

CERTIFIED UPPER BOUNDS FOR THE CHVÁTAL–SANKOFF CONSTANT

TEJASVI SINGH TOMAR

ABSTRACT. Let γ_2 be the Chvátal–Sankoff constant, so that the expected length of a longest common subsequence of two independent uniformly random binary strings of length n is $\gamma_2 n + o(n)$. Lueker proved $\gamma_2 \leq 0.826280$ with a finite automaton that reads a pair of strings as a sequence of minimal match pairs and remembers the last h symbols of each string, for $h = 8$. We carry out his method up to $h = 12$ and prove $\gamma_2 \leq 0.822579$. For $h = 12$ the certificate lists 22,670,087,101 states of the automaton, reduced by a symmetry. The bound rests on an explicit positive vector for the weighted transition matrix of the automaton; a checker rebuilds every transition with its own code and verifies the required inequalities in exact integer arithmetic. We also give a complete proof that the automaton accepts the truncated canonical decomposition of every pair of strings, the property on which the method depends.

1. INTRODUCTION

For strings A and B let $\text{LCS}(A, B)$ be the length of a longest common subsequence of A and B . Let A_n and B_n be independent uniformly random strings of length n over $\Sigma = \{0, 1\}$ and put $L_n = \text{LCS}(A_n, B_n)$. Chvátal and Sankoff [2] showed that $\mathbb{E}L_n/n$ converges as $n \rightarrow \infty$; the limit γ_2 is called the Chvátal–Sankoff constant. Since $\mathbb{E}L_{m+n} \geq \mathbb{E}L_m + \mathbb{E}L_n$, the limit is also the supremum of $\mathbb{E}L_n/n$.

The value of γ_2 is not known. Simulations suggest $\gamma_2 \approx 0.8122$ [1, Section 1]. Dančák [3] proved $\gamma_2 \geq 0.773911$, and Dančák and Paterson [4] (see also [3]) proved $\gamma_2 \leq 0.837623$ with a 52-state automaton [7, Section 3.1]. Lueker [7] improved both bounds, to $0.788071 \leq \gamma_2 \leq 0.826280$. His upper-bound technique [7, Section 3] uses finite automata M_h with a history parameter h , which read a pair of strings as a sequence of minimal match pairs [7, Section 3.2]. The bound 0.826280 comes from M_8 , which has 6,419,109 reachable states after a symmetry reduction [7, Table II], [8], and it shows that the value $2/(1 + \sqrt{2}) \approx 0.828427$ conjectured by Steele is too large [7, p. 17:21]. Lueker also proved that the bounds obtainable from M_h converge to γ_k as $h \rightarrow \infty$, for every alphabet size k [7, Section 3.3]. The lower bound has since been improved to 0.792665992 [9]. In October 2026 Jack [6] posted computations with Lueker’s automaton for $h = 9$ and $h = 10$, which give $\gamma_2 \leq 0.825101$ and $\gamma_2 \leq 0.824120$. They have not been refereed; the repository of optimization constants maintained by Davis, Ivanišvili, Tao and contributors [5, page 31a] lists them with an asterisk, its mark for bounds whose available verification is at a minimal level.

Theorem 1.1. $\gamma_2 \leq 0.822579$.

The proof follows Lueker’s method with history length $h = 12$. Table 1 lists the bounds that the same method and the same checking program give for $h = 8, \dots, 12$; for $h = 8$ we recover Lueker’s value 0.826280, and for $h = 9$ and $h = 10$ the values of [6].

2020 *Mathematics Subject Classification.* 60C05; 68R15, 68Q45, 65G20.

Key words and phrases. Chvátal–Sankoff constant, longest common subsequence, random strings, Lueker’s method, Collatz–Wielandt bound, computer-assisted proof.

Our verification differs from the earlier computations in two respects. First, Lueker justifies his final bound under the assumption that the floating-point operations of his program perform as expected [7, Section 3.2.2], and the computations of [6] use floating-point arithmetic with upward rounding, together with a recheck for $h = 10$ that does not depend on the rounding mode. Our checker verifies the required inequalities exactly, in integer arithmetic, for a rational parameter z and with all weights rounded up. Second, Lueker explains why his automaton never rejects the canonical decomposition of a pair of strings [7, Section 3.2]; we give a detailed proof of this covering property (Lemma 3.1).

