The Complexity of Compressing Subsegments of Images Described by Finite Automata*

Juhani Karhumäki † **Wojciech Plandowski** ‡ **Wojciech Rytter** §

Key-words: square images, finite automata, complexity

Abstract

We investigate how the compression size of the compressed version of a 2-dimensional image changes when we cut off a part of it, e.g. extract a photo of one person from a photo of a group of people, when compression is considered in terms of finite automata. Denote by $c(T)$ the compression size of a square image T in terms of deterministic automata, it is the smallest size of a deterministic acyclic automaton A describing T. The corresponding alphabet of A has only four letters, corresponding to four quadrants. We consider an independent useful combinatorial interpretation of $c(T)$ in terms of *regular* subsquares of T. Denote by $\Psi(n)$ the largest compression size $c(R)$ of a square subsegment R of the image T such that $c(T) = n$. We show that there is a constant $c > 0$ such that:

$$c \cdot n^{2.5} \leq \Psi(n) \leq n^{2.5}$$

For weighted automata we show that the compression size grows only linearly, if T is described by a weighted automaton with n states and m edges then a subimage R can be described by a similar automaton having $O(n)$ states and $O(m)$ edges.
We also show how to construct efficiently (in linear time w.r.t. the total size of the input and the produced output) the compressed representation of subsegments given the compressed representation of the whole image.

1 Introduction

The compression size of images is of crucial importance in multimedia systems and in transferring large images in WWW. Deterministic and weighted finite automata are successful

*A preliminary version of the paper was presented at conference CPM'99 [12].

†Department of Mathematics, Turku University, Finland and Turku Centre for Computer Science, DataCity 4th floor, Turku, 20014 Finland. Supported by Academy of Finland under grant 44087. Email:karhumak@cs.utu.fi.

‡Instytut Informatyki, Uniwersytet Warszawski, Banacha 2, 02–097 Warszawa, Poland and Department of Mathematics, Turku University, Finland and Turku Centre for Computer Science, Finland. Supported in part by KBN grant 8T11C03915 and in part by Academy of Finland under grant 44087. Email:wojtekpl@mimuw.edu.pl.

§Instytut Informatyki, Uniwersytet Warszawski Banacha 2, 02–097 Warszawa, Poland and Department of Computer Science, University of Liverpool, Great Britain. Supported in part by KBN grant 8T11C03915. Email:rytter@mimuw.edu.pl

tools for compressing 2-dimensional images, see [3, 4, 5, 7]. There are several software packages using this type of compression, see [9, 4]. Finite automata can describe quite complicated images, for example deterministic automata can describe the Hilbert's curve with a given resolution, see [11], while weighted automata can describe even much more complicated curves, see also [3, 4, 5]. The objects considered are potentially exponentially compressed, so algorithms which apply decompression are theoretically not polynomial time algorithms. In practice exponential compression does not usually take place, nevertheless the compression ratio for two-dimensional images can be very high, especially compared with the one dimensional case. This happens, for example, for images corresponding to fractals having short description. For one-dimensional words there exist polynomial-time deterministic algorithms for compressed and fully compressed pattern-matching [8, 10, 13], despite the fact that the uncompressed size of objects could be exponential. However, these problems become much harder in the two-dimensional case. Our main result is a constructive proof of the fact that compressed size of subsegments of an automata-compressed images grows only polynomially. This contrasts with the exponential growth of compression size of subimages for compression in terms of recursive descriptions, see [2].

We consider a subsegment image R and the host image T, denote by $Compress(R)$ and $Compress(T)$ the smallest automata describing R and T, respectively.

Our main problem is the **Subsegment Compression Problem**:

Instance: $Compress(T)$ representation of a $2^k \times 2^k$ square image, a point x in T and an integer $k' < k$. Let R be a square $2^{k'} \times 2^{k'}$ subsegment of T whose left-upper corner is positioned at x.

Question: (1) Estimate the size of $Compress(R)$. **(2)** Construct $Compress(R)$.

For deterministic automata we define the size of the automaton as the number of its *essential* states (those which are on accepting paths). The number of edges is only linear with respect to the number of states since the alphabet is constant. Define $c(T)$, the compression size of T in terms of deterministc automata, as the smallest size of an acyclic deterministic automaton describing T. Denote by $\Psi(n)$ the largest compression size $c(R)$ of a square subsegment R of the image T such that $c(T) = n$. Our main result is the tight bound: $\Psi(n) = \Theta(n^{2.5})$.

In case of weighted automata we show that the compression size of the subsegment grows only linearly. In the weighted case the size of the automaton is given by two numbers: the number n of states and the number m of edges. The weights are assumed to be of a constant size.

The algorithms for the main problem are used to improve pattern-matching for images compressed in terms of deterministic automata. The similar compressed pattern-matching problem for weighted automata is NP-complete, see [2]. Another application is a polynomial time algorithm for the pattern-checking (it consists in testing the existence of a fixed occurrence of a large compressed image). The last problem is *co-NP* complete for 2-dimensional compressions in terms of recursive generations, see [2].

2 Deterministic finite automata and functions τ, ϕ

Our alphabet is $\Sigma = \{0,\ 1,\ 2,\ 3\}$, the elements of which correspond to four quadrants of a square array, see Figure 1.

1	3
0	2

Figure 1: Enumeration of the quadrants.

A word w of length k over Σ can be interpreted, in a natural way, as a unique address of a pixel x of a $2^k \times 2^k$ image (array), we write $address(x) = w$. The length k is called the *resolution* of the image. For a language $L \subseteq \Sigma^+$ we denote by $Image_k(L)$ the $2^k \times 2^k$ black-and-white image such that the color of a given pixel x is black iff $address(x) \in L$. We consider also the weighted languages, formally they correspond to functions which associates with each word w a value $weight_L(w)$. A weighted language L over Σ and resolution k determine the gray-tone image $Image_k(L)$ such that the color of a given pixel x equals $weight_L(address(x))$. If all words in L are of the same length k then we can omit the subscript k and write $Image(L)$. Our description of the language is in terms of finite (unweighted or weighted) automaton A, cf. [3, 11]. We define $Image_k(A) = Image_k(L(A))$, where $L(A)$ is the language accepted by A. We count only essential states. We call a state *essential* iff it is on a path to an accepting state.

Let $A = (\{0,1,2,3\}, Q, q_0, \delta)$ be a deterministic acyclic automaton of depth n where Q is a set of states, $q_0 \in Q$ is the initial state, and $\delta : Q \times \Sigma^* \to Q$ is a partial transition function. The automaton A defines the language

$$L(A) = \{w : \delta(q_0, w) \text{ is defined, } |w| = n \text{ and } \delta(q_0, w) \text{is accepting}\}.$$

Note, that in the definition of an automaton we do not need to specify the single accepting state. For $q \in Q$, denote by $Image(q)$ the image which is generated by the automaton which is obtained from A by changing its initial state to q. Clearly, $Image(q_0) = Image(A)$. All considered automata are assumed to be *acyclic*. It is motivated by the fact that we consider only finite resolution images, or more precisely finite resolution approximations of infinite resolution images. We define *the depth* of the automaton as the longest path from the initial state to an accepting state. The depth corresponds to the resolution of the image. An acyclic automaton can be transformed to an equivalent automaton in which for each state q each path from the initial state to q has the same length. We say that a state q *belongs to level* t if all paths from the initial state to q are of length t. Here, the length of a path is the number of edges in the path.

