Optimizing Markov Models with Applications to Triangular Connectivity Coding

Stefan Gumhold*

Abstract

In this work Markov Models are constructed to describe the asymptotic stochastical behavior of regular languages, what allows for optimal arithmetic coding of words from the language. A new method is presented for the optimization of Markov Models such that also constraints are captured that cannot be described within a regular language. The new technique is applied to the encoding of the connectivity graph of triangle meshes of low genus and boundary fraction. The resulting compression rates are up to one percent optimal and the best known upper bound for this class of models.

1 Introduction and Overview

Arithmetic coding [28] became very popular in information theory and coding application because of its optimality. If the statistical properties of the letters in a word from a given language can be expressed in a probability model that is suitable for an arithmetic coder a simple enumeration argument can be used to show the optimality of the arithmetic coder in the asymptotic limit for very long words. The enumeration argument leads to an information theoretical lower bound, which is typically called the "entropy" of the language and specified in bits per symbol (bps).

In section 3 an algorithm is developed for the construction of asymptotically optimal probability models for strings from a regular language. The construction is based on a regular language given in form of a deterministic finite automaton (DFA). It is well known that non-deterministic finite automatons (NFA) can be converted into deterministic ones, although the number of states can explode exponentially and asymptotic counting becomes $\#P$-complete [15]. A construction of a Markov Model from a DFA similar to the one presented in section 3 has been described by Marcus et al. in the context of constrained systems [20]. A constrained system is used to store recorded information in binary codes, which obey constraints that can be expressed in a regular language. The Markov Model is used to map information to the constrained codes in a way that optimally exploits the channel capacity of the constrained system.

An important contribution in section 3 is a new proof for the optimality of the constructed probability models. This proof can be directly generalized to the setting of section 4, where the probability model is optimized for further constraints on the symbol counts of the words from the regular language. These constraints cannot be implemented with a regular grammar but are important for a lot of coding applications including the one developed in sections 5 and 6.

The proposed optimization technique is applied to the encoding of the connectivity graph of triangle meshes over surfaces of low genus and low boundary fraction. A slightly modified version of the Edgebreaker [22] algorithm as introduced in section 5 is used to transform the connectivity of a triangle mesh into a string over a five-symbol alphabet.

A complete set of constraints on the symbol string can be derived from the Edgebreaker algorithm [9]. Although the constraints can be shown to be complete, previous work [9] demonstrated that they cannot be captured within a regular language - even with a huge number of states. Section 6 applies the optimization method of section 4 to the Edgebreaker constraints resulting in a coding performance that is only one percent above the information theoretical lower bound.

The coding of triangular mesh connectivity is an especially interesting application as a lot of recent work in the computer graphics community [5, 10, 19, 25, 24, 22, 17, 2, 14, 12, 1, 18, 13] as well as in the graph coding community [26, 16, 4, 21] has been devoted to it. Denny and Sohler [6] showed that the connectivity graph of a planar triangulation can be efficiently encoded in a permutation of its vertices. This approach has not been generalized to meshes of higher genus and is based on the knowledge of the vertex locations. Most mesh coding applications also compress the geometric information in the vertex locations. All of these schemes are based on special orderings of the vertices and cannot be combined with the method of Denny and Sohler.

For the special case of planar triangulations or equivalently triangle meshes of genus zero the informa-

*Max-Planck-Institut für Informatik Saarbrücken.

tion theoretical lower bound of $\log_2 \frac{256}{27} \approx 3.245$ bits per vertex has been known since 1962 from the work of Tutte [27]. Algorithmic upper bounds have been constantly improving in the last years. The idea of Turan [26] to encode a planar connectivity graph by a vertex and a triangle spanning tree has been applied to the purely triangular case by Taubin and Rossignac [24] in their topological surgery method, for which an upper bound of 6 bits per vertex can be proven. Gumhold [11] established an upper bound of 4.92 bits per vertex for the Cut-Border Machine encoding scheme. The original Edgebreaker encoding scheme by Rossignac [22], on which also this work is based, allows a simple proof of a bound with 4 bits per vertex. Chuang et al. [4] also established an algorithmic upper bound of 4 bits per vertex. King and Rossignac [17] improved the Edgebreaker coding to 3.667 bits per vertex and Gumhold [11] to 3.5 bits per vertex. An interesting observation is due to Alliez and Desbrun [1]: the entropy of the vertex valences coincides with Tutte's lower bound and therefore would the valence based encoding scheme of Touma and Gotsman [25] be optimal if no split codes would arise. This statement was revised by Gotsman in [7], where he showed that the valence entropy is slightly below Tutte's bound and a linear number of split codes is necessary in the general case. Only recently Poulalhon and Schaeffer [21] proposed an optimal coding scheme for planar triangulations, which is based on a bijection between planar triangulations and trees, where each interior node has exactly two children. But the method does not easily generalize to triangle meshes of higher genus and with boundary loops such as the proposed method.

The contributions of the proposed work are

- an optimization scheme for Markov Models, which allows to incorporate constraints that cannot be expressed with a regular language,
- a coding scheme for triangular meshes of low genus and with small boundary fraction, which is only one percent above the information theoretical lower bound and the best known result for non planar triangulations.