The method. A pair of strings decomposes canonically into minimal match pairs, one for each symbol of a particular longest common subsequence, followed by a pair without matches (Lemma 2.3). The automaton M_h reads the minimal match pairs one at a time. Its state consists of the last h symbols of each string and of the last h increments of the dynamic programming table down its last column and along its last row. It rejects an input when the canonical traceback rule shows that the input cannot continue a canonical decomposition. Give a minimal match pair of total length s the weight z^s , and let A_z be the weighted transition matrix of M_h . If $0 < z < 1/2$, $v > 0$ and $A_z v \leq \lambda v$ for some $\lambda < 1$, then $\gamma_2 \leq 2 \ln(2z) / \ln \lambda$ (Theorem 4.1). The automata M_h and this use of them are Lueker's [7]; the counting criterion is due to Dančák and Paterson [3, 4], in the form of [7, Theorem 3.3]. For $h = 12$ and $z = 0.3505$ the certificate lists the 22,670,087,101 states of the reduced automaton that a breadth-first search from the start state finds, with 418,139,991,128 accepted transitions, and we take v from a power iteration.

Section 2 collects facts about longest common subsequences and the canonical decomposition. Section 3 defines M_h and proves the covering property, and Section 4 proves the counting bound. Section 5 describes the certificates, the checker and the results, and Section 6 contains remarks.

2. LONGEST COMMON SUBSEQUENCES

Throughout, $\Sigma = \{0, 1\}$ and $\bar{c} = 1 - c$ for $c \in \Sigma$. For a string A we write $A[i]$ for its i th symbol, $A[i..j]$ for $A[i]A[i+1] \cdots A[j]$ and $|A|$ for its length. For strings A, B let $L(i, j) = \text{LCS}(A[1..i], B[1..j])$, with $L(i, 0) = L(0, j) = 0$. For $i, j \geq 1$,

$$(2.1) \quad L(i, j) = \begin{cases} L(i-1, j-1) + 1, & A[i] = B[j], \\ \max\{L(i-1, j), L(i, j-1)\}, & A[i] \neq B[j]. \end{cases}$$

Lemma 2.1. *Let $a_1, \dots, a_m, b_1, \dots, b_n \in \Sigma$ and let $W: \{0, \dots, m\} \times \{0, \dots, n\} \rightarrow \mathbb{Z}$ satisfy (2.1) with a_r, b_s in place of $A[i], B[j]$ for all $r, s \geq 1$. If $W(r, 0) - W(r-1, 0) \in \{0, 1\}$ and $W(0, s) - W(0, s-1) \in \{0, 1\}$ for all $r, s \geq 1$, then $W(r, s) - W(r-1, s) \in \{0, 1\}$ and $W(r, s) - W(r, s-1) \in \{0, 1\}$ for all $r, s \geq 1$.*

Proof. By induction on $r + s$. Put $u = W(r-1, s-1)$, $u' = W(r-1, s)$ and $u'' = W(r, s-1)$; by induction $u', u'' \in \{u, u+1\}$. If $a_r = b_s$, then $W(r, s) = u + 1$ lies in $\{u', u' + 1\}$ and in $\{u'', u'' + 1\}$. Otherwise $W(r, s) = \max\{u', u''\}$, which lies in $\{u', u' + 1\}$ and in $\{u'', u'' + 1\}$ because $|u' - u''| \leq 1$. $\square$

In particular all increments of L lie in $\{0, 1\}$, so L is nondecreasing in each argument.

2.1. Minimal match pairs. A pair of nonempty strings (u, w) is a *minimal match pair* if $u[|u|] = w[|w|]$ and $u[i] \neq w[j]$ for every other pair of positions (i, j) . Its *size* is $|u| + |w|$. For $c \in \Sigma$ and $l \in \mathbb{Z}$ put

$$T_{c,l} = (\bar{c}^l c, c) \quad (l \geq 0), \quad T_{c,l} = (c, \bar{c}^{|l|} c) \quad (l < 0),$$

a pair of size $|l| + 2$.