A **regular block** of a $2^k \times 2^k$ image T is defined as follows. T is a regular block, and if B is a regular block then all its quadrants are also regular blocks.
We can interpret the state q as a name of a regular block. We can also consider the state as

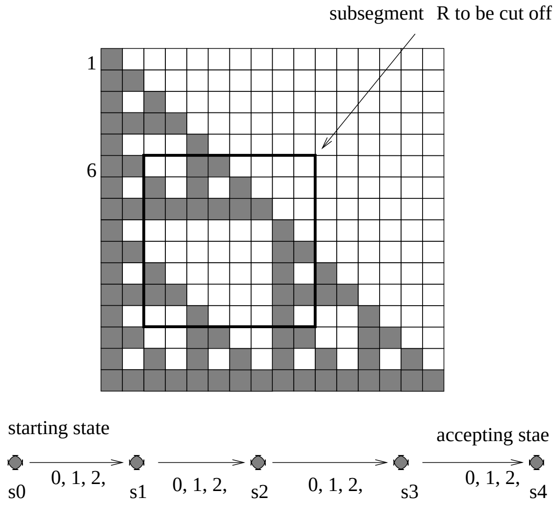

Figure 2: The image S_4 and a smallest acyclic deterministic automaton describing it. Edges which are not on accepting paths are disregarded. $\tau(S_4) = 5$, $\phi(S_4) = 20$, $\tau(R) = 9$.

a nonterminal of a 2-dimensional grammar generating the image. A *balanced 2-dimensional grammar* describes recursively each regular subsquare of the image in terms of the names (nonterminals) of its quadrants. Each production of such grammar corresponds to the decomposition of a square into 4 smaller quadrants of *the same shape* (this motivates the word *balanced*). The terminal symbols are 0 and 1, they correspond to blank and black pixels, respectively. For simplicity denote by $\hat{\emptyset}$ a square blank block consisting only of white pixels, and use the same notation for all possible sizes of $\hat{\emptyset}$. For the automaton from Figure 2 the corresponding 2-dimensional grammar looks as follows

$$s_0 \rightarrow \begin{bmatrix} s_1 & \hat{\emptyset} \\ s_1 & s_1 \end{bmatrix} \quad s_1 \rightarrow \begin{bmatrix} s_2 & \hat{\emptyset} \\ s_2 & s_2 \end{bmatrix} \quad s_2 \rightarrow \begin{bmatrix} s_3 & \hat{\emptyset} \\ s_3 & s_3 \end{bmatrix} \quad s_3 \rightarrow \begin{bmatrix} s_4 & \hat{\emptyset} \\ s_4 & s_4 \end{bmatrix} \quad s_4 \rightarrow \begin{bmatrix} 1 \end{bmatrix}$$

Essential nonterminals are the ones which generate nonblank texts. Observe that essential states of a minimal acyclic deterministic automaton describing an image T correspond to essential nonterminals in a minimal 2-dimensional grammar describing T.

The crucial notion is that of a **basic block** of T, defined as a *regular* block or a subsegment consisting of 4 adjacent *regular* blocks of T. Define:

$\tau(T),\ \phi(T)$ to be the number of distinct nonblank regular (basic) blocks of T;

The basic properties (to be proved later) of these functions are:
(1) $\tau(T)$ corresponds to the automata and grammar ompression size of T;
(2) $\tau(R) \leq \phi(T)$ for each square subsegment R of T.
(3) $\phi(T) \leq \tau(T)^{2.5}$

Example.
For images $S_4,\ R$ in Figures 2 and 3, respectively, we have

$$\tau(S_4) = 5,\ \phi(S_4) = 20,\ \tau(R) = 9.$$

Recall that $c(T)$ was defined as the minimal number of essential states of a deterministic automaton desribing T.

Lemma 2.1 $c(T) = \tau(T)$

Proof: Let d be the number of nonblank regular blocks of T. First, we prove that there is an automaton A describing T with d states. Each state q of A corresponds to one regular block $Image(q)$, and different states corresponds to different regular blocks. There is an edge from state q_1 to q_2 labeled by i if $Image(q_2)$ is in the i-th quadrant of $Image(q_1)$. Assume that there is an automaton A describing T and having $s < d$ states. Consider the set $\{Image(q) : q \in States(A)\}$. By pigeon hole principle there is a regular block of T which does not correspond to any state of A so there is a regular block which is not generated by states of A. This means that no state of A corresponds to T. □

Example. $Image(\{0,1,2\}^k) = S_k$ is the $2^k \times 2^k$ black-and-white square part of Sierpinski's triangle, see Figure 2 for the case $k = 4$. The corresponding smallest acyclic deterministic automaton accepting all paths describing black pixels has 5 states. The automaton for its subsegemnt R is illustrated in Figure 3. The states at depth 2 of the automaton from Figure 3 correspond to the blocks of the subsegment R in the way shown in Figure 4.

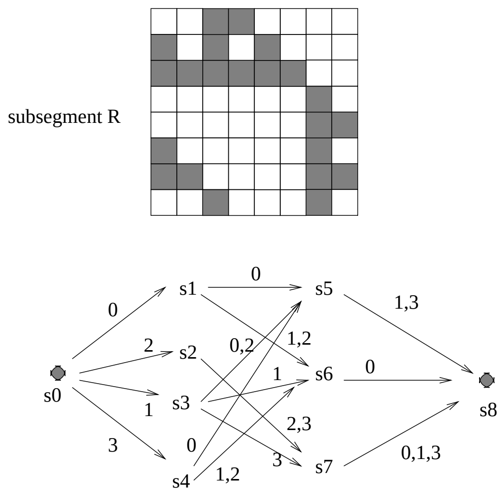

Figure 3: The subsegment R of S_4 and an acyclic 9-state automaton describing R.

3 $\Psi(n) \leq n^{2.5}$

We can use the combinatorial interpretation of $\tau(T)$, which is especially more convenient for the upper bound (as defined in terms idependent of automata). A square subsegment of the shape $2^t \times 2^t$ is said to be of *rank* t.
For a subblock X of a block X define the position of X in Y ($pos_X(Y)$) as the position of left-upper corner of Y inside X.

For a block X of rank $t+1$ and vector α define $Sub_t(\alpha, X)$ to be the sublock Y of X such that $pos_X(Y) = \alpha$.

Example. Figure 4 illustrates 3 nonblank regular blocks of rank 1 in segment R (see Figure 3) which correspond to the states of the automaton for R, each of these blocks is a part of a psuedo-regular block of rank 2 of T with the same vector α. We have $q5 = Sub(\alpha, s3, s3, \emptyset, s3)$, $q6 = Sub(\alpha, s3, s3, s3, \emptyset)$, $q7 = Sub(\alpha, \emptyset, \emptyset, s3, s3)$.