The Cut-Border Machine [8] and the Edgebreaker scheme [23] have been improved for the regular case of meshes with a large number of valence six vertices. Szymczak et al. [23] derived a formula for the upper bound in dependence on the fraction of valence six vertices. For a sufficiently large fraction they achieve an upper bound of 1.622 bits per vertex. In future work it is planed to also generalize the proposed approach to the setting of regular meshes.

2 Preliminaries

For a given not necessarily regular language $\mathcal{L}$ over an alphabet $\mathcal{A}$ all words of length m are called *m-slice* and abbreviated by $\mathcal{L}_m$. With the number of words $|\mathcal{L}_m|$ in a slice the asymptotic lower bound β is defined in bits per symbol

$$\beta \stackrel{\text{def}}{=} \log_2 \left(\sup_{m\to\infty} \frac{|\mathcal{L}_m|}{m} \right) \text{ bps.}$$

A coding scheme $\mathcal{C}$ for the language maps each word $\omega \in \mathcal{L}$ to a binary code $\mathcal{C}(\omega) \in \{0,1\}^*$. The asymptotic upper bound $\mathcal{B}$ of $\mathcal{C}$ is defined as

$$\mathcal{B} \stackrel{\text{def}}{=} \sup_{m\to\infty} \frac{\max_{\omega\in\mathcal{L}_m} |\mathcal{C}(\omega)|}{m} \text{ bps.}$$

A coding scheme is *asymptotic optimal* for $\mathcal{L}$, iff $\beta(\mathcal{L}) = \mathcal{B}(\mathcal{C})$.

An arithmetic coder maps a word $\sigma_1\sigma_2\ldots\sigma_m$ to a sub-interval $I \subset [0,1)$, which is encoded as the shortest binary fraction that defines an interval contained completely in I. Interval I is determined by iterated subdivision of $[0,1)$ according to the probabilities of the symbols. The symbol probability $P(\sigma_i)$ can depend on all previous symbols: $P(\sigma_i|\sigma_1\ldots\sigma_{i-1})$. This dependency is called the probability model and the arithmetic coder is known to be asymptotic optimal for languages in correspondence with their probability model.

A DFA $\mathcal{D}$ over $\mathcal{A}$ is a quadruple $(\mathcal{S}, S_0, \mathcal{T}, \tau)$ of a set of states $\mathcal{S}$, an initial state $S_0 \in \mathcal{S}$, a set of terminal states $\mathcal{T} \subset \mathcal{S}$ and a set of transitions $\tau \subset \mathcal{S} \times \mathcal{A} \times \mathcal{S}$, each of which is composed of a start state A, a symbol σ and an end state B and also abbreviated by $A \xrightarrow{\sigma} B$. For the DFA all outgoing transitions of a state have distinct symbols. The set of states $\mathcal{S}$ together with the transitions form a directed graph, where each directed edge is attributed by a symbol from $\mathcal{A}$. Each path $(A_1 \xrightarrow{\sigma_1} A_2, A_2 \xrightarrow{\sigma_2} A_3, \ldots, A_m \xrightarrow{\sigma_m} B)$ in the graph is identified with the partial word $\sigma_1\sigma_2\ldots\sigma_m$. The set of all paths from A to B of length m is denoted by $A \stackrel{m}{\Rightarrow} B$. If one of the arguments to $.\stackrel{m}{\Rightarrow}.$ is replaced by a set of states, the union over all states is implied. The m-slices of the regular language $\mathcal{L}$ defined by $\mathcal{D} = (\mathcal{S}, S_0, \mathcal{T}, \tau)$ are identified in this notation with $\mathcal{L}_m \simeq S_0 \stackrel{m}{\Rightarrow} \mathcal{T}$. A simple example of a DFA is shown in Figure 1.

A Markov Model $\mathcal{M}$ is a probability model that extends a deterministic state machine $\mathcal{D}$ by a transition probability $p(A \xrightarrow{\sigma} B)$ for each transition, where for each state the probabilities of the outgoing transitions must sum to one

$$\forall A \in \mathcal{S} : \sum_{B\in\mathcal{S},\sigma\in\mathcal{A}} p(A \xrightarrow{\sigma} B) = 1. \tag{2.1}$$

By the use of a Markov Model one assumes (Markov Assumption) that probability $P(\sigma_i|\sigma_1 \dots \sigma_{i-1})$ depends only on the state reached by the state machine after seeing $\sigma_1 \dots \sigma_{i-1}$. An arithmetic coder based on a Markov Model is asymptotic optimal for languages that fulfill the Markov Assumption. Upper and lower bounds coincide with the entropy of the Markov Model

$$\mathcal{H} = -\sum_{A\in\mathcal{S}} P(A) \sum_{B\in\mathcal{S},\sigma\in\mathcal{A}} p(A\xrightarrow{\sigma}B)\log_2 p(A\xrightarrow{\sigma}B),$$

with the state probabilities $P(A)$ of the state machine being in state A. The transition matrix $\mathbf{T}$ with $\mathbf{T}_{AB} = \sum_\sigma p(A\xrightarrow{\sigma}B)$ allows to compute the state probabilities as the left eigenvector to eigenvalue one, i.e. $\vec{P}_\mathcal{S} = \vec{P}_\mathcal{S}\mathbf{T}$, where $\vec{P}_\mathcal{S}$ is the vector of all state probabilities.

