, or "any," which matches everything. The action can either be *permit*, which allows the packet through, or *deny*, which causes the packet to be dropped. A *basic ACL* omits the protocol and port fields.

An ACL operates as follows: When a packet arrives, the router determines the first rule in the list that it matches and performs the action specified by that rule. If there is no match, the packet is dropped. In high-speed routers the classification is performed by special hardware called "Ternary CAMs" (TCAMs) that evaluate all the rules in parallel and output the lowest indexed match. These are expensive and impose limits on the size of ACLs, as do the overall memory constraints in a line card. Thus a natural optimization criterion for ACLs is to minimize their length while preserving the results of their actions. This will also reduce the maximum delay in routers that evaluate rules sequentially.

Other optimization criteria have also been studied. Many researchers have studied data structures and algorithms for quickly determining which rule in an ACL is the first match [GM99, EM01, KMT03, Tho03] or, when information about the data traffic is known in advance, trying to minimize the *average* time to find the applicable rule, either simply by reordering the ACL [Ful05] or by devising sophisticated decision tree classifiers [CL05]. None of these approaches have yet been implemented in real-world routers, however. Although such approaches can be of value when using software simulation to study router behavior, for now ACL minimization remains the most direct way to improve the access control performance of current real-world routers. AT&T's interest in ACL minimization was the inspiration for this paper.

In modeling the ACL minimization problem we can simplify matters by ignoring the protocol and port restrictions. These fields are not present in the simpler basic ACLs that are still sometimes used, and even within extended ACLs, most rules (85% or more of the rules in ACLs studied by Cohen and Lund [CL05] and similar proportions in access router ACLs we ourselves sampled) are basic in that the entries for protocol and port match everything. The more restrictive rules, even when present, are likely to have priority over the basic ones or be independent of them, so that optimizing over the basic rules would at least be a component of an effective heuristic for the general problem. Thus in what follows we shall assume that our ACLs are restricted to basic rules.

The problem of ACL minimization can be modeled in geometric terms. Consider a $2^w \times 2^w$ grid with a cell for each combination of a source and destination IP address (columns and rows indexed from 0 to $2^w - 1$). The action of an ACL can be viewed as coloring the cells of this grid, with the cell colored white if packets with that combination are denied, and black if they are permitted. Each rule applies to the rectangle in this grid whose x-coordinates are the IP addresses in the rule's source range and whose y-coordinates are those in the destination range. Thus each rule in an ACL is once again a pair (R, c), where R is a rectangle and c is a color, just as in our RRL application.

There are two distinctions between ACLs and RRLs, one minor and one major. The minor distinction is that the rules in an ACL are in the reverse order from those in an RRL. The major distinction is that the set of possible rectangles is highly constrained in ACLs since their x- and y-coordinates are determined by IP address prefixes. The widths and heights of rectangles must all be of the form 2^k for $0 \le k \le w$, and if a rectangle's width (height) is 2^k, then it must start in a column (row) whose coordinate is congruent to 0 mod 2^k. Note that this means that the projections of the rectangles on the axes are *laminar* in that for any two rectangles the projections are either disjoint or one is contained in the other.

1.3 Outline

The rest of this paper is organized as follows. In Section 2 we discuss previous results related to RRL and ACL minimization. In Section 3 we introduce and motivate the special case of *strip-rule* patterns. In ACL terms, this is the commonly encountered case in which no single rule restricts both the source and the destination address. We characterize the patterns that can be constructed with such rules and show that they can be efficiently recognized. In Section 4 we present polynomial-time algorithms for constructing optimal strip-rule RRLs and ACLs for 2-color strip-rule patterns. In Section 5 we show that RRL minimization for general 2-color patterns is NP-hard by exploiting insights into the strip-rule special case. In Section 6 we use subroutines for the strip-rule special case to build polynomial-time approximation algorithms for general RRL and ACL minimization with worst-case ratios of $O(\min(n^{1/3}, \mathrm{OPT}^{1/2}))$ and $O(w^2 \min(n^{1/3}, \mathrm{OPT}^{1/2}))$ respectively, currently the best guarantees known to be attainable in polynomial time. We conclude in Section 7 by mentioning some additional results and open problems.

2 Previous Results

We know of no previous theoretical work on the RRL/ACL minimization problems as formulated here, but several special cases have previously been studied. The one-dimensional case where only the destination is restricted corresponds to the problem of minimizing routing tables, where under most routing protocols, the link out which a packet is sent depends only on its destination address. Now our rules correspond simply to intervals, rather than rectangles. The one-dimensional ACL problem can be solved in polynomial time by dynamic programming [DKVZ99, SSW03]. This holds even when $K > 2$ colors are allowed, as is the case with routing tables. If the pattern is specified by an ACL of length n using length-w IP addresses, the algorithm of Suri et al. [SSW03] produces an optimal equivalent ACL in time $O(Knw)$. Dynamic programming also can be used to minimize one-dimensional RRLs. We know of no formal references, but the problem was given with a solution as part of a June 2003 `TopCoder` programming contest under the name `StripePainter` [TC03] and it is not difficult to see how to attain a $O(Kn^3)$ running time bound.

Returning to two dimensions, the case of RRL minimization in which only black rectangle rules are allowed has been well-studied under the name RECTILINEAR PICTURE COMPRESSION. It was shown NP-Hard by Masek [Mas78] and MaxSNP-hard by Berman and DasGupta [BD97]. The best polynomial-time achievable approximation guarantee known is $O(\sqrt{\log n})$, where n is the number of black grid cells in the pattern [AR99]. In contrast, the optimal all-black ACL for a given pattern can be found in polynomial time. This was shown by Lakshmanan et al. in a database context [LNW$^+$02]. The key insight is that when the projections of the allowed rectangles on both axes are laminar in the sense defined at the end of Section 1.2, the ACL consisting of all the maximal black subrectangles in the pattern has minimum length.

Unfortunately, the optimal all-black RRL/ACL can be a very poor approximation to the optimal black-and-white one. Consider a $(2n-1) \times (2n-1)$ grid in which the only black cells are those both of whose coordinates are odd. The optimal RRL/ACL consists of one black rule covering the whole pattern, overwritten by $n-1$ white column rules and $n-1$ white row rules, for a total of $2n-1$ rules, whereas the best all-black-rule list contains n^2 rules, yielding a ratio that is $\Theta(\mathrm{OPT}(P))$.

3 Strip-Rule Patterns: Definitions and Characterizations

In examining a sample of thousands of extended ACLs from AT&T's access routers, we observed that 95% of these lists contained no rule that restricted both source and destination IP addresses. In other words, in the vast majority of ACLs, every rule was a *strip-rule*, that is, one that colored either a set of contiguous full columns or a set of contiguous full rows of the grid. We refer to ACLs/RRLs made up entirely of strip-rules as *strip-rule ACLs/RRLs* and the patterns they generate as *strip-rule patterns*. (Note that the set of patterns generated by strip-rule ACLs is identical to the set of patterns generated by strip-rule RRLs.)

Not all patterns are strip-rule patterns; for example, the simple 4-cell checkerboard pattern, with the lower left and upper right cells black and the other two white, is not. Indeed, every black-and-white non-strip-rule pattern must contain such a checkerboard as a subarray. There are in fact many equivalent characterizations of strip-rule patterns, as illustrated by the following two theorems.

Theorem 3.1 *Suppose P is a 2-color pattern. The following statements are equivalent.*

1. *P is a strip-rule pattern.*
2. *P contains no checkerboard subpattern; that is, every 2×2 (not-necessarily-contiguous) subarray of P contains a monochromatic row or column.*
3. (Monotonicity Property). *For any color C the rows of P are hierarchical: For any two rows, the set of columns where C is present in the first row either contains or is contained in the set of columns where C is present in the second.*
4. *The monotonicity property holds with the roles of row and column reversed.*
5. *The following "Pick-Up-Sticks" algorithm always results in an empty grid: Let* gray *be a color not used in P. Call a row or column* pseudo-monochromatic *if it contains at most one color other than gray. Call the grid* empty *if all its cells are colored gray. Perform the following loop: While there is a pseudo-monochromatic row or column containing a cell with a non-gray color, choose such a row or column and color all its cells gray ("pick it up").*

1	3	1
3	3	2
1	2	2

Figure 2: A non-strip-rule pattern for which every 2×2 subarray has a monochromatic row or column.

The proof of Theorem 3.1 is relatively straightforward and is included in the Appendix. Things get more complicated when we consider patterns P having more than 2 colors. For example, property (2) is no longer a characterization, as can be seen from Figure 2, which shows a 3×3 pattern with three colors that has no monochromatic rows or columns, but each of whose 2×2 submatrices *does* have a monochromatic row or column. Surprisingly, this is the worst counterexample possible, as we have the following general characterization theorem, also proved in the Appendix.

Theorem 3.2 *Suppose P is a k-color pattern, $k > 2$. The following statements are equivalent.*

1. *P is a strip-rule pattern.*
2. *Every 2×2 and 3×3 subarray of P contains a monochromatic row or column.*
3. *The "Pick-Up-Sticks" algorithm always results in an empty grid.*

The forbidden subarray characterizations in Theorems 3.1 and 3.2 are of combinatorial interest, but are not the key to fast recognition of strip-rule patterns. For that we can simply run the Pick-Up-Sticks algorithm. By the characterization theorems, P is a strip-rule pattern if and only if the algorithm succeeds in reducing it to the empty grid, and the Pick-Up-Sticks algorithm can be implemented to run in $O(n^2)$ time ($O(n^2 w)$ for ACLs), where n is the length of an RRL/ACL that produces P, or is the maximum dimension of the *effective grid* for the pattern, defined as follows.

For any rectilinear pattern P, call a horizontal or vertical line segment that separates and bounds two differently-colored regions a *boundary line*. Extend each boundary line in both directions so that it crosses the full canvas. We call the resulting grid the "effective grid" for the pattern, and note that all its grid cells will be monochromatic. Moreover, if the pattern is generated by an RRL/ACL with n rules, then the number of (internal) grid lines is at most $4n$ (or $2n$ if it is a strip-rule RRL/ACL).

In what follows it will be convenient to consider the variant on Pick-Up-Sticks we shall call *Maximal Pick-Up-Sticks* (MPUS), where instead of graying individual rows and columns we gray maximal contiguous sequences of rows and columns that collectively contain precisely one non-gray color. (In the case of ACLs we are further restricted to maximal contiguous sequences that yield legal strips.) We halt as soon as the only colors left in the pattern are gray and the background color (white). Like Pick-Up-Sticks this is not a well-defined algorithm – we don't say how to choose the next operation when more than one are available. However, even without specifying the tie-breaking rule we do get the following results. For a strip-rule pattern P, let $\mathrm{OPT}_S(P)$ denote the length of the minimum-length strip-rule RRL/ACL that generates P (it will always be clear from context whether we are talking about RRLs or ACLs).

Theorem 3.3 *For any strip-rule pattern P,*

1. *All RRLs generable for P using MPUS are no longer than $2\mathrm{OPT}_S(P) + 1$, and.*
2. *All ACLs generable for P using MPUS are no longer than $w\mathrm{OPT}_S(P) + 3$.*

These bounds hold for any number of colors. The bound for RRLs follows from the fact that any maximal rule except possibly the last must reduce the number of effective grid lines by 1. The bounds cannot be significantly improved, as we show in the Appendix.