Lemma 3.1 *Assume R is a subsquare of T, then*
(1) $\tau(R) \leq \phi(T)$. **(2)** *For each t there is a vector α_t such that each regular block of rank t in the subimage R equals $Sub(\alpha_t, (A, B, C, D))$ where (A, B, C, D) is a basic block of T of rank $t+1$.*

Proof: It is enough to show the point (2). We proceed by induction on t. If the rank of R is k, then the lemma is clearly true. We suppose that it is true for some $t \leq k$ and prove it is true for $t-1$. Clearly, $\alpha_t < [2^t, 2^t]$. We assume that $\alpha_t \leq [2^{t-1}, 2^{t-1}]$, the other cases being treated similarly. Take any regular block B of rank t of R. By the induction hypothesis it equals $Sub(\alpha_t, (A, B, C, D))$ for some basic block (A, B, C, D) of T of rank $t+1$. Assume

$$A \rightarrow \begin{bmatrix} A_1 & A_3 \\ A_0 & A_2 \end{bmatrix} B \rightarrow \begin{bmatrix} B_1 & B_3 \\ B_0 & B_2 \end{bmatrix} C \rightarrow \begin{bmatrix} C_1 & C_3 \\ C_0 & C_2 \end{bmatrix} D \rightarrow \begin{bmatrix} D_1 & D_3 \\ D_0 & D_2 \end{bmatrix}.$$

Then the basic block (A, B, C, D) has the following structure:

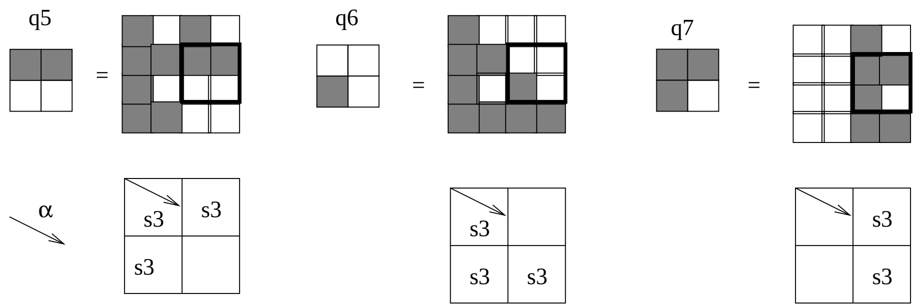

Figure 4: Regular blocks of R (in bold) are parts of basic blocks of T.

$$\begin{bmatrix} B_1 & B_3 & D_1 & D_3 \\ B_0 & B_2 & D_0 & D_2 \\ A_1 & A_3 & C_1 & C_3 \\ A_0 & A_2 & C_0 & C_1 \end{bmatrix}$$

Since $\alpha_t < [2^{t-1}, 2^{t-1}]$ the left upper corner of B is in B_1. Now we consider quadrants of B. The upper left one is

$$Sub(\alpha_t, (B_0, B_1, B_2, B_3)),$$

the upper right one is

$$Sub(\alpha_t, (B_2, B_3, D_0, D_1)),$$

the lower left one is

$$Sub(\alpha_t, (A_1, B_0, A_3, B_2))$$

and the lower rigth one is

$$Sub(\alpha_t, (A_3, B_2, C_1, D_0)).$$

Observe that the vector α_t is common to all psudo-regular blocks and is independent of the choice of a regular subblock B of R. This completes the proof. $\square$

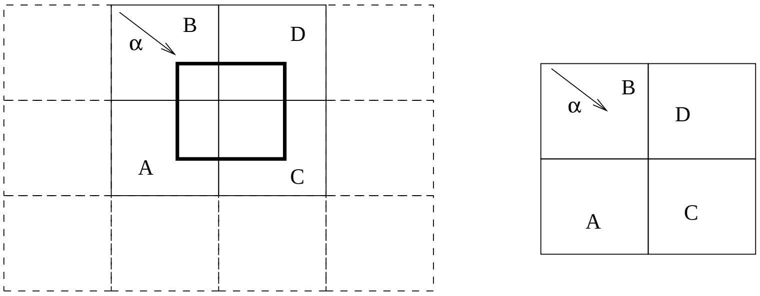

Figure 5: A regular block of the subimage is determined by the 5-tuple (α, A, B, C, D). The 4-tuple (A, B, C, D) corresponds to a basic block.

Lemma 3.2 *Let ϕ_i be the number of basic blocks of rank i for $1 \leq i \leq r$, and k_i be the number of regular blocks of rank i for $0 \leq i \leq r$. Then:*

$$\phi_i \;\leq\; \min(k_{i-1}^4,\; n^2)$$

Proof: The proof is based on the following fact, see Figure 6.

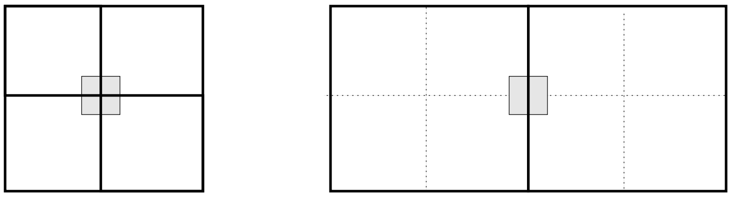

Figure 6: Illustration of the claim.

Claim. Each basic block of rank i is a central block of a regular block of rank k, or a central block of two (horizontally or vertically) adjacent regular blocks of rank k, where $k \geq i$.
Proof (of the claim) Consider a basic block B in T. Let

$$T \rightarrow \begin{bmatrix} A_1 & A_3 \\ A_0 & A_2 \end{bmatrix}$$

where A_0, A_1, A_2 and A_3 are regular in T. There are several possibilities.

Case 1. B is in the center of T. Then we are done.

Case 2. B is inside A_0 or A_1 or A_2 or A_3. Then we proceed recursively. Eventually, we come to one of the cases 1 and 3.

Case 3. B is in the common border of A_1 and A_3 or A_0 and A_2 or A_1 and A_0 or A_3 and A_2. Now either B is in the center of two adjacent regular blocks or we proceed recursively. Eventually, B will be a central block of two adjacent regular blocks, not necessarily of the same rank as B. Figure 7 illustrates a possible situation when B is in the border of A_1 and A_3. It is essential that B is a basic block, and it consists of four regular blocks. This completes the proof of the claim.

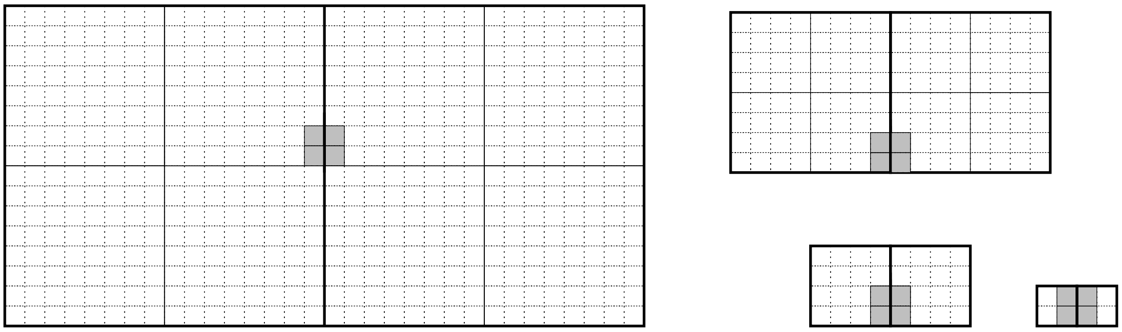

Figure 7: Case 3: a situation when B (shaded basic block) is in the border of A_1, A_3 but not in the center. Then B is in the border of two regular blocks of smaller ranks.