Lemma 2.2. *The minimal match pairs over Σ are exactly the pairs $T_{c,l}$, $c \in \Sigma$, $l \in \mathbb{Z}$.*

Proof. Each $T_{c,l}$ is a minimal match pair. Conversely, let (u, w) be a minimal match pair with last symbol c . If $|u|, |w| \geq 2$, then $u[1] \neq w[|w|] = c$ and $u[|u|] = c \neq w[1]$ give $u[1] = w[1] = \bar{c}$, which contradicts $u[1] \neq w[1]$. So $|u| = 1$ or $|w| = 1$, and every other symbol of the longer string differs from c . $\square$

2.2. The canonical decomposition. The *canonical traceback* starts at a cell (i, j) and, while $i, j \geq 1$, moves up to $(i-1, j)$ if $L(i-1, j) = L(i, j)$, else left to $(i, j-1)$ if $L(i, j-1) = L(i, j)$, and else diagonally to $(i-1, j-1)$. By Lemma 2.1 and (2.1), a diagonal step from (i, j) happens only if $A[i] = B[j]$ and $L(i-1, j) = L(i, j-1) = L(i-1, j-1) = L(i, j) - 1$.

Lemma 2.3 (cf. [7, Lemmas 3.4 and 3.5]). *Let $A, B \in \Sigma^n$ and let $(i_1, j_1), \dots, (i_m, j_m)$, with $i_1 < \dots < i_m$, be the cells at which the canonical traceback from (n, n) takes a diagonal step; put $(i_0, j_0) = (0, 0)$. Then $m = L(n, n)$ and $L(i_k, j_k) = k$ for each k . Each pair $p_k = (A[i_{k-1} + 1..i_k], B[j_{k-1} + 1..j_k])$ is a minimal match pair, and $A[i] \neq B[j]$ for all $i > i_m, j > j_m$. Moreover $L(i_k, j_k - 1) = L(i_k - 1, j_k) = k - 1$, and if $p_k = T_{c,l}$ with $l < 0$, then*

$$(2.2) \quad L(i_{k-1} - 1, j) < L(i_{k-1}, j) \quad \text{for } j_{k-1} < j < j_k \text{ and } k \geq 2.$$

We call $p_1, \dots, p_m$ the *canonical decomposition* of (A, B) .

Proof. The value of L is constant along up and left steps and drops by 1 at diagonal steps, and the traceback ends where $L = 0$. Hence $L(i_k, j_k) = k$ and $m = L(n, n)$. The last symbols of p_k agree, since a diagonal step is taken at (i_k, j_k) . Suppose $A[i] = B[j]$ for a cell $(i, j) \neq (i_k, j_k)$ of $(i_{k-1}, i_k] \times (j_{k-1}, j_k]$. Then $L(i, j) \geq L(i-1, j-1) + 1 \geq L(i_{k-1}, j_{k-1}) + 1 = k$. If $i < i_k$, then $L(i_k - 1, j_k) \geq L(i, j) \geq k$, so the traceback moves up at (i_k, j_k) ; if $i = i_k$ and $j < j_k$, then $L(i_k, j_k - 1) \geq k$, so it moves up or left. Both contradict the diagonal step at (i_k, j_k) . A pair $A[i] = B[j]$ with $i > i_m, j > j_m$ would give $L(n, n) \geq L(i_m, j_m) + 1 = m + 1$. The equalities $L(i_k, j_k - 1) = L(i_k - 1, j_k) = k - 1$ hold at every diagonal step.

For $k \geq 2$, between the diagonal steps at (i_k, j_k) and (i_{k-1}, j_{k-1}) the traceback moves from $(i_k - 1, j_k - 1)$ to (i_{k-1}, j_{k-1}) by up and left steps only. If $p_k = T_{c,l}$ with $l < 0$, then $i_k = i_{k-1} + 1$, so these are left steps along row i_{k-1} , on which $L = k - 1$. A left step at (i_{k-1}, j) with $j \geq 1$ is taken only if the up step is refused, that is, $L(i_{k-1} - 1, j) < L(i_{k-1}, j)$, which is (2.2). $\square$

Lemma 2.4. *For strings U, V and $c \in \Sigma$, $\text{LCS}(cU, cV) = 1 + \text{LCS}(U, V)$. Consequently, for $A, B \in \Sigma^n$ and $A^* = 0^h A$, $B^* = 0^h B$ with table L^* , we have $L^*(h + i, h + j) = h + L(i, j)$ for $0 \leq i, j \leq n$ and $L^*(r, s) = \min\{r, s\}$ whenever $\min\{r, s\} \leq h$.*