The adjacency matrix $\mathbf{A}$ of a deterministic state machine is defined with entries $\mathbf{A}_{AB} = \left|\left\{A\xrightarrow{\sigma}B\right\}_\sigma\right|$, which count the number of transitions/edges from A to B. The adjacency graph of a deterministic state machine is strongly connected, iff for any two states A, B there is a path from A to B. To simplify proofs we restrict ourselves in the following to strongly connected graphs. For these the theory of Perron and Frobenius states

Theorem 2.1. (Perron-Frobenius) *The largest eigenvalue α of the adjacency matrix $\mathbf{A}$ of a strongly connected graph fulfills*

1. *α is positive, unique and all components of the corresponding eigenvector $\vec{r}$ are strictly positive.*

2. *α has the largest absolute value among all eigenvalues.*

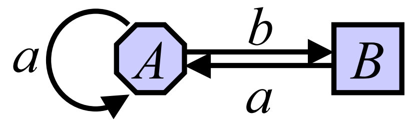
3 Optimal Markov Models for Regular Languages

The Markov Model design problem for a given DFA $\mathcal{D} = (\mathcal{S}, S_0, \mathcal{T}, \tau)$ is to find the transition probability function $p(\tau)$ such that the resulting arithmetic coder is asymptotically optimal for the regular language $\mathcal{L}(\mathcal{D})$. Our approach for the computation of $p(\tau)$ is based on the analysis of the asymptotic growth of a regular language and uses ideas similar to [20, 3].

For our analysis we define the number g_m^A of word suffixes of length m, which can be generated from state A, i.e.

$$g_m^A \stackrel{\text{def}}{=} \left|A \stackrel{m}{\Rightarrow} \mathcal{T}\right|.$$

As all suffixes of length m that can be generated from state A have to be generated with one of the outgoing transitions from the target state with one symbol less,

a) 

b)

m	g_m^A	g_m^B	$A\xrightarrow{a}A$	$A\xrightarrow{b}A$	$B\xrightarrow{a}A$
0	0	1	-	-	-
1	1	0	0	1	1
2	1	1	1	0	1
3	2	1	.500	.500	1
4	3	2	.666	.333	1
5	5	3	.600	.400	1
29	514k	317k	.618	.382	1
30	832k	514k	.618	.382	1

Figure 1: a) DFA $\mathcal{D} = \Big(\{A, B\}, A, \{B\}, \{A\xrightarrow{a}A, A\xrightarrow{b}B, B\xrightarrow{a}A\}\Big)$ for $\mathcal{L} = \{b, ab, aab,\ bab, aaab,\ abab, baab, \dots\}$, b) suffix counts and transition probabilities for different word lengths m.

we get a recursion in m

$$\text{(3.2)}\quad \text{a) } g_m^A = \sum_{B\in\mathcal{S},\sigma\in\mathcal{A}} g_{m-1}^B \qquad \text{b) } \vec{g}_m = \mathbf{A}\vec{g}_{m-1},$$

where equation b) is in matrix notation. If we define g_0^A to be 1 for terminal states and 0 otherwise, we get $\vec{g}_m = \mathbf{A}^m\vec{g}_0$. The first two columns of Figure 1 b) give the first $\vec{g}_m$ for the example state machine in Figure 1, for which equation 3.2 specializes to $g_m^A = g_{m-1}^A + g_{m-1}^B$ and $g_m^B = g_{m-1}^A$.

The optimal transition probabilities for the creation of a suffix of length m from state A can be compute from g_m^A simply to

$$\text{(3.3)}\qquad p_m\left(A\dot{\rightarrow}B\right) \stackrel{\text{def}}{=} \frac{g_{m-1}^B}{g_m^A}.$$

and have to sum to one because of the recursion 3.2. The third and fourth columns of Figure 1 b) tabulate the transition probabilities for the example of Figure 1 and show that the transition probabilities converge for $m \to \infty$.

The transition probabilities p_m are not suitable for the design of a Markov Model as they depend on m. To investigate the limit for $m \to \infty$ we define the total number G_m of suffixes of length m, the relative number r_m^A of generatable suffixes from A and the *growth factor* α_m as

$$G_m \stackrel{\text{def}}{=} \sum_{A\in\mathcal{S}} g_m^A$$

$$r_m^A \stackrel{\text{def}}{=} g_m^A / G_m \in [0,1]$$
$$\alpha_m \stackrel{\text{def}}{=} G_m / G_{m-1}$$

equation 3.2 becomes

(3.4)a) $\alpha_m r_m^A = \sum_{B\in\mathcal{S},\sigma\in\mathcal{A}} r_{m-1}^B$, b) $\alpha_m \vec{r}_m = \mathbf{A}\vec{r}_{m-1}$.