4 2-Color Strip-Rule Patterns: Constructing Optimal RRLs and ACLs

In this section we sketch our polynomial-time algorithms for constructing optimal strip-rule RRLs and ACLs for 2-color strip-rule patterns. We shall assume that the pattern is given by its effective grid. It is not difficult to see that finding an optimal RRL/ACL is equivalent to finding an optimal ordering of operations for MPUS. In what follows, say that a column/row is "picked-up" by a rule if it is covered by the rule and was not all gray at the time the rule was applied. Hence no column/row can be picked up more than once. We exploit the fact that the rows and columns of a pattern P can be divided into equivalence classes, with two columns (rows) being equivalent if their corresponding cells are identically colored in the original pattern P. For example, see Figure 3. Here the column equivalence classes are all-black columns (C_1), black-above-white columns (C_2), and all-white columns (C_3). Each of the row equivalence classes has a single member, with R_1 consisting of the top row and R_2 consisting of the bottom row. The following observations are easy to verify for any 2-color strip-rule pattern P.

1. In any MPUS rule, all the rows/columns that are picked up are in the same equivalence class.

2. If some member of an equivalence class is picked up by a rule of color c, then *all* members of that class must eventually be picked up by c-rules.

3. Every class is either picked up by a white rule, picked up by a black rule, or not picked up, and there is at most one row class and one column class in the third category.

4. For an equivalence class E, let $N_w(E)$ be the number of white cells its members contain in P and $N_b(E)$ the number of black cells. Then no two distinct column (row) equivalence classes have the same value of $N_w(E)$ or $N_b(E)$ (this follows from the monotonicity property in Theorem 3.1).

5. For any color c, no member of a class E can be picked up by a c-rule until all members of classes E' of the same type (row or column) with $N_c(E') > N_c(E)$ have also been picked up by c-rules.

6. At any point in the MPUS process, precisely two equivalence classes will be pseudo-monochromatic and hence ripe for pick-up (by black and white column rules, black and white row rules, black row and column rules, or white row and column rules).

In light of the above, any RRL/ACL created by MPUS can be divided into a sequence of segments, each segment containing all the rules applied when a given pair of equivalence classes were pseudo-monochromatic. We can associate a 5-tuple label (E_1, c_1, E_2, c_2, U) with each such segment, whose entries are specified as follows: E_1 and E_2 are the two monochromatic classes and c_1 and c_2 are the colors of their non-gray cells just before the segment was applied. E_1 is the "newer" of the two classes in that it is the one that was not pseudo-monochromatic during the previous segment. The final parameter, U, is the set consisting of those members of E_2 that had not yet been picked up when the segment started. (Note that

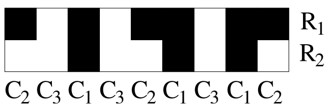

Figure 3: Pattern with rows and columns labeled by their equivalence class.

none of the members of E_1 can have been picked up earlier because this class was not previously pseudo-monochromatic.) The fact that some members of E_2 may already have been picked up in earlier segments, even in an optimal RRL/ACL, follows from the observation that the best way to pick up one equivalence class may sometimes involve picking up members of the other. This is illustrated by the pattern in Figure 3. Recall that an RRL must reduce the pattern to one in which all cells are either gray or (the background color) white. Here the initially available classes are C_1 (all-black columns) and C_2 (all-white columns). Any RRL that picks up all of C_1 before any member of C_2 or vice versa will require at least six rules, whereas there is an RRL of length five for the pattern that finishes C_1 first but picks up one member of C_2 in doing so.

In addition to a label, each segment has an *action* $i \in \{1, 2\}$, which is the index (from the segment's label) of the equivalence class that gets completely picked up as a result of the segment.

Our algorithms exploit a pair of normal-form lemmas. Suppose P is the target pattern, L is an RRL/ACL for P, and S is a segment of L. Define $pattern(P, L, S)$ to be the pattern that existed after all the rules in segments of L preceding S have been applied, i.e., P as modified by turning gray all columns and rows covered by rules in those segments.

Lemma 4.1 *Suppose L and L' are two RRLs/ACLs for a given pattern P, and S and S' are segments of L and L' respectively whose associated 5-tuples are identical. Then $pattern(P, L, S) = pattern(P, L', S')$.*

This is a consequence of property (5) above and additional arguments about the relationship between row and column equivalence classes that are picked up with different colors.

4.1 RRL-Specific Details

The next normal-form lemma applies only to RRLs. There is an analogous lemma for ACLs but it is considerably more complicated to state and we will postpone it until we have finished describing the algorithm for RRLs. Call a pseudo-monochromatic row/column *active* if it contains at least one non-gray cell. Call a sequence of contiguous rows/columns a color-c *block* if it only contains cells that are gray or color-c, it contains at least one active column/row, and it is maximal with respect to this property.

Lemma 4.2 *Suppose a minimum-length RRL for P contains a segment with label (E_1, c_1, E_2, c_2, U) that picks up all active members of class E_i for some $i \in \{1, 2\}$. Then there is a minimum-length RRL for P containing a segment with the same label and action that has the following structure:*

1. *If $c_1 = c_2$, then one class consists of rows and the other of columns. Assume that E_i is a column class. (The row case is handled analogously.) The segment is then a sequence of rules that pick up all the color-c_2 column blocks in the pattern associated with the given label, one rule per block.*

2. *If $c_1 \neq c_2$, then both classes are column classes or both are row classes. Assume that we are dealing with column classes and $i = 1$. (The other cases are again handled analogously.) Call a color-c_2 block in the pattern associated with the given label* embedded *if the columns to its left and right are active color-c_1 columns. Start with rules that pick up all the embedded blocks, one rule per block. Then pick up all the (possibly newly-created) color-c_1 blocks, one rule per block.*

The first case holds because column and row rules that are adjacent in a segment and have the same color can be interchanged without affecting the resulting pattern. The second case is more complicated, but essentially follows from the fact that the prescribed sequence uses the minimum possible number of rules to pick up all of E_i, and at the same time picks up a maximum set of columns from the other class "for free."

We can exploit Lemmas 4.1 and 4.2 to construct a minimum-length RRL by running a Dijkstra-like shortest path algorithm on the following (implicitly constructed) *segment graph* for P. The vertices of our graph are the 5-tuples that can occur in an RRL for P, with the "source" vertex being the 5-tuple

$(F_1, c_1, F_2, c_2, F_2)$, where the F_i are the two monochromatic classes in P and the c_i are their colors. Note that since both classes are new in this case, we can fix an arbitrary order for them. The "sink" vertices are those 5-tuples that correspond to patterns in which no black cells remain. Each non-sink vertex v has two out-arcs, one for each of the possible actions. The out-arc for action i corresponds to the Lemma 4.2 normal-form segment for action i and label v and leads to the 5-tuple that results when one applies that segment to the pattern corresponding to v. The length of the arc is the number of rules in the normal-form segment. Our goal is to find the shortest path from the source vertex to a sink vertex.

One can show that the segment graph is layered, so that we can evaluate vertices in breadth-first order. The running time will be proportional to the number of vertices encountered multiplied by the time to construct the normal form segments for each. Unfortunately, the number of vertices is not *a priori* polynomially bounded since we might have to consider labels with all $2^{|E_2|} - 1$ possible nonempty subsets of E_2 as candidates for U. The next lemma shows that this is fortunately not the case.

Lemma 4.3 (Containment Lemma) *If vertices $v = (E_1, c_1, E_2, c_2, U)$ and $v' = (E_1, c_1, E_2, c_2, U')$ are both reachable from the source vertex in the segment graph for P and $U \neq U'$, then either $U \subseteq U'$ or $U' \subseteq U$.*

Thus there can be at most $|E_2|$ reachable labels of the form (E_1, c_1, E_2, c_2, U) for given values of E_i and c_i. A careful analysis, taking into account the fact that the column classes are all disjoint, as are all the row classes, then leads to the conclusion that the total number of reachable states is $O(n^2)$ where n is the number of effective grid lines. The overall running time for the algorithm can then be shown to be $O(n^3)$.

Proof of Lemma: The proof is by induction on the depth of the vertices in the graph. The lemma clearly holds for depth 0, since the source vertex is the only vertex at this depth. Suppose it holds for depth $d \geq 0$ and that $v = (E_1, c_1, E_2, c_2, U)$ and $v' = (E_1, c_1, E_2, c_2, U')$ are vertices at depth $d + 1$. We may assume that neither U nor U' is the full class E_2, as otherwise the desired conclusion would hold trivially.

The first case we consider is when E_1 and E_2 are both column classes or both row classes, in which case we must have $c_1 \neq c_2$. Assume for specificity that they are column classes. The reader may verify that by Lemma 4.2 the only possible arcs into v and v' come from the vertices (E_1', c_2, E_2, c_2, U) and $(E_1', c_2, E_2, c_2, U')$, respectively, where E_1' is the row equivalence class previously picked up with color c_2 that has the fewest cells of color c_2. But then, by induction either $U \subset U$ or $U' \subset U$, as desired.

So suppose that one of E_1 and E_2 is a column class and the other is a row class, in which case $c_1 = c_2$. Assume for specificity that E_2 is the column class. Arcs into these two vertices can be from vertices of only two types, either (E_1', c, E_2, c_2, X) (*Type 1*) or (E_2, c_2, E_1', c, Y) (*Type 2*), where c is the other color from c_2 and E_1' is the column class previously picked-up with color c that has the fewest cells of color c. Since v and v' are reachable from the source vertex, there must be at least one arc into each from a reachable vertex of depth d. Let a be such an arc into v and a' be such an arc into v' and let x and x' be the tail vertices for a and a' respectively. We break into cases depending on the types of x and x'.

First, assume that they are both of Type 1 and denote them as $x = (E_1', c, E_2, c_2, X)$ and $x' = (E_1', c, E_2, c_2, X')$, where $U \subseteq X$ and $U' \subseteq X'$ since the actions applied to x and x' could not have enlarged the set of active columns in E_2. By our induction hypothesis we know that either $X \subseteq X'$ or $X' \subseteq X$. Assume without loss of generality that $X \subseteq X'$. We claim that this implies that $U \subseteq U'$. Let P_x and $P_{x'}$ be the patterns corresponding to x and x' respectively, and assume, again without loss of generality, that c_2 is black. Each of the normal-form segments corresponding to a and a' picks up all the black blocks in their pattern that are bounded on both sides by active columns from E_1', i.e., the embedded black blocks. For an active black column to survive to be included in U, it must be the case that on at least one side of it there is no active white column between it and the boundary or between it and a non-pseudo-monochromatic column. Turning some set of all-gray columns into active black columns, as one does to $X' - X$ when one goes from x to x', will not change this fact. Thus all active columns that survive to be included in U will also be included in U' and $U \subseteq U'$ as desired.

Next, assume that x and x' are both Type 2 vertices and denote them as $x = (E_2, c_2, E'_1, c, Y)$ and $x' = (E_2, c_2, E'_1, c, Y')$. By the induction hypothesis we know that either $Y \subseteq Y'$ or $Y' \subseteq Y$. We can assume without loss of generality that $Y \subseteq Y'$. We claim that this implies that $U' \subseteq U$. Once again assume c_2 is black. For an active black column to survive to be included in U', it must again be the case that on at least one side of it there is no active white column between it and the boundary or between it and a non-pseudo-monochromatic column. Turning gray some set of active white columns, as one does to $Y' - Y$ when one goes from x' to x, will not change this fact. Thus all active columns that survive to be included in U' will also be included in U and $U' \subseteq U$ as desired.