Proof. Let S be a longest common subsequence of cU and cV . If S begins with c , the rest of S is a common subsequence of U and V ; otherwise S itself is one. Hence $\text{LCS}(cU, cV) \leq 1 + \text{LCS}(U, V)$, and the reverse inequality is clear. The other statements follow by induction. $\square$

By Lemma 2.4, the traceback comparisons of L^* at cells $(h + i, h + j)$ with $i, j \geq 1$ are those of L at (i, j) . So the canonical decomposition of (A, B) is read off from L^* in the same way, with all coordinates shifted by h , and in (2.2) the case $k = 1$ also holds in L^* : there $L^*(h - 1, h + j) = h - 1 < h = L^*(h, h + j)$.

3. LUEKER'S AUTOMATON

We write out Lueker's automaton [7, Section 3.2 and Figure 7] in the form in which we use it. It differs from Figure 7 of [7] in two inessential ways: the inputs are truncated before they are read, and we normalize the table by $W(h, h) = 0$ instead of Lueker's value h , which does not affect the transitions [7, Lemma 3.21]. The start state below corresponds to prepending 0^h

to both strings, as in [7, Lemma 3.8], and the symmetry reduction is the one used in Lueker's programs [8]. Fix $h \geq 2$. A *state* is a quadruple $q = (x, y, d_x, d_y)$ of strings in Σ^h . To q we attach a partial table W : put $a_r = x[r]$ and $b_s = y[s]$ for $1 \leq r, s \leq h$, and

$$W(h, h) = 0, \quad W(r, h) - W(r - 1, h) = d_x[r], \quad W(h, s) - W(h, s - 1) = d_y[s] \quad (1 \leq r, s \leq h).$$

The *inputs* are the pairs $T_{c,l}$ with $c \in \Sigma$ and $|l| \leq h - 1$. The transition $\delta(q, T_{c,l})$ is computed as follows.

- (1) Append the first string of $T_{c,l}$ to a and the second to b ; let α and β be the new lengths, so that $(\alpha, \beta) = (h + l + 1, h + 1)$ if $l \geq 0$ and $(h + 1, h + |l| + 1)$ if $l < 0$.
- (2) Put $W(0, s) = W(0, h)$ for $h < s \leq \beta$ and $W(r, 0) = W(h, 0)$ for $h < r \leq \alpha$.
- (3) Fill every cell (r, s) with $1 \leq r \leq \alpha$, $1 \leq s \leq \beta$ and $\max\{r, s\} > h$, in row-major order, by (2.1) with a_r, b_s in place of $A[i], B[j]$.
- (4) Starting at (α, β) , apply the canonical traceback rule to W until the current cell is (h, h) or has a coordinate smaller than h . The input is *accepted* if the traceback stops at (h, h) , and *rejected* otherwise.
- (5) If the input is accepted, the new state is (x', y', d'_x, d'_y) with $x' = a[\alpha - h + 1.. \alpha]$, $y' = b[\beta - h + 1.. \beta]$, $d'_x[t] = W(\alpha - h + t, \beta) - W(\alpha - h + t - 1, \beta)$ and $d'_y[t] = W(\alpha, \beta - h + t) - W(\alpha, \beta - h + t - 1)$ for $1 \leq t \leq h$.

The *start state* is $q_0 = (0^h, 0^h, 1^h, 1^h)$. Every cell that steps (3)–(5) read is defined: in step (3) because the neighbours of a cell being filled are filled cells, cells of row h or column h , or cells set in step (2), and in steps (4) and (5) because they read only cells with $\max\{r, s\} > h$ and cells of row h and column h . The rectangle $\{h, \dots, \alpha\} \times \{0, \dots, h\}$ has a first row with increments $d_y[s]$ and a constant first column, so by Lemma 2.1 its increments lie in $\{0, 1\}$; then the rectangle $\{0, \dots, \alpha\} \times \{h, \dots, \beta\}$ has a constant first row and a first column with increments in $\{0, 1\}$, and Lemma 2.1 applies again. So the increments of W on the filled region lie in $\{0, 1\}$, and the new d'_x, d'_y are bit strings. On an accepted input the traceback takes exactly one diagonal step, at (α, β) . It takes at least one, because $a_\alpha = b_\beta$ and monotonicity give $W(\alpha, \beta) = W(\alpha - 1, \beta - 1) + 1 > W(h, h)$, while up and left steps keep the value. It takes at most one, because a diagonal step from a cell of row h or column h leaves the region of cells with both coordinates at least h , and a diagonal step at a filled cell (r, s) requires $a_r = b_s$ by (2.1), which for $r, s > h$ happens only at (α, β) , since $T_{c,l}$ is a minimal match pair.