In the limit for $m \to \infty$ the indices $\vec{r}_m$ and $\vec{r}_{m-1}$ are identified with the asymptotic relative numbers $\vec{r}$ of suffixes, which can be computed together with the asymptotic growth factor α as the eigenvector to the largest eigenvalue of the adjacency matrix $\mathbf{A}$ (compare theorem 2.1). The computation of the optimal asymptotic transition probabilities p follows equation 3.3, where the difference in the m-index can be balanced by dividing with α:

THEOREM 3.1. (MARKOV MODEL DESIGN) *Given a deterministic state machine $\mathcal{D}$ and its adjacency matrix $\mathbf{A}$ with unique largest eigenvalue α and eigenvector $\vec{r}$, then*

$$p(A \dot{\to} B) = \frac{r^B}{\alpha r^A}$$

are valid transition probabilities, that are asymptotically optimal for $\mathcal{L}(\mathcal{D})$.

Proof: From theorem 2.1 we know that all transition probabilities are positive. As $\vec{r}$ is eigenvector of $\mathbf{A}$ with eigenvalue α, we have $\alpha\vec{r} = \mathbf{A}\vec{r}$, from which directly follows that the outgoing probabilities of each state sum to one and are all valid transition probabilities.

The asymptotic lower bound β can be computed from the asymptotic growth factor to be $\log_2 \alpha$ bits per symbol as α is the largest eigenvalue of $\mathbf{A}$. There can be further complex and negative eigenvalues of the same absolute value, but the absolute value grows per symbol by a factor of α.

To show that the upper bound $\mathcal{B}$ of arithmetic coding matches β, we could compute its entropy, which would result in $\log_2 \alpha$. A simpler proof that generalizes to the optimization approach in the following section shows for all cycles of finite length m that the arithmetic coder does exactly consume $m \log_2 \alpha$ bits. This is sufficient as the state machine is finite and we are interested in its asymptotic behavior, which can only be based on cycles. The arithmetic coder consumes for an arbitrary cycle $(A_1 \dot{\to} A_2, A_2 \dot{\to} A_3, \ldots, A_m \dot{\to} A_1)$ minus $\log_2 (p(A_1 \dot{\to} A_2) \cdot p(A_2 \dot{\to} A_3) \cdot \ldots \cdot p(A_m \dot{\to} A_1))$ bits. If we plug in the probabilities as defined in the theorem, all factors of r^{A_i} cancel out as they appear once in the counter and once in the denominator. Only m αs remain in the denominator, yielding $m \log_2 \alpha$ and completing the proof. $\square$

The optimal transition probabilities are independent of the set of terminal states. To compute the transition probabilities for the state machine in Figure 1 we need $\mathbf{A} = \begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix}$, of which $\alpha = \frac{1+\sqrt{5}}{2}$ and $\vec{r} = \frac{1}{1+\alpha}\begin{pmatrix} \alpha \\ 1 \end{pmatrix}$. According to theorem 3.1 we compute $p(A \xrightarrow{a} A) = \frac{1}{\alpha} \approx .618$, $p(A \xrightarrow{b} B) = \frac{1}{\alpha^2} \approx .382$ and $p(B \xrightarrow{a} A) = 1$, what allows us to encode the words of $\mathcal{L}(\mathcal{D})$ with $\beta = \mathcal{B} = \log_2 \alpha \approx .694$ bps.

In general can α and $\vec{r}$ be computed efficiently with the always convergent power iteration $(\alpha_m + 1)\vec{r}_m = (\mathbf{A} + \mathbf{1})\vec{r}_{m-1}$, which avoids problems of negative and complex eigenvalues of the same absolute value.

4 Optimization of Markov Models under Constraints

The language of five-symbol strings used for triangle mesh connectivity coding is not regular. There are additional constraints that relate the counts of different symbols in each valid word. To generalize this notion we define $\#_\sigma(\omega)$ as the number of σ in the word ω and $\vec{\#}(\omega)$ as the vector of all symbol counts for ω.

DEFINITION 4.1. (LINEARLY CONSTRAINED LANGUAGE) *Let $\mathcal{A}$ be an alphabet with n symbols and $\mathbf{\Gamma} \in \mathbb{Z}^{k\times n}$ the matrix of $k \leq n-1$ linearly independent constraints, then*

$$\mathcal{C}_{\mathbf{\Gamma}} \stackrel{def}{=} \left\{ \omega \in \mathcal{A}^* | \mathbf{\Gamma}\vec{\#}(\omega) = 0 \right\}.$$

Only $n-1$ constraints are possible because of the implicit constraint $\sum_\sigma \#_\sigma(\omega) = |\omega|$. With $\mathcal{L}_{\mathbf{\Gamma}}(\mathcal{D})$ we abbreviate $\mathcal{L}(\mathcal{D}) \cap \mathcal{C}_{\mathbf{\Gamma}}$. It is clear that $\beta(\mathcal{L}_{\mathbf{\Gamma}}) \leq \beta(\mathcal{L})$, what allows to save more bits during coding.

As the constraints apply to every word of the language, they also hold for the probabilities $P(\sigma)$ of seeing a symbol σ. With the vector $\vec{P}_{\mathcal{A}}$ that combines all symbol probabilities $P(\sigma)$ we get

$$\mathbf{\Gamma}\vec{P}_{\mathcal{A}} = 0, \quad \text{and} \quad \sum_\sigma P(\sigma) = 1. \tag{4.5}$$

For the simple example of Figure 1 we can add only one constraint $P(b) = \lambda P(a)$, where λ is a rational that includes the coefficient of $P(b)$ in the denominator. From $P(a) + P(b) = 1$ follows $P(a) = 1/(1+\lambda)$ and $P(b) = \lambda/(1+\lambda)$.