Finally assume one of x and x' is of Type 1, say x, and the other is of Type 2, in which case we can denote them as $x = (E'_1, c, E_2, c_2, X)$ and $x' = (E_2, c_2, E'_1, c, Y)$. Consider the potential vertex $x'' = (E'_1, c, E_2, c_2, E_2)$, which corresponds to the same pattern $P_{x''}$ as does $(E_2, c_2, E'_1, c, E'_1)$, and consider the result of applying the normal-form segment that picks up all members of E'_1, which we can denote by $(E_1, c_1, E_2, c_2, U'')$. Then, since $X \subseteq E_2$ we have by the argument in the first case above that $U \subseteq U''$. On the other hand, since $Y \subseteq E'_1$ we have by the argument in the second case that $U'' \subseteq U'$. Hence $U \subseteq U'$, as needed. Thus the desired conclusion holds in all three cases and by induction the lemma follows. ■

4.2 ACL-Specific Details

Our algorithm for finding optimal strip-rule ACLs is much more complicated than that for RRLs, although the basic structure is the same. The key step is to identify the normal form for a segment when the two pseudo-monochromatic classes are either both rows or both columns. Suppose we are dealing with columns.

In an ACL, a column strip-rule can be identified with an IP prefix of length w or less. Call a strip-rule *undominated* in a segment if none of its proper prefixes is also a rule in that segment. Note that the set of undominated rules determines precisely which active columns get picked up. Define the *signature* for a segment to be a $(w + 1)$-tuple $(x_0, x_1, x_2, \ldots, x_w)$, where x_i is the number of prefixes of length i that are undominated rules for the segment. As an example, suppose $w = 5$ and the strip-rule prefixes for a segment are 001, 0010, 011, 10001, 11, and 1. The the undominated prefixes are 001, 011, and 1, and the signature is $(0, 1, 0, 2, 0, 0)$. Given two different signatures, consider the first component in which they disagree. We say that the signature with the larger value in this component is the *stronger* of the two. To determine the normal form for a segment with a given label (E_1, c_1, E_2, c_2, U) and action i, consider all minimum-length lists of rules that pick up all the active members of E_i, and choose one with the strongest signature. Such a list can be found by dynamic programming in $O(wn)$ time, and we can prove that any such list must have the same set of undominated rules. Full details of the ACL algorithm, along with proofs of the appropriate analogues of Lemmas 4.2 and Containment Lemma (Lemma 4.3), are given in the Appendix. The result extends to any restricted class of rules whose projections on the horizontal and vertical axes form laminar families.

5 NP-completeness for 2-color RRLs

In this section we provide a brief overview of our proof that the problem of constructing a minimum-length (general) RRL is NP-hard. The proof is by a transformation from RECTILINEAR PICTURE COMPRESSION. Suppose we are given a rectilinear black pattern P on a grid with a white background and asked if there is a collection of k or fewer black rectangles whose union is the pattern. We will construct a rectilinear black and white pattern P' and an integer k' such that P' can be generated by a length-k' RRL if and only if P can be represented as the union of k or fewer black rectangles.

If we could somehow restrict the RRL to use only black rules, we could simply use $P' = P$. Unfortunately, as we have seen, white rules can make a difference and cannot be ignored. Our solution is to embed

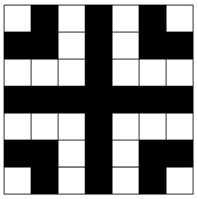

Figure 4: Tile in NP-completeness proof for RRLs.

P into a much bigger pattern which will effectively eliminate the usefulness of white rules in constructing P. To do this, we design a background for the figure that must be picked up by strip-rules if the RRL is to be sufficiently short. The background is a large grid constructed out of multiple copies of the 7×7 tile depicted in ure 4, for which the minimum-length RRL is a strip-rule RRL with a specific structure. The construction is relatively straightforward, but the argument that it works is quite intricate because of all the flexibility that two-color rules provide. The full proof is included in the Appendix.

6 Approximation Algorithms for Unrestricted Patterns

Given that the general problem of RRL minimization is NP-hard and the general ACL problem may be NP-hard as well, it is reasonable to look for good approximation algorithms. The only previously-studied algorithms that can be viewed as approximation algorithms for these problems are the ones that construct RRLs/ACLs using only black rules. Unfortunately, as observed in Section 2, even the best possible ACL/RRL consisting only of black rules can be off by a factor of $\mathrm{OPT}(P)$ for patterns where $\mathrm{OPT}(P) = \Theta(n)$.

Our new algorithms, which work for arbitrary numbers of colors and use Maximal Pick-Up-Sticks as a subroutine, represent a substantial improvement. In the RRL case, our algorithm, which we call the *Iterated Strip-Rule* algorithm (ISR), works as follows. We assume we are given the pattern by a length-n RRL that generates it. Suppose the effective grid for the pattern has N_{col} columns and N_{row} rows and that $N_{col} \geq N_{row}$. (If not, we can simply rotate the pattern 90 degrees.) Note that $\mathrm{OPT}(P) \geq \lfloor N_{col}/2 \rfloor$ since no rule can introduce more than two new effective vertical grid lines. Our basic approach is to partition P into subrectangles that are all strip-rule patterns, each of which can be handled by MPUS. We do this for several different partitions, and take the best result, unless all are worse than the input RRL, in which case we return the latter.

More formally, for each integer q, $1 \leq q \leq N_{col}$, we do the following. First we partition the grid into q vertical strips of widths as close to N_{col}/q as possible. Then we partition each strip into a sequence $B_1, B_2, \ldots$ of blocks, starting from the bottom. The odd-indexed blocks (the *good* blocks) are subrectangles that extend to the full width of the strip and as far up as they can while still constituting a strip-rule pattern. Each good block that does not extend all the way up to the top boundary of P is followed by a *bad* block that is a height-1 subrectangle immediately above the good block, extending the full width of the strip. (Note that such rectangles are trivially strip-rule patterns.) Let $\mathcal{R}_q$ be an RRL constructed by using MPUS to handle each of the blocks separately. (Strips for a block will typically be rectangles in terms of the overall pattern.) Our output is the best of the $\mathcal{R}_q$ RRLs.

Theorem 6.1 *The iterated strip-rule algorithm runs in polynomial time and produces an RRL whose length is* $O(\mathrm{OPT}(P)\min(n^{1/3}, \mathrm{OPT}^{1/2}))$. *Moreover, the bound is tight in that there exist patterns with arbitrarily large values of* $\mathrm{OPT}(P) = \Theta(N_{col})$ *for which the algorithm produces RRLs whose lengths are* $\Theta\left(\mathrm{OPT}(P)^{3/2}\right)$.

Key to the proof are the observations that (1) the RRL created by MPUS for a $p \times q$ rectangle uses at most $p + q$ rules, (2) the union of each bad block with its preceding good block contains either a 2×2 or

3×3 subarray with no monochromatic rows or columns, and (3) the following lower bound lemma.

Denote the grid cells by $P_{i,j}$, $1 \le i \le N_{col}$ and $1 \le j \le N_{row}$. If $1 \le h \le i \le N_{col}$ and $1 \le j \le k \le N_{row}$, let $R(h,i,j,k)$ be the subrectangle of the grid formed by the intersection of columns h through i with rows j through k. If $1 \le h \le i \le N_{col}$ and $1 \le j \le k \le N_{row}$, subrectangle $E(h,i,j,k)$ is a *forbidden rectangle* if it contains a 2×2 or 3×3 subarray involving its top and bottom rows and left and right columns, such that this subarray has no monochromatic row or column.

Lemma 6.2 *Suppose there are r disjoint forbidden rectangles. Then $OPT(P) \ge r/4$.*

Proof. Let $\mathcal{R}$ be a collection of r disjoint forbidden rectangles, and let E be any rectangle in $\mathcal{R}$. Note that by definition of forbidden rectangle, all the bounding rows/columns of E (the top and bottom rows and the leftmost and rightmost columns) are non-monochromatic in P. Let A be an optimal RRL for P, which we shall view in pick-up-sticks order. The first rule in A to include a corner of E can include only one of its four corners, as otherwise a bounding row/column of E would be monochromatic in P. This means that the rectangle corresponding to the rule must have one of its corners in E. Since all the rectangles in $\mathcal{R}$ are disjoint and nonmonochromatic, every one must be intersected by at least one rule. Since no rule has more than four corners, we conclude that $r \le 4|A| = 4 \cdot OPT(P)$, and the claim follows. ■

The full proof of Theorem 6.1, including lower bound examples, in given in the Appendix. Note that to get an approximation algorithm for ACLs, one merely has to replace each rectangle in the RRL solution by the minimum set of ACL rules needed to generate it, yielding an $O(w^2 \mathrm{OPT}(P) \min(n^{1/3}, \mathrm{OPT}^{1/2}))$ worst-case bound.

7 Additional Results and Open Problems

In much of this paper we have concentrated on strip-rule RRLs and ACLs for strip-rule patterns. However, in some applications one might be able to use arbitrary RRLs/ACLs for such patterns, in which case one might ask about the performance penalties imposed by the restriction to strip-rule RRLs and ACLs. Fortunately the penalty is not too severe, at least in the case of RRLs.

Theorem 7.1 *For RRLs and for any strip-rule pattern P, $\mathrm{OPT}_S(P) \le 4 \cdot \mathrm{OPT}(P) + 1$.*

This follows from the fact that no rectangle rule can introduce more than 4 effective grid lines and every strip-rule must eliminate at least 1, with the possibility of one final rule to color any cells remaining after all effective grid lines have been eliminated. The asymptotic bound is at least 3.5, based on a sequence of patterns P_k presented in the Appendix that have $\mathrm{OPT}(P_k) = 2k + 2$ and $\mathrm{OPT}_S(P_k) = 7k + 6$. It would be interesting to close this gap, as well as that for ACLs, where we get upper and lower bounds proportional to the IP address length w.

At present, however, the main open problem concerns the complexity of general ACL minimization. Our NP-completeness result for RRL minimization does not preclude the fact that ACL minimization might be in P, and the corresponding problem for black rules only *is* in P, as we have already mentioned, although the techniques used do not extend to the 2-color case.

Another open problem is that of improving on the approximation algorithms of the previous section for the case of arbitrary patterns, both for RRLs and ACLs. We can prove that the greedy heuristic that repeatedly picks up the pseudo-monochromatic rectangle with the most non-gray cells has worst-case examples that are at least as bad as those for ISR. Are polylogarithmic guarantees possible in polynomial time, as they are for the black-rule-only case? Are additional $\Omega(w)$ factors necessary in the guarantees for ACLs?

Finally, can our strip-rule optimization algorithms be extended to more than 2 colors? This appears likely for the ACL case, but in the RRL case, once one gets beyond 2 colors, potential normal forms for segments of an optimal RRL appear to be much more complex.

References

[AR99] V. S. Anil Kumar and H. Ramesh. Covering rectilinear polygons with axis-parallel rectangles. In *Proc. 31st Ann. ACM Symposium on Theory of Computing*, pages 445–454, New York, NY, 1999. ACM.

[BD97] P. Berman and B. DasGupta. Complexities of efficient solutions of rectilinear polygon cover problems. *Algorithmica*, 17:331–356, 1997.

[Cis01] Cisco Systems. Controlling network access with access control lists (Chapter 10). In *Catalyst 6500 Series Switch and Cisco 7600 Series Firewall Services Module Configuration Guide*, 2001. Available at `http://www.cisco.com/univercd/cc/td/doc/product/lan/cat6000/mod_icn/fwsm/fwsm_2_2/fwsm_cfg?mngacl.pdf`.

[CL05] E. Cohen and C. Lund. Packet classification in large ISPs: Design and evaluation of decision tree classifiers. In *Proc. ACM Sigmetrics '05*, pages 73–84, 2005.