Due to the claim and inequality $k_i + k_{i+1} + \ldots k_r \ \leq n$ we have:

$$\phi_i \ \leq \ k_i^2 + k_{i+1}^2 + \ldots + k_r^2 + k_i + k_{i+1} + \ldots k_r \ \leq n^2.$$

On the other hand each basic block of rank i is composed of 4 regular blocks of rank $i-1$, hence $\phi_i \leq k_{i-1}^4$. Consequently $\phi_i \leq \min(k_{i-1}^4 \ n^2)$. $\square$

We omit the proof of the following simple lemma which says that the maximum of the sum of 4-th powers of nonegative numbers bounded by a certain number δ is the largest when all these number are equal δ except possibly one number.

Lemma 3.3
Assume $0 < \delta < n$, $0 < k$, $0 \leq \alpha_i \leq \delta$ *for* $1 \leq i \leq k$ *and* $\sum_{i=1}^k \ \alpha_i \ = \ n$. *Then*

$$\sum_{i=1}^k \ \alpha_i^4 \ \leq \ \frac{n}{\delta} \cdot \delta^4.$$

Theorem 3.4 $\Psi(n) \leq n^{2.5}$.

Proof: Due to Lemma 3.1 it is enough to show $\phi(T) \leq \tau(T)^{2.5}$. The conclusion of the theorem is now a consequence of Lemma 3.2 and of the following fact.

Claim. If $n_1 + n_2 + \ldots + n_k = n$, then $S = \sum_{i=1}^{k} \min\{n_i^4, n^2\} \leq n^{2.5}$

Proof (of the claim)

Let $\delta = \sqrt{n}$, $A = \{i : n_i \leq \delta\}$, and $B = \{i : n_i > \delta\}$. Then:

$$\sum_{i=1}^{k} \min\{n_i^4, n^2\} \leq \sum_{i\in A} n_i^4 + \sum_{i\in B} n^2.$$

Denote $x = \sum_{i=1}^{k} n_i$. Then:

$$\sum_{i\in A} n_i^4 \leq \frac{x}{\delta}\delta^4$$

due to Lemma 3.3 and:

$$\sum_{i\in B} n^2 \leq |B| \cdot n^2 \leq \frac{n-x}{\delta} \cdot n^2$$

Putting these inequalities together we have:

$$S \leq \frac{x}{\sqrt{n}}\sqrt{n}^4 + \frac{n-x}{\sqrt{n}} \cdot n^2 \leq \frac{n}{\sqrt{n}} \cdot n^2 \leq n^{2.5}.$$

This completes the proof. □

4 $\Psi(n) \geq c \cdot n^{2.5}$

In the proof of the lower bound we first need to generate two objects of sizes $O(n)$ which composed together give an object whose compression size is $\Omega(n^2)$. The construction is based on morphic representations of strings. Assume that the alphabet consists of nonnegative integers and we have morphisms h and g defined by

$$h(i) = 2i\ 2i\ 2i+1\ 2i+1 \text{ and } g(i) = 2i\ 2i+1\ 2i\ 2i+1, \text{for } i = 0, 1 \ldots.$$

Example. The generation of

$$h^2(0) = 0\ 0\ 1\ 1\ 0\ 0\ 1\ 1\ 2\ 2\ 3\ 3\ 2\ 2\ 3\ 3$$
$$g^2(0) = 0\ 1\ 0\ 1\ 2\ 3\ 2\ 3\ 0\ 1\ 0\ 1\ 2\ 3\ 2\ 3$$

is illustrated in Figure 8.
Observe that the set of pairs $(h^2(0)[i], g^2(0)[i])$ for $i \leq 16$ is exactly the set $\{(s,t) : 0 \leq s \leq 3, 0 \leq t \leq 3\}$.

For a 1-dimensinal string w whose length is a power of two, define *1-regular blocks* similarly as (2-dimensional) regular blocks for 2-dimensional texts. Both halves of w are regular, and recursively both halves of them are also. Denote by $\tau_1(w)$ the number of distinct 1-regular blocks of a 1-dimensional string.

Lemma 4.1 [key-lemma]
Assume $n = 2^k$ for a natural k. Then there are two strings u_n, w_n of size n^2 over the alphabet $\{0, 1, \ldots, n-1\}$ such that:
(1) *All n^2 pairs $(u_n[i], w_n[i])$ are different for $1 \leq i \leq n^2$.*

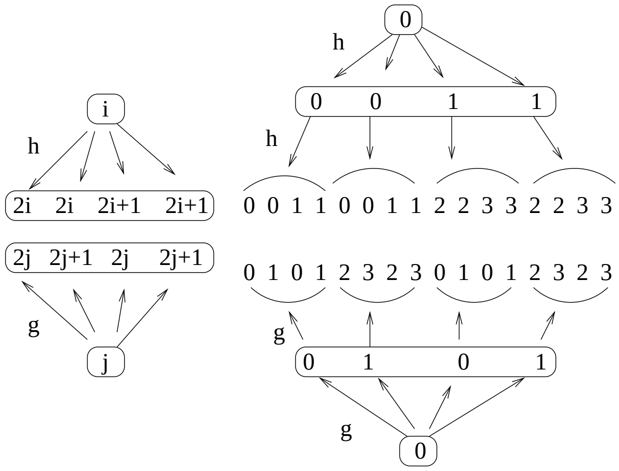

Figure 8: The generation of $h^2(0)$ and $g^2(0)$.

(2) $\tau_1(w_n),\ \tau_1(u_n)\ \le\ 6n.$

Proof:
Take $u_n = h^k(0)$ and $w_n = g^k(0)$.

Proof of point (1). We proceed by induction on k. For $k = 0$ this is clear. Suppose the lemma is true for $k > 0$. We will prove that it holds for $k+1$. Take any pair (a, b) such that $0 \le a, b < 2^{k+1}$. The numbers a, b can be uniquely written in the form $a = 2a_1 + a_2$ and $b = 2b_1 + b_2$, where $0 \le a_1, b_1 < 2^k$ and $0 \le a_2, b_2 \le 1$. By inductive hypothesis $a_1 = h^k(0)[i]$ and $b_1 = g^k(0)[i]$ for some $0 \le i < 2^k$. Then,

$$\begin{array}{rcl} h^{k+1}(0)[4i] = 2a_1 + 0 & , & g^{k+1}(0)[4i] = 2b_1 + 0 \\ h^{k+1}(0)[4i+1] = 2a_1 + 0 & , & g^{k+1}(0)[4i+1] = 2b_1 + 1 \\ h^{k+1}(0)[4i+2] = 2a_1 + 1 & , & g^{k+1}(0)[4i+2] = 2b_1 + 0 \\ h^{k+1}(0)[4i+3] = 2a_1 + 1 & , & g^{k+1}(0)[4i+3] = 2b_1 + 1. \end{array}$$

Hence, among pairs $(h^{k+1}(0)[j], g^{k+1}(0)[j])$, for $4i \le j < 4i+4$, there is a pair (a, b). Since the length of the word $h^{k+1}(0)$ is $(2^{k+1})^2$ each pair (a, b) with $0 \le a, b < 2^{k+1}$ occurs among the pairs $(h^{k+1}(0)[i], g^{k+1}[i])$ exactly once.