Complementing every symbol of a , b and of the input does not change any comparison, so

$$(3.1) \quad \delta(\bar{q}, T_{\bar{c},l}) = \overline{\delta(q, T_{c,l})}, \quad \text{where } \overline{(x, y, d_x, d_y)} = (\bar{x}, \bar{y}, d_x, d_y),$$

and one side is a rejection if and only if the other is. Let $\text{rep}(q) = \bar{q}$ if $x[1] = 1$ and $\text{rep}(q) = q$ otherwise. The *reduced automaton* has the states q with $x[1] = 0$ and the transitions $\delta_{\text{red}} = \text{rep} \circ \delta$; it is the automaton of Lueker's programs.

Truncation sends an arbitrary minimal match pair to an input: $\mu(T_{c,l}) = T_{c,l'}$ with $l' = \text{sign}(l) \min\{|l|, h - 1\}$. For $0 < z < 1$ the weight of an input is the total weight z^{size} of the minimal match pairs that truncate to it:

$$(3.2) \quad w_z(T_{c,l}) = z^{|l|+2} \quad (|l| < h - 1), \quad w_z(T_{c,l}) = \sum_{j \geq h-1} z^{j+2} = \frac{z^{h+1}}{1-z} \quad (|l| = h - 1).$$

Lemma 3.1 (covering). *Let $A, B \in \Sigma^n$ have the canonical decomposition $p_1, \dots, p_m$. Then the inputs $\mu(p_1), \dots, \mu(p_m)$, fed to M_h from q_0 , are all accepted.*

Proof. We work with $A^* = 0^h A$, $B^* = 0^h B$ and L^* as in Lemma 2.4. Let $q_k = (x_k, y_k, d_{x,k}, d_{y,k})$ be the state after $\mu(p_1), \dots, \mu(p_k)$, let W_k be its table, and let $I = h + i_k$, $J = h + j_k$; then

$L^*(I, J) = h + k$. We prove by induction on k that the inputs are accepted and that the following invariant holds:

(I_k) : there are strictly increasing maps $\phi, \psi: \{0, \dots, h\} \rightarrow \{0, 1, \dots\}$ with $\phi(h) = I$, $\psi(h) = J$, $x_k[r] = A^*[\phi(r)]$ and $y_k[s] = B^*[\psi(s)]$ for $1 \leq r, s \leq h$, and an integer C with

$$W_k(r, h) + C \leq L^*(\phi(r), J), \quad W_k(h, s) + C \leq L^*(I, \psi(s)) \quad (0 \leq r, s \leq h),$$

with equality at $r = s = h$.

For $k = 0$ the table of q_0 is $W(r, h) = r - h$, $W(h, s) = s - h$, and $L^*(r, h) = r$, $L^*(h, s) = s$ by Lemma 2.4; so (I_0) holds with $\phi = \psi = \text{id}$ and $C = h$.

Assume (I_k) and let $p_{k+1} = T_{c,l^\circ}$, read as $\mu(p_{k+1}) = T_{c,l}$. Let $I' = I + |u|$ and $J' = J + |w|$, where $(u, w) = p_{k+1}$, so that (I', J') is the next diagonal cell of the traceback of L^* . Extend ϕ to $\{0, \dots, \alpha\}$ by sending $h + 1, \dots, \alpha$ to the last $\alpha - h$ rows up to I' , that is, $\phi(r) = I' - \alpha + r$ for $h < r \leq \alpha$, and extend ψ in the same way. The extensions are strictly increasing, since $\phi(h) = I \leq I' - (\alpha - h)$, and they preserve symbols: if $|l^\circ| > h - 1$, the truncation removes symbols $\bar{c}$ on the left of the longer string, and the last $\alpha - h$ (respectively $\beta - h$) symbols are kept.