[DKVZ99] R. Daves, C. King, S. Venkatachary, and B. Zill. Constructing optimal IP routing tables. In *Proc. IEEE INFOCOM 1999*, pages 88–97, 1999.

[EM01] David Eppstein and S. Muthukrishnan. Internet packet filter management and rectangle geometry. In *Proc. 12th Ann. ACM-SIAM Symp. of Discrete Algorithms*, pages 827–835, Philadelphia, PA, 2001. SIAM.

[Ful05] E. W. Fulp. Optimization of network firewall policies using directed acyclic graphs. In *Proc. IEEE Internet Management Conf.*, 2005.

[GM99] P. Gupta and N. McKeown. Packet classification on multiple fields. In *Proc. Conf. on Applications, Technologies, Architectures, and Protocols for Comp. Communication* (SIGCOMM), pages 147–160, New York, NY, 1999. ACM.

[GT85] H. N. Gabow and R. E. Tarjan. A linear-time algorithm for a special case of disjoint set union. *Journal of Computer and System Sciences*, 30(2):209–21, 1985.

[KMT03] H. Kaplan, E. Molad, and R. E. Tarjan. Dynamic rectangular intersection with priorities. In *Proc. 35th Ann. ACM Symposium on Theory of Computing*, pages 639–648, New York, NY, 2003. ACM.

[LNW+02] L. V. S. Lakshmanan, R. T. Ng, C. X. Wang, X. Zhou, and T. J. Johnson. The generalized MDL approach for summarization. In *Proc. Int. Conf. on Very Large Databases* (VLDB'02), pages 766–777, 2002.

[Mas78] W. J. Masek. Some NP-complete set covering problems. Unpublished manuscript, 1978.

[SSW03] S. Suri, T. Sandholm, and P. Warkhede. Compressing two-dimensional routing tables. *Algorithmica*, 35:287–300, 2003.

[TC03] June 2003. `TopCoder` Match Summary, Single Round Match 150, `http://www.topcoder.com/index?t=statistics&c=srm150_prob`.

[Tho03] M. Thorup. Space efficient dynamic stabbing with fast queries. In *Proc. 35th Ann. ACM Symposium on Theory of Computing*, pages 649–658, New York, NY, 2003. ACM.

A Appendix

A.1 Strip Rule Patterns: Characterization Results

Theorem 3.1. *Suppose P is a 2-color pattern. The following statements are equivalent.*

1. *P is a strip-rule pattern.*
2. *P contains no checkerboard subpattern; that is, every 2×2 (not-necessarily-contiguous) subarray of P contains a monochromatic row or column.*
3. (Monotonicity Property). *For any color C the rows of P are hierarchical: For any two rows, the set of columns where C is present in the first row either contains or is contained in the set of columns where C is present in the second.*
4. *The monotonicity property holds with the roles of row and column reversed.*
5. *The following "Pick-Up-Sticks" algorithm always results in an empty grid: Let* gray *be a color not used in P. Call a row or column* pseudo-monochromatic *if it contains at most one color other than gray. Call the grid* empty *if all its cells are colored gray. Perform the following loop: While there is a pseudo-monochromatic row or column containing a cell with a non-gray color, choose such a row or column and color all its cells gray ("pick it up").*

Proof. Let C_1 and C_2 be the two colors. We proceed by showing that (1) and (5) are equivalent, that (2) and (3) are equivalent, that (2) and (4) are equivalent, and that (1) implies (2) and (3) implies (5).

To see that (1) and (5) are equivalent, we first show that (1) implies (5). Suppose P can be constructed using a strip-rule ACL $A = (r_1, r_2, \ldots r_m)$. Then the rows/columns to which r_1 refers are monochromatic in P, and so long as these have not all been picked up by the Pick-Up-Sticks algorithm, there will be a move available to it. Inductively, suppose r_i is the lowest-index rule from A that still has an unpicked row/column. If there is no such r_i then we must have picked up everything, and the Pick-Up-Sticks algorithm has succeeded. Otherwise, any of the unpicked columns to which r_i refers must be pseudo-monochromatic in the remaining pattern and so the Pick-Up-Sticks algorithm can proceed. Consequently, it never gets stuck until the empty grid is reached as claimed. Conversely, if the Pick-Up-Sticks algorithm reaches an empty grid, the sequence of columns/rows that it picked up constitute an ACL for the original pattern. Thus (1) and (5) are equivalent.

To see that (2) and (3) are equivalent, we first show that (2) implies (3). Suppose every 2×2 subarray of P contains a monochromatic row or column. Consider two non-identical rows of P, R_i and R_j. One of these two, say R_i, must have a black cell in a column C_h where the other has a white cell. The two rows will have the monotonicity property unless there is some column C_k where R_j has a black cell and R_i has a white cell. But then these two rows and columns yield a checkerboard subpattern, a contradiction. On the other hand, suppose P does have a checkerboard subpattern M and suppose M is the intersection of rows i and j with columns h and k. Let $c(a, b)$ denote the color of the cell in column b of row a, and assume without loss of generality that $c(i, h) = C_1$. Then $c(i, k) = C_2$, $c(j, h) = C_2$, and $c(j, k) = C_1$. Consequently the set of columns containing C_1 in row i is neither a subset nor superset of the set of columns containing C_1 in row j, so property (3) is violated. The equivalence of (2) and (4) follows analogously.

To see that (1) implies (2) we argue by contradiction. Suppose P has a strip-rule ACL A and contains a 2×2 subarray M with no monochromatic row or column. Suppose M is the intersection of rows i and j with columns h and k. Let r be the first rule in A to include one of these rows and columns. Then the two cells of M in that row/column must get the same color in the final pattern, contradicting the assumption that M had no monochromatic row or column.

Finally, to see that (3) implies (5), we also argue by contradiction. Suppose (3) holds but that the Pick-Up-Sticks algorithm does not result in an empty grid. Then it must get stuck at some subarray M without any monochromatic row or column. Let i be a row of M with the maximum number of cells colored C_1. Since this row is not monochromatic, there must be some column k in M such that $c(i,k) = C_2$. We claim that column k must be monochromatic in M. Suppose not. Then some row j in M must have $c(j,k) = C_1$. But then by (3) row j must have color C_1 in all the columns that row i did, so h has more cells colored C_1 than does row i, a contradiction. ■

Theorem 3.2. *Suppose P is a k-color pattern, $k > 2$. The following statements are equivalent.*

1. *P is a strip-rule pattern.*
2. *Every 2×2 and 3×3 subarray of P contains a monochromatic row or column.*
3. *The "Pick-Up-Sticks" algorithm always results in an empty grid.*

Proof. We have already observed that (1) and (3) are equivalent, and that argument did not rely on the number of colors. The same basic argument as before also shows that (1) implies (2), since the first rule in an ACL to affect a subarray forces the affected rows/columns of the subarray to all get the same color in the final pattern, and the forbidden subarrays have no monochromatic rows or columns.

The non-trivial challenge is to show that (2) implies (3). So suppose that the Pick-Up-Sticks algorithm gets stuck before it can empty the grid. Then P must contain a subarray with no monochromatic rows or columns. Let M be a minimal such subarray, in that every proper subarray of M has either a monochromatic row or column (or both). We shall argue by contradiction that M must be a 2×2 or 3×3 subarray. Suppose it contains no such subarray.

Let r and c be the number of rows and columns in M. Suppose first that $\max(r,c) > 3$, and assume without loss of generality that $r > 3$. Let M_i be the subarrays obtained from M by deleting its ith row, $1 \le i \le r$. Each of these must have a monochromatic row or column by hypothesis. They cannot have monochromatic rows, since their rows are all rows of M, so each must have a monochromatic column. No two can have the same monochromatic column, as this would imply that the column must get the same color in each and that the column would be monochromatic in M as well. Furthermore, no two can get the same color. For suppose two did. Say M_i and M_j, $i \neq j$, both have monochromatic columns with color C. Suppose that M_i's monochromatic column is h and M_j's monochromatic column is k. We have already observed that we must have $h \neq k$, so consider the 2×2 subarray determined by rows i and j and columns h and k. Row i is present in M_j and row j is present in M_i, so we have $c(j,h) = c(i,k) = C$. However, since M does not contain any monochromatic column itself we must have $c(i,h) \neq C$ and $c(j,k) \neq C$. Thus this 2×2 subarray has no monochromatic row or column, a contradiction. Thus each of the M_i must have a different monochromatic column (implying $c \ge r$) and each monochromatic column has a different color.

Consider an $r \times 3$ subarray of M' of M obtained by deleting all the columns except for three of these monochromatic columns. This is a proper subarray of M since as we have observed we must have $c \ge r > 3$. It has no monochromatic columns because M had none. Moreover, if we let C_1, C_2, and C_3 be the three distinct colors associated with the columns of M', then any row of M' must contain at least two of these colors, as row i can only omit the color associated with M_i. Thus M' has no monochromatic row or column and yet is a proper subarray of M, a contradiction.

Thus we must have $\max(r,c) \le 3$. The only remaining case to rule out is the case where M is a 3×2 or 2×3 subarray. Assume without loss of generality that we have the first case, and construct the subarrays M_i, $1 \le i \le r = 3$, as before. Each must have a monochromatic column, and since there are only two columns available, two must share a column, and hence a color, and hence M must have a monochromatic column, a final contradiction. ■

A.1.1 Lower Bound Examples for Theorem 3.3

A first observation is that the Monotonicity Property implies that any black-and-white strip-rule pattern can be constructed with an ACL/RRL consisting of only white row and black column rules (and also entirely out of black row and white column rules). Figure 5 shows the first three patterns P_1, P_2, and P_3 in a series of patterns P_k, where the minimum-length RRL has $2k + 1$ rules but MPUS restricted to black row and white column rules requires $4k + 3$ rules, or $2\mathrm{OPT}_S(P_k) + 1$, as claimed.

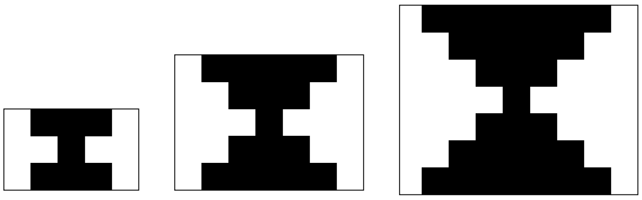

Figure 5: First three patterns in a sequence for RRLs such that $OPT(P_k) = 2k+1$ but Maximal Pick-Up-Sticks can generate RRLs with as many as $4k + 3$ rules.

Figure 6 shows the first two patterns Q_1 and Q_2 in an analogous sequence for ACLs, in which $OPT_S(Q_k) = 2^k$ while MPUS restricted to white row and black column rules will use $2^k(w - k) + 1$ rules. Note that for these examples, the greedy MPUS tie-breaking rule that always picks up the class (of either color or orientation) containing the most unpicked rows/columns will do essentially as poorly. There are also examples that show that greedy MPUS can also be off by an asymptotic factor of 2 for RRLs, as can the algorithm that returns the better of the black-row/white-column and white-row/black-column MPUS RRLs. We know of no simple tie-breaking rule for MPUS that does better asymptotically.

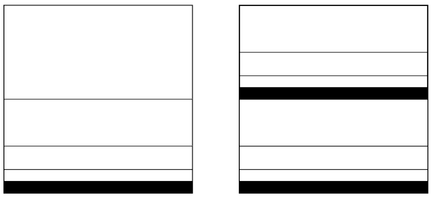

Figure 6: First two patterns in a sequence for ACLs such that $OPT(Q_k) = 2^k$ but MPUS can generate ACLs with as many as $2^k(w - k) + 1$ rules.