Proof of point (2). It can be easily seen that each 1-regular block of u_n is of the form
(*) $\qquad h^s(a)$ for $0 \le a < 2^{k-s}$,
or it is one of the halves of a word of this type.

There are 2^{k-s} words of type (*). Summing over $0 \le s \le k$ we have at most $2n$ such words. Together with their halves we have an upper bound of $2n + 2 * 2n \ = \ 6n$ on the number of 1-regular blocks. Similarly we proceed for the string w_k. This complete the proof. $\square$

4.1 Construction of the image T_n

For each n, which is a power of two, we construct an image T_n as illustrated in Figure 9. It consists of two vertical central columns being sequences of *medium* subsquares

$L_{i_1}, L_{i_2}, \ldots L_{i_m}$ and $R_{j_1}, R_{j_2}, \ldots R_{j_m}$ where:

$$i_1 i_2 \ldots i_m \;=\; u_n \text{ and } j_1 j_2 \ldots j_m \;=\; w_n.$$

Everything outside medium subsequares is blank, also most of medium subsqures is blank, but the size of a *large* blank area will matter later.

Let $r = \log(\sqrt{n})$, assume it is a power of two, otherwise we take the closest (from above) power of two. Each *medium* subsqure L_i is a $2^{\sqrt{n}}r \times 2^{\sqrt{n}}r$ square which is blank except two small subsquares $Small(p),\ Small(q)$, see Figure 10, where $(p,q) = code(i)$, and *code* is any injective function mapping the interval $[1 \ldots n]$ into the set of pairs $[1 \ldots \sqrt{n}] \times [1 \ldots \sqrt{n}]$. Each *small* subsquare $Small(p)$ is a $r \times r$ black-white square whose first row encodes the number p and other rows are blank. In fact any family of small subsquares encoding natural numbers not exceeding $\sqrt{n}$ will do.

These small subsquares ($Small(p),\ Small(q)$ subsquares) are called *code-blocks.* Two code-blocks of L_i (R_i) encode together the number i.

Lemma 4.2 $\tau(T_n) \;=\; O(n)$

Proof: Let s, r be the lengths of the sides of medium blocks (one of $L_i,\ R_j$) and code-blocks.

Denote by $\tau_{small}(n),\ \tau_{large}(n),\ \tau_{others}(n)$ the number of different regular nonblank blocks of T_n whose sides have length at most r, greater than s and other, repectively. Observe that all pixels of T_n are blank except of the ones contained in code-blocks.

Each small code-block is of a polylogarithmic size, hence:

$$\tau_{small}(n) \;=\; o(n) \tag{1}$$

Each nonblank regular block whose side length is between r and s has side of length at most $r2^{\sqrt{n}}$ which is a power of two. There are $O(\sqrt{n})$ possible lengths. Each such block is blank except one corner containing a code-block. There are $O(\sqrt{n})$ such subsquares. Hence:

$$\tau_{others}(n) \;=\; O(n) \tag{2}$$

Each regular block of T_n whose side length is greater than s consists of a blank area and a segment of vertical column of medium subsquares which corresponds to a 1-regular block of u_n or w_n. Consequently, due to Lemma 4.1 we have:

$$\tau_{large}(n) \;=\; O(n) \tag{3}$$

Due to Equations (1), (2) and (3) we have :

$$\tau(T_n) \;=\; \tau_{small}(n) + \tau_{large}(n) + \tau_{others}(n) \;=\; O(n)$$

This completes the proof. □

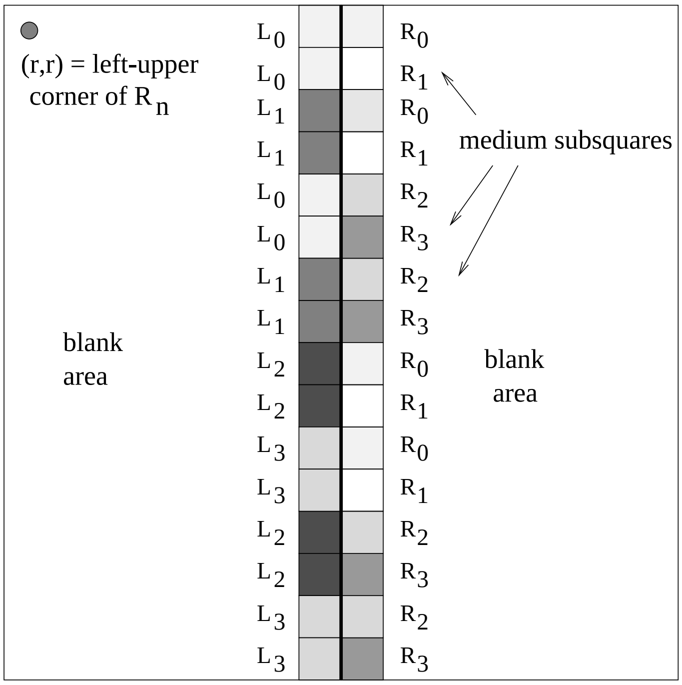

Figure 9: The structure of the image T_n, for $n = 4$, of shape $(s \cdot 2n^2) \times (s \cdot 2n^2)$.

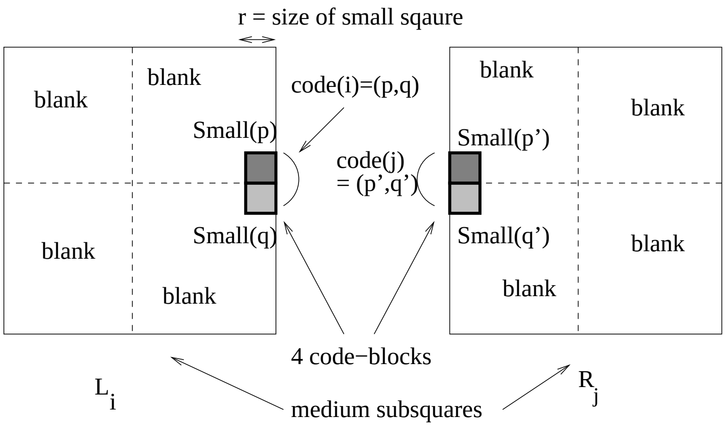

Figure 10: The structure of the medium $s \times s$ subsquares L_i, R_j, where $s = r \cdot 2^{\sqrt{n}}$. They are totally blank except the indicated $r \times r$ code-blocks.