We claim that $W(r, s) + C \leq L^*(\phi(r), \psi(s))$ for every cell of the filled region, where W is the table of step (3). For the constant extensions of step (2) this follows from (I_k) and the monotonicity of L^* , and for the cells of row h and column h it is (I_k) . For a filled cell, induct in the order of filling. If $a_r = b_s$, then $\phi(r - 1) \leq \phi(r) - 1$ and $\psi(s - 1) \leq \psi(s) - 1$ give

$$\begin{aligned} L^*(\phi(r), \psi(s)) &\geq L^*(\phi(r) - 1, \psi(s) - 1) + 1 \geq L^*(\phi(r - 1), \psi(s - 1)) + 1 \\ &\geq W(r - 1, s - 1) + C + 1 = W(r, s) + C. \end{aligned}$$

Otherwise $L^*(\phi(r), \psi(s)) \geq \max\{L^*(\phi(r - 1), \psi(s)), L^*(\phi(r), \psi(s - 1))\} \geq W(r, s) + C$.

Acceptance when $l \geq 0$. Then $\beta = h + 1$, $J' = J + 1$, and Lemma 2.3 gives $L^*(I', J) = L^*(I' - 1, J') = h + k$. Hence $W(\alpha, h) + C \leq h + k = W(h, h) + C$, and since W is nondecreasing, $W(r, h) = W(h, h)$ for $h \leq r \leq \alpha$. Likewise $W(\alpha - 1, h + 1) \leq W(h, h)$, because $\phi(\alpha - 1) = I' - 1$ (also when $\alpha = h + 1$, since then $l^\circ = 0$), and $W(\alpha - 1, h + 1) \geq W(\alpha - 1, h) = W(h, h)$. Since $a_\alpha = b_\beta$, $W(\alpha, \beta) = W(\alpha - 1, h) + 1 = W(h, h) + 1$, which exceeds $W(\alpha - 1, \beta) = W(\alpha - 1, h + 1)$ and $W(\alpha, h)$. So the traceback steps diagonally at (α, β) and then moves up along column h , where W is constant, to (h, h) : the input is accepted.

Acceptance when $l < 0$. Then $\alpha = h + 1$, $I' = I + 1$, and Lemma 2.3 gives $L^*(I, J') = L^*(I', J' - 1) = h + k$. As before $W(h, s) = W(h, h)$ for $h \leq s \leq \beta$ and $W(h + 1, \beta - 1) = W(h, h)$, so the traceback steps diagonally at (α, β) to $(h, \beta - 1)$. At a cell (h, s) with $h < s < \beta$ the value $\psi(s)$ lies strictly between J and J' . Since $h + k = L^*(I, J) \leq L^*(I, \psi(s)) \leq L^*(I', J' - 1) = h + k$, we have $L^*(I, \psi(s)) = h + k$, and (2.2) (in L^* , which includes the first block) gives $L^*(I - 1, \psi(s)) < h + k$. Hence

$$W(h - 1, s) + C \leq L^*(\phi(h - 1), \psi(s)) \leq L^*(I - 1, \psi(s)) \leq h + k - 1 < W(h, s) + C.$$

So the traceback does not move up; it moves left, since $W(h, s - 1) = W(h, s)$, and reaches (h, h) : the input is accepted.

The invariant. Now $W(\alpha, \beta) + C = W(h, h) + 1 + C = h + k + 1 = L^*(I', J')$, and the cells that define q_{k+1} are filled cells, for which the claim above gives the required inequalities. The table of q_{k+1} is $W_{k+1}(r, h) = W(\alpha - h + r, \beta) - W(\alpha, \beta)$ and $W_{k+1}(h, s) = W(\alpha, \beta - h + s) - W(\alpha, \beta)$, and $W(\alpha, \beta) = W(h, h) + 1 = 1$. So (I_{k+1}) holds with the maps $r \mapsto \phi(\alpha - h + r)$, $s \mapsto \psi(\beta - h + s)$ and the constant $C + 1$. $\square$

4. THE COUNTING BOUND

Theorem 4.1 (Dančik–Paterson; [7, Theorem 3.3 and (41)]). *Let $0 < z < 1/2$ and let S be a set of states of M_h that contains q_0 and is closed under the transitions of M_h , that is, $\delta(q, p) \in S$ for every $q \in S$ and every input p accepted at q . If $v: S \rightarrow (0, \infty)$ and $0 < \lambda < 1$ satisfy*

$$(4.1) \quad \sum_{p \text{ accepted at } q} w_z(p) v(\delta(q, p)) \leq \lambda v(q) \quad (q \in S),$$

then $\gamma_2 \leq 2 \ln(2z) / \ln \lambda$.