A.2 Strip-Rule Patterns: Polynomial-Time Algorithms for RRL/ACL Minimization

A.2.1 Counting the Reachable Vertices in the Segment Graph

Lemma A.1 *Suppose P is a 2-color pattern with n effective gridlines. Then the number of reachable vertices in the RRL segment graph for P is $O(n^2)$.*

Proof. Let n_R be the number of rows in the effective grid and n_C be the number of columns. Since each effective gridline creates one additional row or column, we must have $n_R + n_C \le n + 2$. Let R_i, $1 \le i \le N_R$, denote the row equivalence classes in P, and C_j, $1 \le j \le N_C$, denote the row equivalence classes. Note that we must have $\sum_{i=1}^{N_R} |R_i| = n_R$ and $\sum_{j=1}^{N_C} |C_j| = n_C$. In addition, by Lemma 4.3 the number of reachable states for an ordered pair (E_i, E_j) of equivalence classes that can appear in a label is no more than $|E_j|$. If T is the total number of reachable states summed over all such ordered pairs, we thus have

$$
\begin{aligned}
T &\leq \sum_{i=1}^{N_R} \Big((N_C + N_R - 1)|R_i| \Big) + \sum_{j=1}^{N_C} \Big((N_C + N_R - 1)|C_j| \Big) \\
&< (n+1)\Big(\sum_{i=1}^{N_R} |R_i| + \sum_{j=1}^{N_C} |C_j| \Big) \\
&= (n+1)(n_R + n_C) \leq (n+1)(n+2) = O(n^2). \ \blacksquare
\end{aligned}
$$

A.2.2 The ACL Case

In ACLs the rectangles allowed in rules are restricted. If w is the length of an IP-address, the underlying grid is a $2^w \times 2^w$ grid with rows and columns indexed from 0 to $2^w - 1$. Any legal column and row strip-rule can be specified by a color and an IP-address prefix p, representing the set of all columns/rows whose coordinate x, when written as a w-bit binary number with leading zeros included, has p as a prefix. In what follows we shall represent such rules as pairs (p, c) where p is the prefix and c is the color. If (p, c) and (p', c') are both row or both column rules and p is a proper prefix of p' we shall say that (p, c) *contains* (p', c') – every column/row addressed by the latter rule is addressed by the former. Note that in any optimal ACL, (p', c') must precede (p, c) if it is contained in it; otherwise it would have no effect and could be deleted.

Now let us consider a segment of an ACL as defined in our description of the RRL algorithm. Say a rule in such a segment is *undominated* if it is not contained in any other rule from that segment, and note that the set of columns picked up by the segment is completely determined by the undominated rules it contains. Moreover, in a minimum-length ACL the undominated rules in a segment can all be postponed to the end of the segment and arranged in any order without altering the resulting pattern, since they must already be preceded by any rules they dominate.

Define the *signature* for a segment to be a $(w+1)$-tuple $(x_0, x_1, x_2, \ldots, x_w)$, where x_i is the number of prefixes of length i that are undominated rules in the segment. Given two different signatures, consider the first component in which they disagree. We say that the signature with the larger value in this component is the *stronger* of the two. The *dominant signature* for a collection of segments is the strongest of their signatures.

Lemma A.2 *Suppose we are given a segment with label (E_1, c_1, E_2, c_2, U) and action i and $c_1 \neq c_2$. Then any two minimum-length lists of rules that pick up all the active members of E_i and have the dominant signature among such segments must have the same set of dominant rule prefixes and thus pick up the same set of columns/rows.*

Proof: Suppose not and assume for specificity that we are dealing with column classes. Let the two lists be L_1 and L_2 and let m be the length of the shortest prefix such that they do not have the same set of undominated rule prefixes of that length. Since L_1 and L_2 have the same signature, each must contain a dominant rule prefix of length m that the other does not. Let (p, c) denote the corresponding rule for L_1. Let L_1' be the set of rules in L_1 that are contained in or equal to (p, c), and let L_2' be the set of rules in L_2 that are contained in (p, c). Since (p, c) is a dominant rule in L_1, only rules in L_1' can pick up columns addressed by p. Similarly, since L_1 and L_2 agree on all dominant prefixes shorter than p, no rule in L_2 outside those in L_2' can pick up any columns addressed by p. Thus the segment obtained from L_1 by replacing the rules in L_1' by those in L_2' will also pick up all active rules in E_i and hence can be no shorter than L_1. We thus have $|L_2'| \geq |L_1'|$, and so the segment obtained from L_2 by replacing L_2' by L_1' will also pick up all active rules in E_i and will be no longer than L_2 and hence will also be of minimum length. However, it now has one more undominated rule prefix of length m than did L_2. Since it still has the same number of undominated rule prefixes of all shorter lengths, it thus has a stronger signature than L_2, a contradiction of our assumption that L_2 already had the dominant signature for such segments. ■

We now have the following analogue of Lemma 4.2 for RRLs.

Lemma A.3 *Suppose a minimum-length ACL for P contains a segment with label (E_1, c_1, E_2, c_2, U) that picks up all active members of class E_i. Then there is a minimum-length ACL for P containing a segment with the same label and action that has the following structure:*

1. *If $c_1 = c_2$ (in which case one class consists of rows and the other of columns), let us assume for specificity that E_i is a column class. (The case of row classes is handled analogously.) The segment is then a minimum-length list of column rules that picks up all active members of E_i and has the dominant signature among such segments.*

2. *If $c_1 \neq c_2$ (in which case both classes are column classes or both are row classes), the segment then consists of a minimum-length list of rules that picks up all active members of E_i and has the dominant signature among such segments.*

Proof: Suppose the Lemma is false, and consider an ACL L for P with the maximum number of initial segments that do satisfy the Lemma. Consider the first segment S in L that fails to satisfy the Lemma.

Suppose first that S is of the first type. Then all its rules must be color-c_1 rules and hence can be permuted in any way without affecting the resulting pattern. Thus we may rearrange rules so that the column rules come first, and pick up all the active columns in E_i. Thus in the revised ACL the segment will end with the last of these rules. If the overall ACL is of minimum length, this segment must itself be of minimum length, and hence can have the structure claimed by the Lemma for this case. Thus the new ACL satisfies the Lemma for one additional initial segment, a contradiction.

So suppose that S is of the second type. Assume for specificity that E_1 and E_2 are both column classes and i denotes the white class. As remarked above, we may assume that in S all undominated rules come at the end, with the undominated white rules preceding the undominated black rules. But note that this means that all the active white columns must have been picked up before the first undominated black rule is encountered, and so the segment must end before the first undominated black rule appears. Thus S can be assumed to contain no undominated black rules. Suppose it nevertheless fails to have the desired properties, and let S' be a segment that *does* have them. If S picks up the same set of active columns as S', S cannot be shorter, and so we can replace S by S' and obtain an overall ACL for P that is no longer than the given one and satisfies the Lemma for one additional initial segment, again a contradiction. So S must either pick up some active black column not picked up by S' or fail to pick up some active black column that S' does pick up.

First suppose there is an active black column x picked up by S but not by S'. Moreover, assume that over all minimum-length ACLs that contain segments S with the given label and action, this is the one that picks up the *fewest* black columns not picked up by S'. Let (p, c) be the undominated rule in S that covers column x. By the above argument, (p, c) must be a white rule. S' cannot contain (p, c) or any rule that dominates (p, c) since if it did it would pick up x. Let L_1 be the set of rules in S used to pick up the columns addressed by p, and L'_1 be the corresponding set of rules for S'. Note first that we must have $|L_1| > |L'_1|$, as otherwise we could replace L'_1 by L_1 in S' and obtain a new segment that was no longer than S' and yet had a stronger signature, contradicting our assumption that S' had the dominant signature among all minimum-length segments that picked up all active members of E_i. Let L''_1 be the set of rules obtained by adding $(p, black)$ to L'_1. Then we have $|L''_1| \leq |L_1|$, and note that L''_1 will correctly pick up all the active columns addressed by p – all columns addressed by L'_1 must already be correctly picked up, and any column addressed by p but not picked up by L'_1 must be pseudo-monochromatic black. Thus replacing L_1 by L''_1 in S will yield an ACL that is no longer and has the same effect. However, now note that the segment has an undominated black rule, which we can assume goes at the end, and hence is no longer part of the segment. Thus the resulting segment S'' is shorter than S and so by assumption can pick up no black columns not picked up by S'.

Thus we may assume that S only picks up active black columns that are also picked up by S'. Since S picks up all active members of E_i and S' has minimum length among segments that do so, we must have $|S'| \leq |S|$. Thus we can swap S' for S in our ACL without increasing its length. At the end of the segment fewer black columns may be active than before, but this will not prevent any subsequent rule in the ACL from being applied, and so the ACL will still handle the pattern P. Thus we obtain an ACL for P that satisfies the lemma for an additional initial segment, a final contradiction. ■

To complete our description of the ACL algorithm, we must show how to construct minimum-length segments that consist only of column rules or of row rules, pick up all members of E_i, and have the dominant signature among such segments. To show that the algorithm runs in polynomial time, we will then need to prove an analogue of Lemma 4.3.

We can assume for specificity that we are dealing with column classes. The case of rows is analogous. We may also assume that E_i is a white column class. In the general setting, corresponding to the second part of Lemma A.3, we may have both active white and active black columns. If there are no active black columns, we have the special case corresponding to the first part of that lemma.

Let P' be the pattern corresponding to the label (E_1, c_1, E_2, c_2, U). In what follows, we shall call a column "black" if in P' it contains no white cells and at least one black one, "white" if contains no black cells and at least one white one, "gray" if all its cells are gray, and "red" if it contains both white and black cells in P', that is, if it is not pseudo-monochromatic.

Recall that we say that a rule $r = (p, c)$ is *contained in* a rule $r' = (p', c')$ if p' is a prefix of p, i.e., the range of columns covered by r' contains that for r. Note that if r and r' are such that neither contains the other, the ranges of columns covered by the two rules must be disjoint. Note further that in a minimum-length segment, if r' contains r then the containment must be proper (p' must be a proper prefix of p), since the later of the two rules with the same range would be redundant and could be omitted without affecting the final pattern. Similarly, in a minimum-length segment, if r is contained in r' it must precede it in the rule ordering. The relative ordering of disjoint rules, however, is irrelevant. Thus a minimum-length segment can be specified simply by the rules it contains; any ordering consistent with the above remarks will suffice.

Note that the only rules eligible for inclusion in our segment are ones whose prefixes do not cover red columns. Call such prefixes *legal*. Our algorithm works by computing two values $cost(p, x)$ and $rule(p, x)$ for each legal prefix p and character from $\{W, B\}$, and using these to identify the rules in the segment. The value $cost(p, x)$, $x \in \{W, B\}$, is defined to be the size of the smallest set of rules that have p as a (not-necessarily proper) prefix and collectively pick up all active white columns addressed by p, given that our segment contains a color-c rule whose prefix p' is a proper prefix of p. The value $rule(p, x)$ equals 1 if such a smallest set can contain a rule whose prefix is p and otherwise is 0. Note that since we do not have to pick up black columns, $cost(p, b)$ is also the size of the smallest set of rules that have p as a (non-necessarily proper) prefix and collectively pick up all active white columns addressed by p, given that our segment contains *no* rule whose prefix p' is a proper prefix of p.