The optimization problem for a Markov Model is to minimize the achieved upper bound $\mathcal{B}\left(p(A \xrightarrow{\sigma} B)\right)$ for all transition probabilities under the constraints on the symbol probabilities 4.5. The main problem here is to efficiently compute $\mathcal{B}$ for arbitrary $p(A \xrightarrow{\sigma} B)$. The proposed solution follows the idea of the proof of theorem 3.1, where we showed with the examination of an arbitrary cycle that each symbol costs the arithmetic coder $\log_2 \alpha$ bits. The transition probabilities of the theorem assigned the same cost for all different symbols in $\mathcal{A}$.

For the generalization we introduce a different α_σ for each symbol σ resulting in different costs $\log_2 \alpha_\sigma$. The α_σs are collected in the n-dimensional vector $\vec{\alpha}$. The adjacency matrix of the transition graph is modified to $\mathbf{M}(\vec{\alpha})$ with $\mathbf{M}_{AB} = \sum_{\sigma | A \xrightarrow{\sigma} B \in \tau} 1/\alpha_\sigma$. This fixes the eigenvalue to one

$$\vec{r} = \mathbf{M}(\vec{\alpha})\vec{r} \tag{4.6}$$

with the matrix $\mathbf{M}$ depending on $\vec{\alpha}$. In the special case, where all $\alpha \equiv \alpha_\sigma$ are the same, one can multiply with α, what brings us back to $\alpha\vec{r} = \mathbf{A}\vec{r}$. In the general case one component of $\vec{\alpha}$ is fixed by the condition that the largest eigenvalue of $\mathbf{M}$ must be one. A research issue for future work is to examine, whether there is always an α_σ with which $\mathbf{M}$ can be adjusted in this way.

For now we assume that such an α_σ exists, as was the case for the application to triangle connectivity coding. One can show with a proof very similar to the proof of theorem 3.1 that with the following transition probabilities each symbol σ asymptotically costs $\log_2 \alpha_\sigma$ and that the upper bound $\mathcal{B}$ of the coding scheme can be computed according to:

$$\begin{aligned} & p\left(A \xrightarrow{\sigma} B\right) \stackrel{\text{def}}{=} \frac{r^B}{\alpha_\sigma r^A} \\ \Rightarrow \quad & \mathcal{B}(\alpha_\sigma) = \max_{\vec{P}_\mathcal{A} | \mathbf{\Gamma} \vec{P}_\mathcal{A} = 0} \sum_{\sigma \in \mathcal{A}} P_\sigma \log_2 \alpha_\sigma. \end{aligned} \tag{4.7}$$

The optimization strategy for the Markov Model can now be stated as the minimization of the upper bound:

$$\min_{\vec{\alpha} | \exists \vec{r} : \vec{r} = \mathbf{M}(\vec{\alpha})\vec{r}} \mathcal{B}(\vec{\alpha}). \tag{4.8}$$

The vector $\vec{\alpha}$ that minimizes the upper bound together with the corresponding eigenvector $\vec{r}$ to eigenvalue one are used to compute the optimized transition probabilities according to the top of equation 4.7.

Let us exemplarily optimize the transition probabilities for $\mathcal{D}$ in Figure 1 under the constraint $P(a) = \lambda P(b)$. As λ fixes both symbol probabilities, no maximum is necessary to compute $\mathcal{B}$. We vary α_a and compute α_b to fulfill equation 4.6. With $q_a = \frac{1}{\alpha_a}$ and $q_b = \frac{1}{\alpha_b}$ we can construct $\mathbf{M} = \begin{pmatrix} q_a & q_a \\ q_b & 0 \end{pmatrix}$, with largest eigenvalue $\alpha = (q_a + \sqrt{q_a^2 + 4 q_a q_b})/2$. From $\alpha = 1$ we get $q_b = \alpha_a - 1$. Plugging all in yields $\mathcal{B}(\alpha_a) = [\log_2 \alpha_a - \lambda \log_2(\alpha_a - 1)]/(\lambda + 1)$, what we minimize by setting the derivative for α_a to zero and get $\alpha_a = 1/(1 - \lambda)$. The resulting upper bound

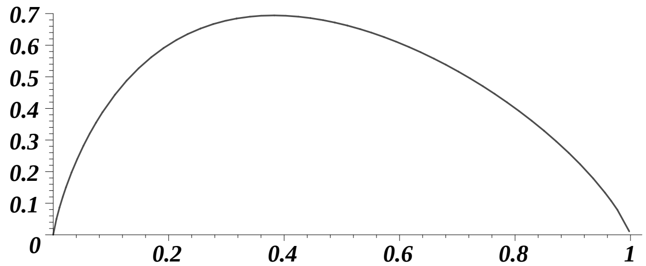


Figure 2: The achieved upper bound for the DFA of Figure 1 in bps with the constraint $P(a) = \lambda P(b)$, plotted over λ.