4.2 Construction of the subimage $\mathcal{R}_n$

Let $\mathcal{R}_n$ be the square subsegement of T_n whose side is twice smaller than the side of T_n and which starts (with its left-upper corner) in T_n at the position (r, r).

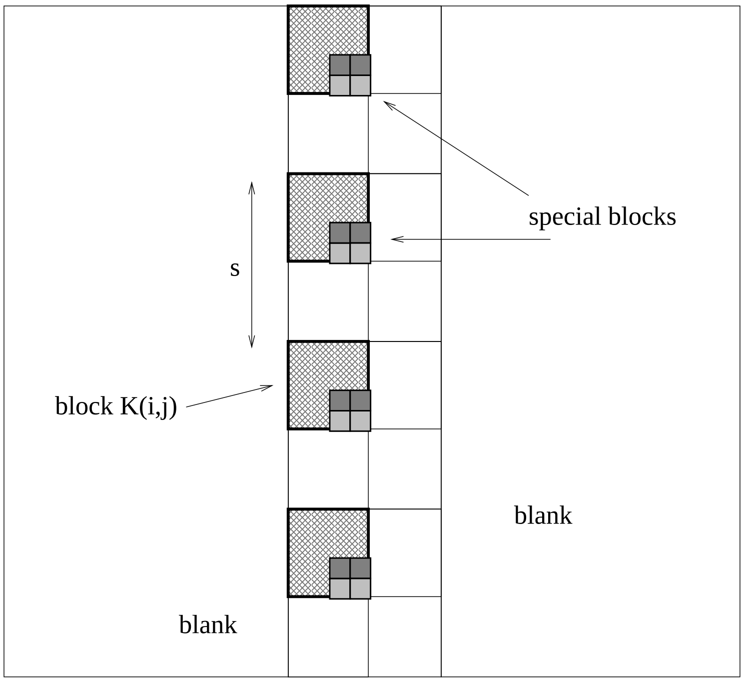

Figure 11: The structure of the subsegment $\mathcal{R}_n$. All elements are blank except special blocks, each of them is composed of four small *code-blocks*.

Lemma 4.3 $\tau(\mathcal{R}_n) = \Omega(n^{2.5})$

Proof: After placing the subimage with the origin at point (r, r) the grid of regular blocks is shifted by the vectore (r, r), see Figure 11 and Figure 12. Let us define *special* blocks as $2r \times 2r$ basic blocks of T_n composed of 4 adjacent code-blocks. The first half of the vertical sequence of special blocks of T_n become now regular blocks of $\mathcal{R}_n$. After shifting the grid each special block is a right-bottom *corner block* of a regular $\frac{s}{2} \times \frac{s}{2}$ block $K(i, j)$, see Figure 12. The block $K(i, j)$ overlaps adjacent blocks L_i and R_j in T_n, it is of the same shape as each of them. $K(i, j)$ can be viewed as a shift by the vector (r, r) of top-right quadrant of L_i, see Figure 12. Denote by $\mathcal{K}$ the set of all such blocks $K(i, j)$. Due to Lemma 4.1 there are $\Omega(n^2)$ special blocks and they are pairwise different, so $\mathcal{K}$ satisfies:

- **(1)** $|\mathcal{K}| = \Omega(n^2)$,
- **(2)** each block in $\mathcal{K}$ is of the shape $\frac{1}{2}r2^{\sqrt{n}} \times \frac{1}{2}r2^{\sqrt{n}}$,
- **(3)** the $2r \times 2r$ right-bottom corners of blocks in $\mathcal{K}$ are pairwise different.
- **(4)** Each block of $\mathcal{K}$ is a regular block in $\mathcal{R}_n$.

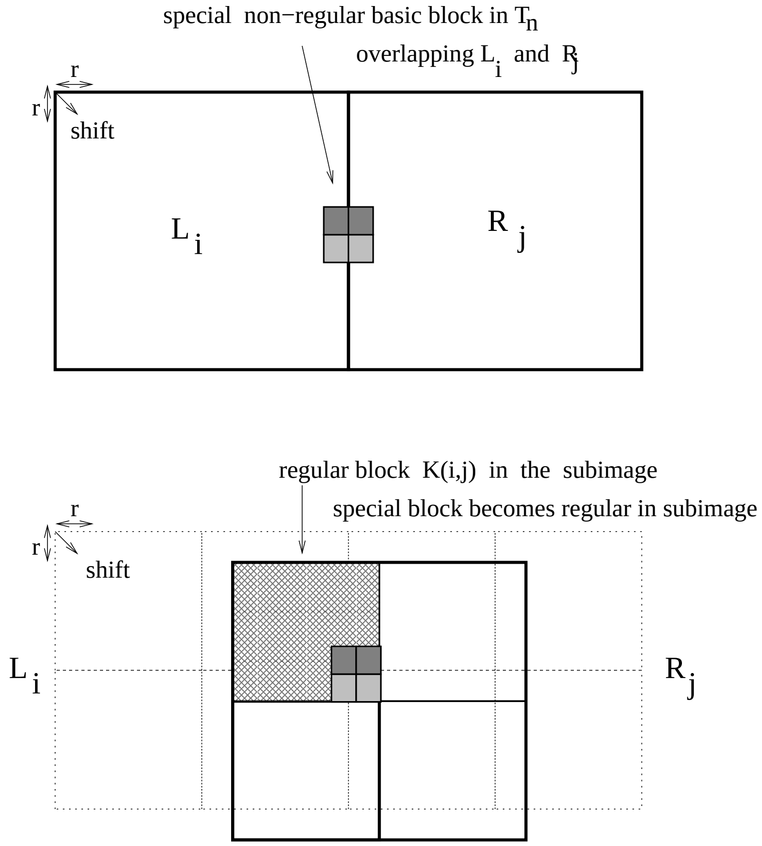

Figure 12: The regular blocks $K(i,j)$ in $\mathcal{R}_n$ result by shifting the *structure* of regular blocks in T_n by the vector (r,r) over a pair of adjacent medium blocks $L_i,\ R_j$.

In this moment we have shown that we have a set $\mathcal{K}$ of $\Omega(n^2)$ different regular blocks, now we show that we have in fact $\Omega(n^{2.5})$ such blocks.

For each regular block $K(i,j) \in \mathcal{K}$ we consider also all regular sub-blocks of $K(i,j)$ which contain the special block of $K(i,j)$. Denote the set of these blocks by $\mathcal{K}'$. More formally we can define K' as the smallest set of regular blocks of $\mathcal{R}_n$ satisfying: all blocks in $\mathcal{K}$ are in $\mathcal{K}'$, and for every block B in $\mathcal{K}'$ its bottom-right quadrant is also in $\mathcal{K}'$, if it contains (or equals) the special block of B. The basic argument is that the size of each $K(i,j)$ is $\Omega(\sqrt{n})$ larger than the size of a special block. We have $\Omega(\sqrt{n})$ different sizes of regular sub-blocks in $\mathcal{K}'$. Each block in $\mathcal{K}'$ is regular and these blocks are pairwise different because they differ with respect to the special block (which they contain) or with respect to the size.