We compute the values for $cost$ and $rule$ using dynamic programming, starting with the values for prefixes addressing individual columns. The values for such prefixes are easily seen to be as follows. Let $c \in \{W, B, G\}$ be the color of the addressed column, where the given characters abbreviate "white," "black,", and "gray," respectively. In what follows, let $\bar{W} = B$ and $\bar{B} = W$. Then the values for $cost(p, x)$ when p addresses a single column can be seen to be as follows:

$$cost(p, x) = \begin{cases} 1, & c = \bar{x} \\ 0, & \text{otherwise} \end{cases} \tag{1}$$

It is not difficult to see that for such p we have $rule(p,x) = cost(p,x)$.

If p is a legal prefix that does not address a single column, then both $p0$ and $p1$ are also legal prefixes. In this case we have the following recurrences:

$$cost(p,x) = \min\Big(cost(p0,x) + cost(p1,x), 1 + cost(p0,\bar{x}) + cost(p1,\bar{x})\Big), and$$

$$rule(p,x) = \begin{cases} 1, & cost(p0,x) + cost(p1,x) \geq 1 + cost(p0,\bar{x}) + cost(p1,\bar{x}). \\ 0, & \text{otherwise.} \end{cases}$$

Using the above relations and definitions we can compute the values of $cost(p,x)$ and $rule(p,x)$ for all pairs (p,x) where p is a legal prefix and $x \in \{W,B\}$ in time $O(2^w)$. However, we won't in fact need all these values. We can in fact start with those legal prefixes that address maximal pseudo-monochromatic blocks of columns. And it is easy to see that the above relations imply that (1) and the fact that $rule(p,x) = cost(p,x)$ continue to hold even for legal prefixes p that address blocks of columns, where a block has color B if its columns contain no white cells and a least one black one, color W if its columns contain no black cells and at least one white one, and color G if its columns contain only gray cells. Thus the time to compute all the relevant values will actually be $O(N)$ where N is the number of legal prefixes p that address pseudo-monochromatic blocks and themselves have no prefix that addresses a pseudo-monochromatic block, which is in turn $O(wn)$, where n is the number of effective gridlines in the pattern.

Having computed all these values, we construct the normal-form segment by calling the following recursive function f with arguments p and B for all minimal-length legal prefixes p.

> Function $f(p,x)$
> {
> If $rule(p,x) = 1$, add the rule $(p,\bar{x})$ to the segment and set $x = \bar{x}$.
> If p addresses a pseudo-monochromatic block, return.
> Otherwise, call $f(p0,x)$ and then $f(p1,x)$.
> }

Let $S((E_1,c_1,E_2,c_2,U),i)$ denote the segment constructed by this algorithm (or its appropriate analogue with columns replaced by rows and/or white replaced by black). The following result is relatively straightforward.

Lemma A.4 *For label (E_1,c_1,E_2,c_2,U) and action i, $S((E_1,c_1,E_2,c_2,U),i)$ is a minimum length segment that picks up all active members of E_i and has the dominant signature among all such segments.*

Our remaining task is to prove the analogue of the Containment Lemma (Lemma 4.3) for this ACL normal form segment. For this we will exploit an auxiliary function of legal prefixes that we shall call $bias$. We once again assume for specificity that we are dealing with columns and that our goal is to pick up all the active white columns. We define $bias$ as follows:

$$bias(p) \equiv \begin{cases} -1, & cost(p,B) < cost(p,W) \\ 0, & cost(p,B) = cost(p,W) \\ +1, & cost(p,B) > cost(p,W). \end{cases}$$

Bias in a sense indicates the preferred background color for the given prefix, with -1 indicating a preference for black, $+1$ indicating a preference for white, and 0 indicating indifference.

Note that the following recurrence can be used for computing the values of $bias(p)$. If p is a prefix identifying a single column whose color is $c \in \{W, B, G\}$, then

$$bias(p) \equiv \begin{cases} -1, & c = B \\ 0, & c = G \\ +1, & c = W. \end{cases}$$

Otherwise, $p0$ and $p1$ must be legal prefixes. In this case, the fact that we must have $|cost(p, B) - cost(p, W)| \leq 1$ for all legal p implies that

$$bias(p) = \begin{cases} -1, & bias(p0) + bias(p1) < 0 \\ 0, & bias(p0) + bias(p1) = 0 \\ +1, & bias(p0) + bias(p1) > 0. \end{cases}$$

The following two lemmas are the key to exploiting the $bias$ function:

Lemma A.5 (Monotonicity Lemma) *The bias function is monotonic, in the sense that if any of the original columns is changed from black to gray or from gray to white, then for no prefix p does $bias(p)$ decrease.*

Proof. Follows from the recurrence by which $bias(p)$ is computed. ■

Lemma A.6 *For all legal prefixes p, $bias(p) = +1$ if and only if $rule(p, B) = 1$.*

Proof. First, suppose p addresses a column of color c. If $bias(p) = 1$ then by definition we must have $c = W$ in which case by definition $rule(p, B) = 1$. Conversely, suppose $rule(p, B) = 1$. Then $c = W$, $cost(p, W) = 0$, and $cost(p, B) = 1$, all by definition. But, then $bias(p) = +1$ as desired.

Now let us consider the case where p does not address a column and hence $p0$ and $p1$ are legal prefixes. To simplify the expressions in what follows, we introduce the following shorthand notation:

$$\begin{aligned} c_1 &= cost(p0, B) \\ c_2 &= cost(p1, B) \\ c_3 &= cost(p0, W) \\ c_4 &= cost(p1, W). \end{aligned}$$

First suppose that $rule(p, B) = 1$. By definition, this means that $1 + c_3 + c_4 \leq c_1 + c_2$, which implies that

$$cost(p, B) = \min\Big(c_1 + c_2, 1 + c_3 + c_4\Big) = 1 + c_3 + c_4$$

But then

$$cost(p, W) = \min\Big(c_3 + c_4, 1 + c_1 + c_2\Big) = c_3 + c_4 < cost(p, B)$$

and so $bias(p) = +1$.

Suppose now that $bias(p) = +1$, which means that

$$cost(p, W) = \min\Big(c_3 + c_4, 1 + c_1 + c_2\Big) < cost(p, B) = \min\Big(c_1 + c_2, 1 + c_3 + c_4\Big).$$

If $rule(p, B) = 0$, then by definition we must have $c_1 + c_2 < 1 + c_3 + c_4$ and hence $c_1 + c_2 \leq c_3 + c_4$. This implies $cost(p, B) = c_1 + c_2$. But then since $cost(p, W) < cost(p, B)$ we must have either $1 + c_1 + c_2 <$

$c_1 + c_2$ or $c_3 + c_4 < c_1 + c_2$, both of which yield contradictions. Hence we must have $rule(p, B) = 1$, as desired. ■

The following is an immediate corollary, with analogous results holding when white is replaced by black and/or columns by rows.

Corollary A.7 *Suppose we are given label (E_1, c_1, E_2, c_2, U) and action i, where E_1 and E_2 are column classes and E_i is a white column class. Let X be the set of prefixes of undominated rules in the list $S((E_1, c_1, E_2, c_2, U), i)$ constructed by our algorithm and let Y be the set of legal prefixes p that have $bias(p) = +1$ but which themselves have no legal prefix with this property. Then $X = Y$.*

We are now prepared to prove the analogue of the RRL Containment Lemma for our ACL algorithm. Recall that the algorithm works by computing a shortest path in the segment graph for our given pattern P.

Lemma A.8 (ACL Containment Lemma) *Suppose that vertices $v = (E_1, c_1, E_2, c_2, U)$ and $v' = (E_1, c_1, E_2, c_2, U')$ are both reachable from the source vertex in the segment graph for P and $U \neq U'$. Then either $U \subset U$ or $U' \subset U$.*

Proof. As in the RRL case, we argue by induction on the depth of the vertices in the graph, with the claim clearly holding for the source vertex. Suppose it holds for all vertices with depth d or less and v and v' are at depth $d + 1$. Once again the hard case is when $c_1 = c_2$, so that one of E_1 and E_2 is a column class and the other is a row class. Assume for specificity that E_2 is the column class. (The other cases can be argued analogously.)

As before, arcs into these two vertices can be from vertices of only two types, either (E_1', c, E_2, c_2, X) (*Type 1*) or (E_2, c_2, E_1', c, Y) (*Type 2*), where c is the other color from $c_1 = c_2$ and E_1' is the previously picked-up column class with the least cells of color c among those that were picked up with color c. Since our two given vertices are by assumption reachable from the start vertex, there must be at least one arc into each from a reachable vertex of depth d. Let a be such an arc into the first and a' be such an arc into the second and let x and x' be the tail vertices for a and a' respectively. We break into cases depending on the types of x and x', with the case where the types are different following from the two cases where they are the same.

First, assume that they are both of Type 1, in which case we can denote them as $x = (E_1', c, E_2, c_2, X)$ and $x' = (E_1', c, E_2, c_2, X')$, where $U \subseteq X$ and $U' \subseteq X'$ and by our induction hypothesis we know that either $X \subseteq X'$ or $X' \subseteq X$. We can assume without loss of generality that $X \subseteq X'$. we claim that this implies that $U \subseteq U'$. Let P_x and $P_{x'}$ be the patterns corresponding to x and x' respectively, and let us assume, again without loss of generality, that c_2 is white. The sets of legal prefixes for the segments corresponding to arcs A and A' are the same, but our normal-form algorithm may yield different results since some of the columns that are black in $P_{x'}$ are gray in P_x. Let the corresponding bias functions be denoted by $bias_x(p)$ and $bias_{x'}(p)$. By our Monotonicity Lemma (Lemma A.5), we have that for all legal prefixes p, $bias_x(p) \geq bias_{x'}(p)$. But then Corollary A.7 implies that every undominated rule (p, c) in the normal-form segment for a' is either undominated in the segment for a or dominated by a rule (p', c') where p' is a prefix of p. Thus every column addressed by a rule in the segment for a' is addressed by a rule in the segment for a, which implies that $U \subseteq U'$, as desired.

The argument for the case where both x and x' are of Type 2 is analogous. ■

Thus the total number of states we encounter in the algorithm is once again $O(n^2)$ by Lemma A.1. Because our bound on the running time for generating the segments is $O(nw)$, the overall running time becomes $O(n^3 w)$.

A.3 NP-completeness for 2-color RRLs: Full Details

In this section we prove that the problem of constructing a minimum-length (general) RRL is NP-hard. The proof is by a transformation from RECTILINEAR PICTURE COMPRESSION. Suppose we are given a rectilinear black pattern P on a grid with a white background and asked if there is a collection of k or fewer black rectangles whose union is the pattern. We will construct a rectilinear black and white pattern P' and an integer k' such that P' can be generated by a length-k' RRL if and only if P can be represented as the union of k or fewer black rectangles.

If we could somehow restrict the RRL to use only black rules, we could simply use $P' = P$, but unfortunately, as we have seen, white rules can make a difference and cannot be ignored. Our solution is to embed P into a much bigger pattern which will effectively eliminate the usefulness of white rules in constructing P. To do this, we exploit some basic observations about strip-rule patterns.