$\mathcal{B} = [\lambda \log_2 \lambda - (1 - \lambda) \log_2(1 - \lambda)]/(\lambda + 1)$ is plotted in Figure 2. It hits the unconstrained upper bound for $\lambda = (3 - \sqrt{5})/2 \approx .382$ and is less otherwise.

5 Modified Edgebreaker Coding

Our application is the encoding of triangle mesh connectivity. A triangle mesh consists of an indexed list of vertices with 3D locations and a list of index triples defining the so called *connectivity*, i.e. the information about the connectedness of the vertices. We restrict ourselves to manifold triangle meshes with boundary, where to each edge are one or two triangles incident and the triangles incident upon a vertex form an open or closed fan.

We used the well known Edgebreaker technique [22] to transform the connectivity information into a word over the alphabet $\mathcal{A} = \{C, L, R, S, E\}$. A growing region is defined by a stack of closed edge loops to separate the processed triangles from the unprocessed ones. One edge of each loop is labeled as *gate* edge. The region is initialized to a boundary loop or an arbitrary triangle with an arbitrarily chosen gate. Each letter corresponds to one of the five operations shown in Figure 3. It defines how the triangle incident to the gate of the loop on top of the stack is incorporated to the processed region. C introduces a new vertex, L and R shorten the current loop, S splits the current loop and pushes the left loop on the stack, E removes the loop on top of the stack and proceeds with the loop below or terminates if the stack is empty. Each operation defines the gate location after the operation – S defines two, one for each loop, and E none – in order to fix the traversal order.

To avoid the need of a position index for the third vertex in the S operation, decoding is performed in reverse order [14] with the inverse operations shown in the bottom row of Figure 3. The inverse split

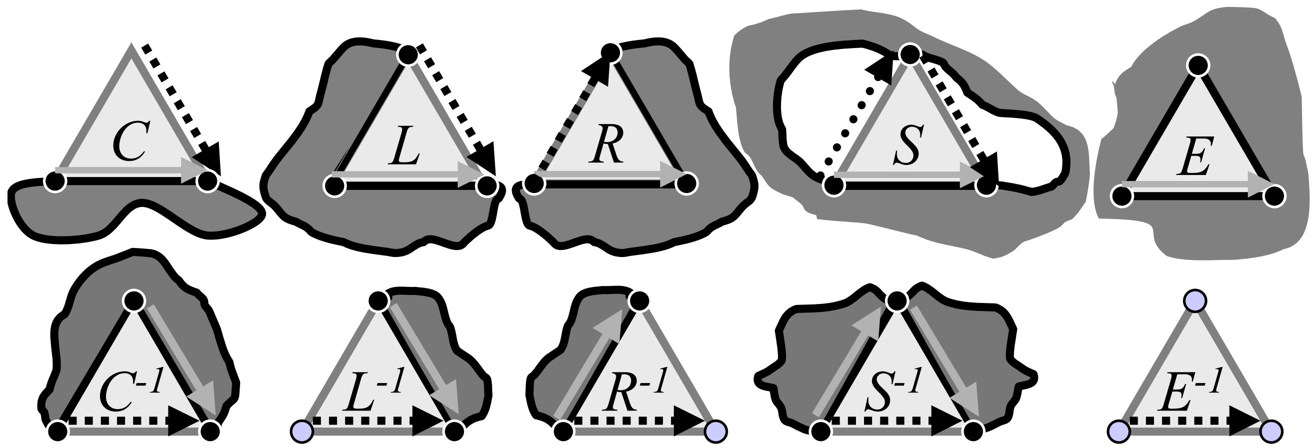


Figure 3: top: Edgebreaker operations, bottom: inverse operations for spiral reversi decoding. Processed region before operation shaded dark with black boundary, currently processed triangle shaded bright, operation name inside. Gate edge(s) before grey solid arrows, gate(s) after operation dashed black arrows. Old vertices and edges black and new ones grey.

combines the two top most loops on the stack at their gates without the need of any additional information. For the treatment of holes and handles we basically follow the approach of spiral reversi [14] but do not use additional symbols M and H, but instead encode a position index in the CLRSE-word. For meshes with g handles and b boundary loops, additional $2b + 4g + 2$ integers are necessary, which we assume to be asymptotically negligible.

6 Optimized Triangular Connectivity Coding

The used arithmetic coder also works in reverse order. By examining the inverse operations we find similarly to [9] the following constraints:

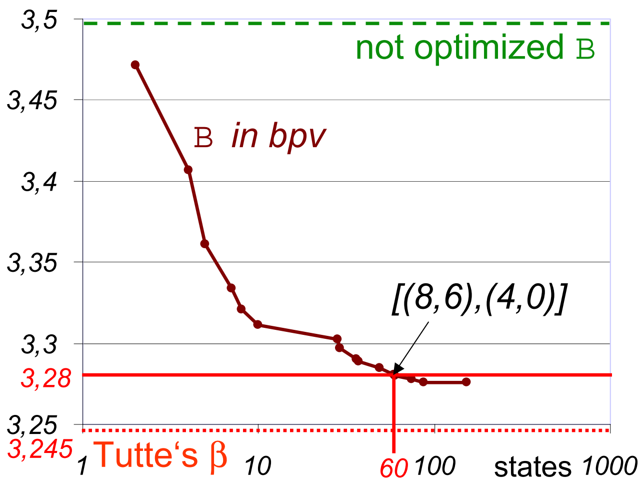
1. *manifold constraint:* C^{-1} must not re-create an already existing edge as this would make this edge non-manifold in the next step.