We have $\Omega(n^2)$ different special blocks and $\Omega(\sqrt{n})$ sizes. Altogether $|\mathcal{K}'| = \Omega(\sqrt{n} \cdot n^2)$. Hence $\tau(\mathcal{R}_n) = \Omega(n^{2.5})$. This completes the proof. $\square$

The following theorem is a direct consequence of the last two lemmas.

Theorem 4.4 $\Psi(n) \geq c \cdot n^{2.5}$.

5 Construction of compressed representation of subimages for deterministic automata

Each basic block X is identified by a composite name

$$(name(X_1), name(X_2), name(X_3), name(X_4)),$$

where $X_1, \ldots X_4$ are regular blocks which are quadrants of X, listed in the order corresponding to Figure 1.

The compressed size of the subsegment can grow since the number of basic blocks could be much larger than the number of regular blocks in the same image. For example, there are 2 regular blocks of rank 1 in T in Figure 2, but 8 different basic blocks of rank 1 (including blank ones), which have the following composite names:

$$\begin{array}{llll}\{(s4, s4, s4, \hat{\emptyset}), & (s4, \hat{\emptyset}, \hat{\emptyset}, \hat{\emptyset}), & (\hat{\emptyset}, \hat{\emptyset}, \hat{\emptyset}, \hat{\emptyset}), & (s4, \hat{\emptyset}, s4, s4), \\ (s4, s4, \hat{\emptyset}, s4), & (\hat{\emptyset}, s4, \hat{\emptyset}, s4) & (\hat{\emptyset}, s4, s4, \hat{\emptyset}) & (\hat{\emptyset}, \hat{\emptyset}, s4, s4)\}.\end{array}$$

Theorem 5.1
If the compression is in terms of deterministic automata, then the compressed representation of a square subsegment R of T can be computed in $O(|Compress(T)|^{2.5})$ time.

Proof: Due to Lemma 3.1 for each rank t there is a vector α_t such that each regular block of rank t in the subimage R equals $Sub(\alpha_t, (A, B, C, D))$ where (A, B, C, D) is a basic block of T of rank $t + 1$.

The states of the automaton A' for the subsegment are tuples $(\alpha_k, (A, B, C, D))$ where (A, B, C, D) are basic blocks of T of rank $k + 1$, and α_k is a vector from Lemma 3.1. Due to Theorem 3.4 the number of basic blocks is $O(n^{2.5})$, where $n = |Compress(T)|$.

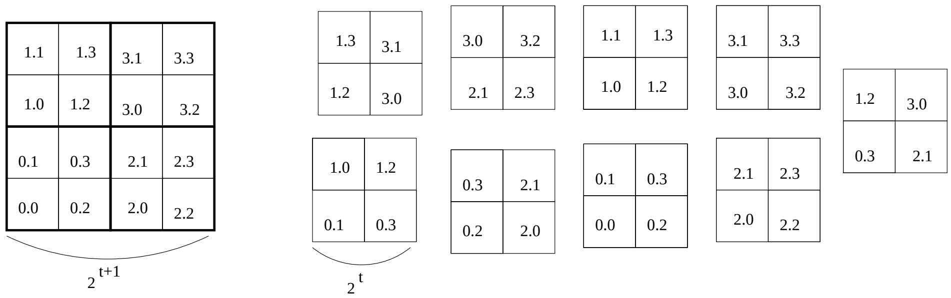

Figure 13: The name of each basic block of rank t is derived from the name of a basic block of rank $t + 1$. The t-basic blocks which are not quadrants of $t + 1$-basic blocks are shown.

We compute names of basic blocks in a *top down* way. this means that we construct tuples (α_t, A, B, C, D) on the basis of the tuples $(\alpha_{t+1}, A, B, C, D)$.

We consider only these basic blocks which contain a nonblank block of the subsegment. If we know the names of basic blocks of rank $t+1$ then each basic block of rank t consists of 4 sub-blocks of rank t of a single basic block of rank $t+1$, see Figure 13. Hence the number

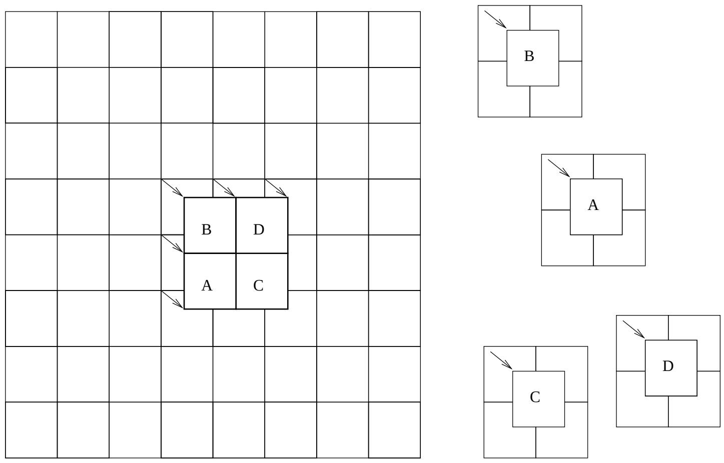

Figure 14: Initialization of the computation of names for the work blocks of P.

of generated tuples is is of the same order as the time complexity of our algorithm. The number of generated tuples is at most the number of basic blocks of T which is bounded by $O(n^{2.5})$.

We need also to initialize the construction by finding the smallest basic block of T containing the subsegment R, see Figure 14.

The complexity is dominated by the number of regular blocks which are constructed for R, which, due to Theorem 3.4, is $O(n^{2.5})$. Hence the time complexity of the whole construction is $O(n^{2.5})$. $\square$

6 The Subsegment Compression Problem for Weighted Automata

For weighted automata, contrary to deterministic ones, the compressed size of the subsegment can grow only linearly. This surprising phenomenon is due to the fact that for weighted automata we can have many edges from the same state labeled with the same symbol, but having possibly different weights. This enables to do operations similar to matrix addition, such a trick is not possible in the deterministic case.

A weighted finite automaton describing an image is specified by (see [9] for details):

- the set of states Q,
- the alphabet $\{0, 1, 2, 3\}$,
- weight of edges given by the function $W_a : Q \times Q \to (-\infty, \infty)$ for edges labeled by the symbol a, for $a \in \{0, 1, 2, 3\}$,
- a function $I : Q \to (-\infty, \infty)$ called the *initial distribution function* and

- a function $F : Q \to (-\infty, \infty)$ called the *final distribution function.*

The weight of a word $w = a_1 a_2 \ldots a_k$ is interpreted to give a color $W(w)$ for the pixel $entry(w)$. It is defined as $W(w) = IW_{a_1} W_{a_2} \ldots W_{a_k} F$.