The basic building block of our new pattern is the 7×7 tile depicted in Figure 4. Note that this figure contains 12 effective gridlines, 6 horizontal and 6 vertical. (We do not count the boundaries of the tile itself, which will not be effective gridlines in our overall construction.) A strip-rule RRL for this pattern will need at least 3 horizontal and 3 vertical rules since no strip-rule can create more than 2 effective gridlines. Moreover, such an RRL exists: Let us view the RRL in pick-up-sticks order, that is, the order in which earlier rules overwrite later ones and we model this by saying that a rule "picks up" or "turns gray" the cells to which it refers. We first pick up the vertical and horizontal black strips covering the middle column and row, then the width-3 horizontal and vertical white strips covering the middle three columns/rows, and finally the vertical and horizontal black strips covering the middle five columns/rows. This RRL is uniquely optimum among strip-rule RRLs: There are four equivalence classes of rows and of columns, so we must pick up at least three column classes and three row classes. Hence in a length-6 RRL for this pattern, each rule must pick up an entire equivalence class, and consequently the sequence of moves is forced (up to an interchange of horizontal/vertical moves of the same width).

Let N be a large number to be specified later. Our basic construction involves a $21N \times 21N$ grid (the *overall grid*) constructed from these tiles as follows. First completely fill up the grid with $3N$ rows of $3N$ tiles. Let the *frame* of the pattern consist of the union of all those tiles in the first N rows of tiles, the last N rows of tiles, the first N columns of tiles, and the last N columns of tiles. The remaining (central) $N \times N$ grid of tiles constitutes the *picture portion* of the grid. To complete the construction, we modify the picture portion of the grid by turning a subset of its white cells black, that subset to be specified later. There are certain common observations we can make about RRLs for the overall grid, independent of how this last alteration is performed. In what follows, we shall continue to view the RRL in pick-up-sticks order.

Consider just the $12N - 4$ tiles on the boundary of the grid. Collectively they contain $72N$ transitions between black and white perpendicular to the boundary and touching it. Note that no rectangle rule can generate more than four of these transitions, so the RRL must be of length at least $18N$. Suppose in what follows that the RRL is of length $18N + x$ for $x < N/4$. What can we say about it? Note that a strip-rule can generate four of the $72N$ transitions, two on one side of the overall grid and two on the other. Call a rule *defective* if it either is not a strip-rule, is a strip-rule one of whose boundaries is not an effective gridline of the pattern, or is a strip-rule one of whose boundaries is also the boundary of an earlier strip-rule in the RRL. Note that no defective rule can generate more than two of the $72N$ boundary transitions. Thus our length-$(18N + x)$ RRL can contain at most $2x$ defective rules and must contain at least $18N - x$ non-defective strip-rules.

Say a row of tiles is *clean* if no defective rule in the RRL has a horizontal boundary inside it. Similarly, call a column of tiles *clean* if no defective rule has a vertical boundary inside it. By the above, there must be at least $3N - 4x$ clean rows of tiles and $3N - 4x$ clean columns of tiles, since each defective rule has at most two horizontal and two vertical boundaries. Say a tile is *pure* if it is in both a clean row and a clean column of tiles. Since every row and column contains $2N$ frame tiles and $4x < N$, every clean row of tiles must contain more than N pure frame tiles, as must every clean column. Moreover, each clean row and column must contain a pure tile on each side of the picture portion of the overall grid.

Now by definition, the only rules with boundaries inside a pure tile are strip-rules. As the reader may verify, the only way to generate the pattern in a pure frame tile using strip-rules (even ones with only one boundary inside the tile) is to use the previously mentioned set of rules that were optimal for an individual tile, in the specified order. As can be confirmed by case analysis, any failure to follow that scheme while using strip rules leads to the necessity of using two strip-rules that share a boundary, implying that one of them is defective, which is not allowed since this is a pure tile. (Note that a pure tile can be contained in the *interior* of a defective rule, so one must consider separately the cases in which it is so contained or it is not, and if it is, the case in which the first defective rule containing it is black and the case in which the first such rule is white. Fortunately, this white rule case is equivalent to that in which no defective rule contains the tile, since in the latter case the default background applies, and it is white.) Thus, since every clean row and column must contain at least one pure frame tile, we have the following: for each clean row of tiles, our RRL must contain three non-defective strip-rules that pass through its center – a width-1 black rule, a width-3 white rule, and a width-5 black rule – and they must occur in this order. The analogous property holds for clean columns.

Our NP-completeness proof works by embedding our given RECTILINEAR PICTURE COMPRESSION pattern P into the picture portion of the grid in such a way that P can be constructed out of k rectangles if and only if the resulting $21N \times 21N$ grid pattern has an RRL of length $18N + k$ or less.

We may assume without loss of generality that P is given as a pattern in a $3k \times 3k$ grid, since if it can be realized as a union of k black rectangles it has at most $2k$ effective horizontal gridlines and $2k$ effective vertical gridlines. Let $M = 4k + 1$ and $N = 3kM$. We can then view our overall grid (before the picture portion is modified) as a $9k \times 9k$ grid of *supertiles*, each supertile in turn being an $M \times M$ grid of our standard tiles. The picture portion of the grid is thus a $3k \times 3k$ grid of supertiles, which we can put in one-to-one correspondence with the grid cells of the $3k \times 3k$ grid that contains P. The modification of the picture portion of the overall grid is then obtained by turning black all the white cells in the set of supertiles corresponding to black grid cells in P. This completes the construction of P'. See Figure 7 for a schematic example, where in order to fit the grid on the page, we have taken $N = 3$ and have pretended that a supertile consists of a single tile rather than an $M \times M$ grid of them.

We now wish to show that P' has an RRL of length $18N + k$ if and only if P is representable by a collection of k or fewer rectangles. One direction is easy. If P is so representable, then our RRL consists of the $18N$ strip-rules required to construct the frame of the grid, preceded by the k or fewer black rectangles of supertiles corresponding to the rectangles in the collection of rectangles that represent P.

Now suppose that the overall grid has an RRL of length $18N+k$ or less. Since by construction $k < N/4$ the above structural arguments apply with $x = k$ and we know that the RRL contains at most $2k$ defective rules, yields at most $4k$ unclean rows and $4k$ unclean columns, and is structured as claimed above for the clean rows and columns. Consider any black supertile in the picture portion of the grid. Since it is M tiles high and M tiles wide and $M - 4k = 1$, there must be at least one clean row and one clean column of tiles passing through the interior of the supertile, and the supertile must contain at least one pure tile. If one looks only at the effect of non-defective rules on this tile, they must yield the default pattern of Figure 4. Since the tile is all-black in the final pattern, it must be intersected by at least one defective black rule that occurs before the width-3 white strip-rules that pass through its center. Since the tile is pure, that defective black rule must include all of the tile. Call such a defective rule a *covering black rule*.

For each black supertile, pick one covering black rule that contains a pure tile from that supertile and label the supertile with that rule. Call this the *chosen rule* for the supertile. Also, pick a pure tile in the intersection of the rule with the supertile and call this the *chosen pure tile* for the supertile.

Claim A.9 *Let S be a black supertile, R the chosen rule for S, and T the chosen pure tile for S. Then any*

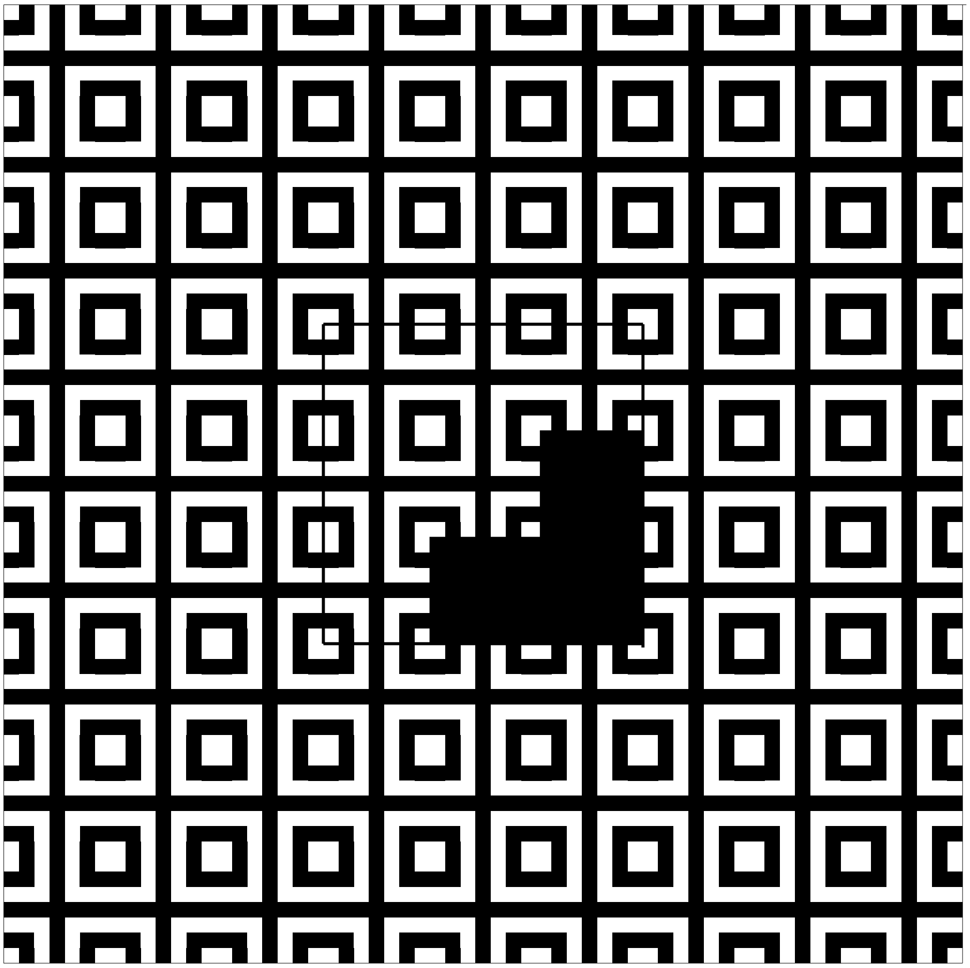

Figure 7: The grid constructed in the NP-completeness proof for RRLs. The picture portion of the construction is outlined. In order to fit the picture on a page, we take $N = 3$ and we consider a supertile to consist of a single tile.

pure tile T' that is in R and in the same clean row or column as T must be contained in a black supertile from the image P' of P.

Proof of Claim. Suppose that T' is not in a black supertile from the image of P. Then T' must look like Figure 4 in the final pattern. As mentioned above, R must precede the width-3 white rules that go through T', and hence would turn the whole tile black unless it in turn were preceded by additional rules that recreated the desired pattern for the tile. The only rules from our required set that can precede the required width-3 white rules are the required width-1 black rules through the center of the tile. So at least one additional rule is needed to create the transitions that occur at the top and bottom (or left and right) of the width-3 white rule. But rule boundaries are needed if transitions are to be created, so this would mean that two rules sharing a common boundary would intersect the tile (this new rule and the original width-3 white rule). One of these would by definition have to be defective, thus contradicting our assumption that T' is a pure tile. ■

Note that the Claim implies that R cannot share any border with the boundary of the overall grid, for if it did, it would of necessity contain a pure tile from the frame that shared a clean row or column with T, and the black supertiles in P' are all contained in the picture portion of the overall grid. Thus the chosen rules cannot create even one of the $72N$ boundary transitions and so there can be at most k such rules if there are to be $18N + k$ or fewer rules over all.

Now, for each chosen covering black rule, consider the smallest rectangle that contains all the supertiles that it labels. Each such rectangle will be made up of supertiles entirely contained in the picture portion of our grid, and by the previous paragraph there will be at most k of them. We claim that the corresponding collection of rectangles in the $3k \times 3k$ grid for P must be the desired representation for P, that is, their union will precisely equal P.