2. *loop constraint:* E and S operations must be balanced $\Rightarrow P(S) = P(E)$.

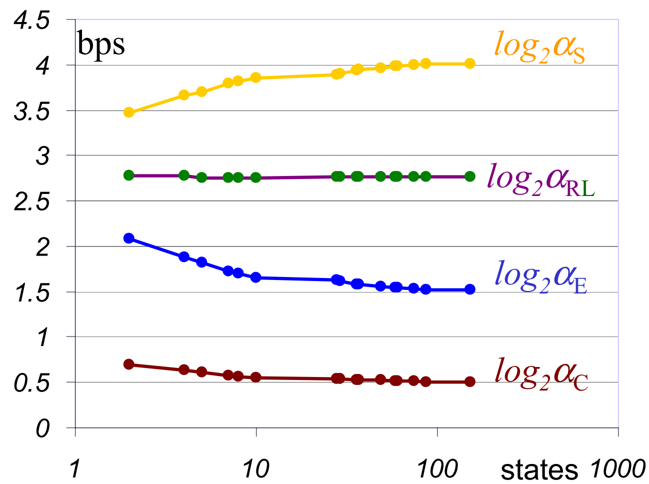
3. *Euler constraint:* for triangle meshes of low genus and small border fraction the Euler equation says that the number of triangles is about twice the number of vertices. As each Edgebreaker operation introduces one triangle and only each C operation introduces in addition to a triangle also one vertex, this leads to the second constraint: $P(C) = P(L) + P(R) + P(S) + P(E)$.

Only the manifold constraint cannot be mapped to a linear constraint on the symbol probabilities. Instead we approximated the constraint with a regular language. The edge introduced by an inverse C can be present because the length of the current loop is three or because the same edge has been introduced before, i.e. an interior edge connecting the target vertex of the gate with another vertex of the current loop. The inverse L introduces an interior edge that disallows a succeeding inverse C. The inverse R operation on the other hand resolves all possible conflicts with interior edges. The right interior edge created by the inverse S is also a candidate, which disallows a sequence of k successive inverse C operations, where k is the number of edges in the left of the two loops merged by the inverse S. In terms of the manifold constraint one can identify the inverse L with $k = 1$. The inverse E finally does not create an interior edge but a loop of length three.

The manifold constraint is implemented in a state machine, where the states describe the top most loops on the stack. Each loop is represented by its length l and the position k of a potentially conflicting interior edge. If two top most loops are considered, a state is represented by two pairs of integers. $[(5, 1), (4, .)]$ would for example correspond to a top most loop of length 5 with a conflict edge at $k = 1$ and a loop of length 4 without conflicting edge below. The initial state is always $S_0 = [(3, .)]$. From this we determined all states that are reachable. The user specifies maximal values for the different loop lengths and k-values. Transitions to states with lengths or ks larger than the limits were redirected to S_0. As in the end of decoding all loops must have been encoded, S_0 also served as the only



a)



b)

Figure 4: a) the achieved upper bound in bits per vertex (bpv) plotted over the number of states used to approximate the manifold constraint, b) plot of the symbol cost $\log_2 \alpha_\sigma$ in bps over the number of states

terminal state. The manifold constraint is implemented by removing all C transitions, when the length of the top loop is 3 or $k = 1$.

With the number of considered top loops and the maximal values for l and k the number of states in the DFA could be varied. For the optimization of the Markov Models with respect to the two linear constraints stated above, the technique proposed in section 4 was combined with a gradient descent optimization technique. Figure 4 a) shows a diagram of the achieved upper bound in dependence of the number of states used to approximate the manifold constraint. The diagram includes the information theoretic lower bound as well as the result achieved in previous work without the new optimization technique.

Theorem 6.1. (Connectivity Coding) *The connectivity of a triangle mesh with v vertices, genus g and b boundary loops, which contain a sub-linear fraction of the mesh vertices, can be encoded and decoded in linear time in asymptotically less than $3.28v + (2 + 4g + 2b)\log_2 v$ bits with a Markov Model with 60 states.*

7 Conclusion and Future Work

In this paper we showed how to design asymptotic optimal arithmetic coders based on Markov Models for regular languages given in form of DFAs. We introduced the notion of constrained regular languages and proposed a feasible scheme to optimize Markov Models for these languages. We applied the optimization scheme to the encoding of triangular meshes of low genus and border fraction. The resulting coding scheme is up to one percent optimal with relation to the information theoretic lower bound. This shows that the proposed optimization scheme performs very well in the presented example.

In future work we want to examine further applications of the proposed technique in the area of mesh compression, such as polygonal mesh and progressive mesh coding. We also want to develop the theory of constrained regular languages and optimized Markov Models further and examine the important open questions: which non regular languages can be asymptotically approximated well with constrained regular languages and whether optimized Markov Models always allow to encode constrained regular languages asymptotically optimal.