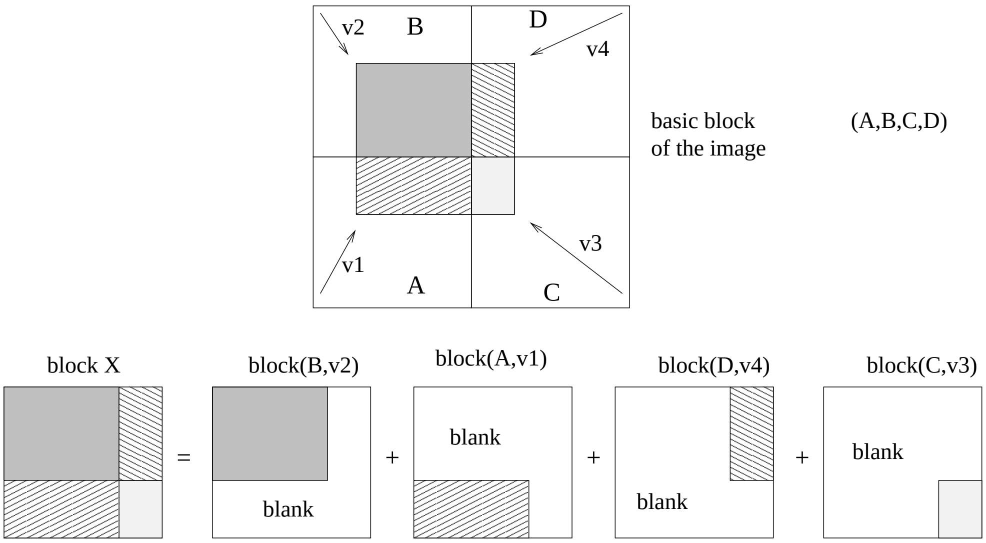

Figure 15: Regular block X cut off from a basic block of T (as a matrix) can be treated as the sum: $X = block(A, v1) + block(B, v2) + block(C, v3) + block(D, v4)$.

Our result for weighted automata is as follows.

Theorem 6.1
Let A be a weighted automaton defining T and having n states and m edges, and let R be a subsegment of T. Then there is a weighted automaton A' which defines R and has $O(n)$ states and $O(m)$ edges. The Subsegment Compression Problem for weighted automata for a square subsegment R can be computed in linear time.

Proof: For a regular block Y of T and a vector v, denote by $block(Y, v)$ the square which is identical to Y on the overlap of Y with the square of the same shape shifted from the corner of Y inside Y by the vector v, all other entries are blank (contain zeros), see Figure 15.

We identify states of A with regular blocks. For each rank t a regular block of rank t of a subsegment is located in a basic block of T, where vectors $v1, v2, v3, v4$ are as in Figure 15.

We create the automaton A' for R. Its states are identified with $block(Y, v)$, where Y is a regular block of T and v is one of the vectors $v1, \ldots, v4$ which depend on the rank. The correctness is based on the fact that the subblock X can be treated as a matrix and it is the sum of 4 matrices, as shown in the figure. For each rank there are 4 different vectors, so the number of states of A' is $O(n)$. The construction goes top-down similarly as in Theorem 5.1 using the ideas from the proof of Theorem 4.2 in [11], where *atomic dependencies* are decomposed into smaller ones. In our case atomic dependencies correspond to sub-blocks of the type $block(Y, v)$. □

7 Conclusions.

We have shown tight bounds for the compression sizes of subimages of images compressed in terms of deterministic and weighted automata, moreover we have shown how to construct efficiently compressed representation of subimages. A first simple applications is an improvement (and simplification) upon a similar result in [11], and the second one gives the first polynomial time algorithm for the compressed checking problem for weighted automata.

Fact 7.1 *There is an algorithm for compressed pattern-matching for deterministic automata working in time $O(n^2(m+\sqrt{n}))$ where n is the compressed representation of T and m is the total size of an uncompressed pattern P.*

Proof: In the construction of the representation for a subsegment we compute basic blocks. Clearly, P occurs in T if it occurs in a basic block of rank $t+1$, where t is the rank of P. We can construct all basic blocks of rank $t+1$ in time $O(n^{2.5})$ using top down algorithm which constructs a set of basic blocks of rank i on the basis of the set of basic blocks of rank $i+1$. By Lemma 3.2 there are $O(n^2)$ basic blocks of each rank in particular of rank $t+1$. For each of them we check (by known linear time algorithms for uncompressed two-dimensional matching) if P occurs in one of those. It costs $O(n^2m)$ time. $\square$

Fact 7.2 *There is a polynomial time algorithm for the fully compressed checking problem for weighted automata.*

Proof: It follows directly from the proof of Theorem 8.2 in [6] that equivalence of two weighted automata can be checked in polynomial time. We can construct the compressed representation of the subsegment R of T which is of the same shape as the pattern P and starts at the same location. Then we check the equality of the images R, P in polynomial time using the polynomial time test for the equivalence of two weighted automata. $\square$

References

[1] A. Amir and G. Benson, Efficient two dimensional compressed matching, *Proc. of the 2nd IEEE Data Compression Conference* 279-288 (1992).

[2] P. Berman, M. Karpinski, L. Larmore, W. Plandowski, W. Rytter, On the complexity of pattern matching of highly compressed two-dimensional texts, in *Proc. CPM'97*, LNCS 1264, 40-51, 1997.

[3] K. Culik and J. Karhumäki, Finite automata computing real functions, *SIAM J. Comp.* **23**(4), 789-814, 1994.

[4] K. Culik and J. Kari, Image compression using weighted finite automata, *Computer and Graphics* 17, 305-313 (1993).

[5] D. Derencourt, J. Karhumäki, M. Latteux and A. Terlutte, On continuous functions computed by finite automata, *RAIRO Theor. Inform. Appl.* 28, 387-404 (1994).

[6] S. Eilenberg, *Automata, Languages and Machines*, Vol.A, Academic Press, New York (1974).

[7] K. Culik and J. Kari, *Fractal image compression: theory and applications*, (ed. Y. Fisher), Springer Verlag 243-258 (1995).

[8] M. Farach and M. Thorup, String matching in Lempel-Ziv compressed strings, in *STOC'95*, pp. 703-712.

[9] J. Kari, P. Franti, Arithmetic coding of weighted finite automata, *RAIRO Theor. Inform. Appl.* 28 343-360 (1994).

[10] L. Gąsieniec, M. Karpiński, W. Plandowski and W. Rytter, Efficient algorithms for Lempel-Ziv encoding (extended abstract), in *Proc. SWAT'96*, LNCS 1097, 392-403, 1996.

[11] J. Karhumäki, W. Plandowski, W. Rytter, Pattern matching for images generated by finite automata, *Nordic Journal of Computing* (to appear), see also in *Proc. FCT'97*, LNCS 1279, 245-256, 1997.

[12] J. Karhumäki, W. Plandowski, W. Rytter, The compression of subsegments of images described by finite automata, in *Proc. CPM'99*, LNCS 1645, 186-195.

[13] M. Karpinski, W. Rytter and A. Shinohara, Pattern-matching for strings with short description, in *Proc. CPM'95*, LNCS 937, 205-214, 1995.

[14] W. Rytter, Compressed and fully compressed pattern-matching, *Proceedings of IEEE*, November 2000, Volume 88, Number 11, pp. 1769-1778