By construction their union *includes* all the black cells of P. Suppose it also includes a white cell C, and let R be the rectangle in our cover that includes it. Let R' be the covering black rule in our RRL that gave rise to R and $h(R')$ be the smallest rectangle that encloses all the supertiles labeled by R'. Let C' be the supertile in our overall grid corresponding to C, and note that C' is not one of the black supertiles of P' since C is not black in P. Since C' is in $h(R')$, there must be supertiles C'' and C''' labeled by R' such that the two are either to the left and right of C', above and below C', or one is above/below and the other is left/right. Consider the chosen pure tiles T'' and T''' for C'' and C''' respectively. Let $p(R')$ be the smallest rectangle that encloses these two tiles. This rectangle is contained in R' and must contain tiles in C'.

If C'' and C''' are respectively to the left and right of C' or respectively above and below it, then $p(R')$ must contain the intersection of C' with the clean row/column that contains T''. Since C' is of width/height $M = 4k + 1$, this clean row/column must be intersected by at least one clean column/row inside C', and hence must contain a pure tile inside C'. But by the Claim this pure tile would have to be in a black supertile of P', a contradiction.

Suppose on the other hand that C'' and C''' are not on opposite sides of C'. Assume without loss of generality that C'' is to the left or right of C' and C''' is above or below. Now $p(R')$ must contain the tile that is in the intersection of the clean row containing T'' and the clean column containing T'''. This tile would thus be a pure tile in R' sharing a clean row with T'' and hence would have to be in a black supertile of P', again a contradiction.

Thus the hypothesized white cell C contained in our cover of P cannot exist and so our cover is exact. We conclude that there does exist a representation of P with k or fewer black rectangles, as claimed, and our construction represents the desired polynomial transformation. Hence it is NP-hard to construct an optimal RRL for an arbitrary rectilinear black and white pattern.

A.4 Approximation Algorithms for Unrestricted Patterns

In this section we prove that the Iterated Strip-Rule Algorithm (ISR) described in Section 6 provides the claimed approximation guarantee for unrestricted patterns with arbitrarily many colors. Suppose we are given such a pattern P by an arbitrary-rule RRL of length n. Let the effective grid for P have N_{col} columns and N_{row} rows, and note that both N_{col} and N_{row} are at most $n + 1$.

Theorem 6.1. *The iterated strip-rule algorithm runs in polynomial time and produces an RRL whose length is* $O(\mathrm{OPT}(P)\min(n^{1/3}, \mathrm{OPT}^{1/2}))$. *Moreover, the bound is tight in that there exist patterns with arbitrarily large values of* $\mathrm{OPT}(P) = \Theta(N_{col})$ *for which the algorithm produces RRLs whose lengths are* $\Theta\left(\mathrm{OPT}(P)^{3/2}\right)$.

Proof. The polynomial time bound for ISR is easy to verify, so here we will just consider the upper and lower bounds on the approximation guarantee. Recall the description of the algorithm from Section 6. As in that description, we shall assume without loss of generality that $N_{col} \geq N_{row}$. Note that if a block has x rows and y columns, the number of rules needed to handle it by Maximal Pick-Up-Sticks is at most $x+y-1$. We can thus bound the total number of rules in $\mathcal{R}_q$, the RRL constructed by ISR when P is divided into q vertical strips, as follows. First note that each combination of a good block together with a bad block immediately above it constitutes a forbidden rectangle. These forbidden rectangles are all disjoint, and so if the total number of bad blocks is B, then $OPT(P) \geq B/4$ by Lemma 6.2. Consider the blocks in strip i. These all have width roughly N_{col}/q. They alternate between good and bad blocks, and so if there are

B_i bad blocks in the ith horizontal strip, there are at most $B_i + 1$ good blocks. The total height of the good blocks in the strip is at most N_{row}.

Thus there are at most $B + q$ good blocks, all with width roughly N_{col}/q and cumulative height at most qN_{row}, and B bad blocks each of which is one column high and roughly N_{col}/q rows wide. Exploiting the fact that $N_{row} \leq N_{col}$ we thus have The total number of rules $|\mathcal{R}_q|$ is roughly bounded by

$$\begin{aligned}(B+q)\frac{N_{col}}{q} + qN_{col} + B\frac{N_{col}}{q} &= 2B\frac{N_{col}}{q} + N_{col} + qN_{col} \\ &\leq 8\mathrm{OPT}(P)\frac{N_{col}}{q} + N_{col}(q+1).\end{aligned}$$

Now since $\mathrm{OPT}(P) \leq N_{col}N_{row} \leq N_{col}^2$ we have $\sqrt{\mathrm{OPT}(P)} \leq N_{col}$. Since no rule can create more than two effective vertical gridlines, we also must have $N_{col} \leq 2\mathrm{OPT}(P) + 1$. Consider the above bound on $|\mathcal{R}_q|$ for $q = \lceil\sqrt{\mathrm{OPT}(P)}\rceil$. In this case we obtain

$$|\mathcal{R}_q| = O\left(\mathrm{OPT}(P)\frac{N_{col}}{\sqrt{\mathrm{OPT}(P)}} + N_{col}\sqrt{\mathrm{OPT}(P)}\right) = O\left(\mathrm{OPT}(P)^{3/2}\right).$$

Thus the best of the $\mathcal{R}_q$'s contains at most $O\left(\mathrm{OPT}(P)^{3/2}\right)$ rules. This will be $O(\mathrm{OPT}(P)n^{1/3})$ unless $\mathrm{OPT}(P) > n^{2/3}$, in which case the input RRL, which has length n, will be $O(\mathrm{OPT}(P)n^{1/3})$. Our claimed overall bound of $O\left(\mathrm{OPT}(P)n^{1/3}, \mathrm{OPT}(P)^{1/2})\right)$ follows from the fact that we return the input RRL if it is shorter than all the constructed $\mathcal{R}_q$'s. (Thanks to Mihai Patrascu for this observation.) This completes the upper bound proof.

Finally, we need to show that ISR can be this bad. For every M we exhibit a 2-color pattern P_M on a $2M^2 \times 2M^2$ grid, that has $OPT(P) \leq 3M^2$ and yet $ISR(P) = \Omega(M^3)$. To construct our pattern, we start with a white background, then color all even rows and columns black, and finish off by recoloring the following M^2 black cells white, with each recoloring forming a local checkerboard pattern. The recolored cells have coordinates $\big(M(i-1) + 2i, M(j-1) + 2j\big)$ for $1 \leq i, j \leq M$. Note that this pattern can be described with $3M^2$ rules – M^2 row rules plus M^2 column rules plus M^2 individual cell rules. See Figure 8 for the pattern P_4.

We suppose that the pattern is given to us inefficiently, with one rule for each of its $4M^4$ cells, in which case returning the input RRL will be of no use to us. Now note that for a given value of q, ISR must handle q strips of approximate dimensions $2M^2 \times 2M^2/q$, having roughly $2M^2 + 2M^2/q$ effective grid lines. Thus it generates $\Theta(M^2)$ rules for each strip. If $q > M$ the overall RRL will thus have to contain $\Theta(M^3)$ rules and we are done. So assume $q \leq M$. Then all the strips have height roughly $2M$ or greater. In running the algorithm we will encounter M^2 checkerboards overall, no two at the same time. Each will yield a bad rectangle for which MPUS will generate a number of rules equal to roughly half the strip height, for a total once again of $\Omega(M^3)$ rules. (Note that the above argument works even if we run the algorithm with the roles of rows and columns interchanged, so trying both possibilities and taking the better will not yield improved results.) ■

A.5 $\mathrm{OPT}_S(P)$ Versus $\mathrm{OPT}(P)$ for Strip-Rule Patterns

Figure 9 shows the first three patterns in a series of patterns P_k for which $\mathrm{OPT}_S(P_k)/\mathrm{OPT}(P_k)$ approaches 3.5 in the limit. Pattern P_k contains $8(k+1)$ effective gridlines, and hence requires at least $2k+2$ rules, and an RRL of this length is easy to discover. The fact that a minimum-length strip-rule RRL requires $7k + 6$ rules can be verified by a detailed analysis of how our optimization algorithm for strip-rule RRLs handles patterns with the given structure. Figure 10 gives examples of ACL's for which $\mathrm{OPT}_S(P_k)/\mathrm{OPT}(P_k) = \Theta(w)$.

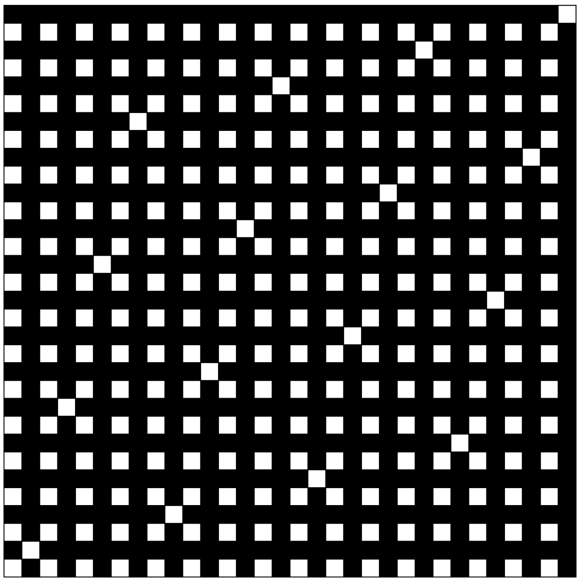

Figure 8: ISR lower bound example for $M = 4$.

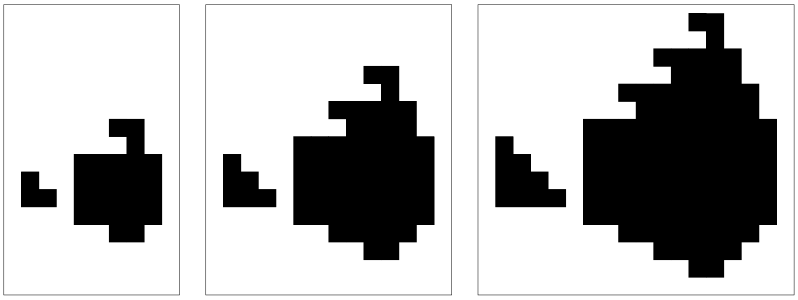

Figure 9: First three patterns P_1, P_2, and P_3 in a sequence for RRLs such that $\mathrm{OPT}(P_k) = 2k + 2$ and $\mathrm{OPT}_S(P) = 7k + 6$.

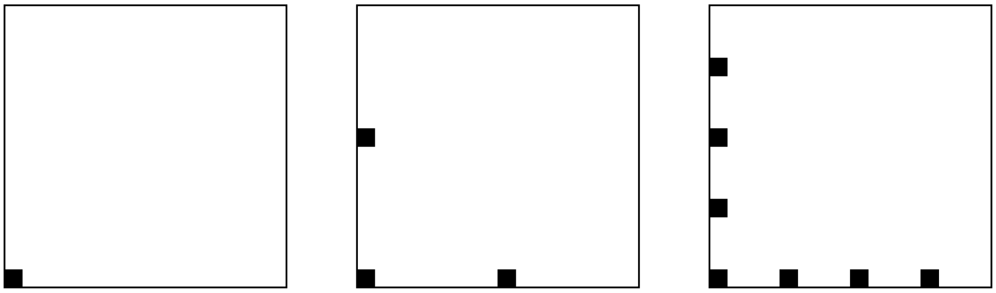

Figure 10: First three patterns Q_1, Q_2, and Q_3 in a sequence for ACLs such that $\mathrm{OPT}(Q_k) = 2^k - 1$ and $\mathrm{OPT}_S(P) = 2^{k-1}(w - k) + 2^k - 1$.