References

[1] P. Alliez and M. Desbrun. Valence-driven connectivity encoding for 3D meshes. *Computer Graphics Forum*, 20(3), 2001. ISSN 1067-7055.

[2] C. Bajaj, V. Pascucci, and G. Zhuang. Single resolution compression of arbitrary triangular meshes with properties. In *Data Compression Conference'99 Proceedings*, pages 247–256, 1999.

[3] T. Ceccherini-Silberstein and W. Woess. Growth-sensitivity of context-free languages. *Theoretical Computer Science*, 307(1):103–116, September 2003.

[4] R. C.-N. Chuang, A. Garg, X. He, M.-Y. Kao, and H.-I. Lu. Compact encodings of planar graphs via canonical orderings and multiple parentheses. *Lecture Notes in Computer Science*, 1443:118–127, 1998.

[5] M. Deering. Geometry compression. In *SIGGRAPH'95 Conference Proceedings*, pages 13–20, 1995.

[6] M. Denny and C. Sohler. Encoding a triangulation as a permutation of its point set. In *Proceedings of 9th Canadian Conference on Computational Geometry*, pages 39–43, 1997.

[7] C. Gotsman. On the optimality of valence-based connectivity coding. *Computer Graphics Forum*, 22(1):99–102, March 2003.

[8] S. Gumhold. Improved cut-border machine for triangle mesh compression. In *Erlangen Workshop 99 on Vision, Modeling and Visualization*, pages 261–268, Erlangen, Germany, November 1999.

[9] S. Gumhold. Connectivity coding: New perspectives for mesh compression. *Informationstechnik und Technische Informatik*, December 2002. available at `http://www.mpi-sb.mpg.de/~sgumhold`.

[10] S. Gumhold and W. Strasser. Real time compression of triangle mesh connectivity. In *SIGGRAPH'98 Conference Proceedings*, pages 133–140, 1998.

[11] Stefan Gumhold. New bounds on the encoding of planar triangulations. Technical Report WSI–2000–1, Wilhelm-Schickard-Institut für Informatik, University of Tübingen, Germany, January 2000.

[12] J. Ho, K.-C. Lee, and D. Kriegman. Compressing large polygonal models. In *Proc. of IEEE Visualization 2001*, pages 357–362, 2001.

[13] M. Isenburg and S. Gumhold. Out-of-core compression for gigantic polygon meshes. *ACM Transactions on Graphics*, 22(3):935–942, July 2003.

[14] M. Isenburg and J. Snoeyink. Spirale reversi: Reverse decoding of the Edgebreaker encoding. In *Proceedings of 12th Canadian Conference on Computational Geometry*, pages 247–256, 2000.

[15] S. Kannan, Z. Sweedyk, and S. Mahaney. Counting and random generation of strings in regular languages. In *Symposium on Discrete Algorithms*, pages 551–557, 1995.

[16] K. Keeler and J. Westbrook. Short encodings of planar graphs and maps. In *Discrete Applied Mathematics*, pages 239–252, 1995.

[17] D. King and J. Rossignac. Guaranteed 3.67v bit encoding of planar triangle graphs. In *Proceedings of 11th Canadian Conference on Computational Geometry*, pages 146–149, 1999.

[18] H. Lee, P. Alliez, and M. Desbrun. Angle-analyzer: A triangle-quad mesh codec. *Computer Graphics Forum*, 21(3):383–383, 2002.

[19] J. Li and C. C. Kuo. A dual graph approach to 3D triangular mesh compression. In *Proceedings of ICIP'98*, pages 891–894, 1998.

[20] B.H. Marcus, R.M. Roth, and P.H. Siegel. Constrained systems and coding for recording channels. In W.C. Huffman V.S. Pless, editor, *Handbook of Coding Theory*, pages 1635–1764. Elsevier, Amsterdam, 1998.

[21] D. Poulalhon and G. Schaeffer. A bijection for triangulations of a polygon with interior points and multiple edges. *Theoretical Computer Science*, 307(2):385–401, October 2003.

[22] J. Rossignac. Edgebreaker: Connectivity compression for triangle meshes. *IEEE Transactions on Visualization and Computer Graphics*, 5(1):47–61, 1999.

[23] A. Szymczak, D. King, and J. Rossignac. An Edgebreaker-based efficient compression scheme for connectivity of regular meshes. In *Proceedings of 12th Canadian Conference on Computational Geometry*, pages 257–264, 2000.

[24] G. Taubin and J. Rossignac. Geometric compression through topological surgery. *ACM Transactions on Graphics*, 17(2):84–115, 1998.

[25] C. Touma and C. Gotsman. Triangle mesh compression. In *Graphics Interface'98 Conference Proceedings*, pages 26–34, 1998.

[26] G. Turan. Succinct representations of graphs. *Discrete Applied Mathematics*, 8:289–294, 1984.

[27] W. T. Tutte. A census of planar triangulations. *Canadian Journal of Mathematics*, 14:21–38, 1962.

[28] I. H. Witten, R. M. Neal, and J. G. Cleary. Arithmetic coding for data compression. *Communications of the ACM*, 30(6):520–540, 1987.