Proof. Fix n and k . A pair $(A, B) \in \Sigma^n \times \Sigma^n$ with $L(n, n) \geq k$ is determined by the first k pairs $p_1, \dots, p_k$ of its canonical decomposition and by the symbols of A and B after them. By Lemma 3.1 the sequence $\mu(p_1), \dots, \mu(p_k)$ is accepted from q_0 . The total size $s = |p_1| + \dots + |p_k|$ is at most $2n$, and the pair (A, B) has probability 2^{-2n} , so

$$\Pr(L_n \geq k) \leq \sum_{p_1, \dots, p_k} 2^{-s} \leq (2z)^{-2n} \sum_{p_1, \dots, p_k} z^s,$$

where the sums run over the sequences of minimal match pairs of total size $s \leq 2n$ whose truncations are accepted from q_0 ; the first inequality counts the 2^{2n-s} choices of the remaining symbols, and the second uses $2z < 1$. Dropping the condition $s \leq 2n$ and grouping the sequences by their truncations, (3.2) shows that the last sum is at most $e_{q_0}^T A_z^k \mathbf{1}$, where A_z is the $S \times S$ matrix whose entry $A_z(q, q')$ is the total weight of the inputs that take q to q' . By (4.1), $A_z^k v \leq \lambda^k v$, so $e_{q_0}^T A_z^k \mathbf{1} \leq \lambda^k v(q_0) / \min_S v$ (S is finite, since there are 2^{4h} states). Let $\theta > 2 \ln(2z) / \ln \lambda$ and $k = \lceil \theta n \rceil$. Then

$$\Pr(L_n \geq \theta n) \leq \frac{v(q_0)}{\min v} \exp(n(-2 \ln(2z) + \theta \ln \lambda)),$$

which decays geometrically in n . Since $L_n \leq n$, $\mathbb{E} L_n \leq \theta n + n \Pr(L_n \geq \theta n)$, so $\gamma_2 \leq \theta$. Let θ decrease to $2 \ln(2z) / \ln \lambda$. $\square$

The certificates are stated for the reduced automaton; the next lemma transfers them to M_h with the same λ .

Lemma 4.2. *Let S_{red} be a set of reduced states that contains q_0 and satisfies $\delta_{\text{red}}(q, p) \in S_{\text{red}}$ for every $q \in S_{\text{red}}$ and every input p accepted at q . Let $v: S_{\text{red}} \rightarrow (0, \infty)$ satisfy (4.1) with δ_{red} in place of δ . Then $S = S_{\text{red}} \cup \{\bar{q} : q \in S_{\text{red}}\}$ and $v(\bar{q}) = v(q)$ satisfy the hypotheses of Theorem 4.1 with the same λ .*

Proof. The extension of v is well defined because $\bar{q}$ is not reduced when q is, and $v(q') = v(\text{rep}(q'))$ for all $q' \in S$. For $q \in S_{\text{red}}$ and an accepted input p , $\delta(q, p)$ and $\text{rep}(\delta(q, p)) = \delta_{\text{red}}(q, p) \in S_{\text{red}}$ differ at most by complementation, so $\delta(q, p) \in S$, and the left side of (4.1) at q is the same for δ and δ_{red} . At $\bar{q}$, (3.1) gives $\delta(\bar{q}, T_{c,l}) = \overline{\delta(q, T_{\bar{c},l})} \in S$, and since $w_z(T_{c,l})$ does not depend on c ,

$$\sum_{c,l} w_z(T_{c,l}) v(\delta(\bar{q}, T_{c,l})) = \sum_{c,l} w_z(T_{\bar{c},l}) v(\delta_{\text{red}}(q, T_{\bar{c},l})) \leq \lambda v(q) = \lambda v(\bar{q}),$$

where both sums run over the accepted inputs. $\square$

5. CERTIFICATES AND COMPUTATIONS

5.1. Certificates. A certificate for history length h consists of a rational $z \in (0, 1/2)$ given as a decimal, a sorted list of states, and a vector $v > 0$ on it stored as double-precision floating-point numbers, which we read as exact dyadic rationals. Each state is stored as the integer $x2^{3h} + y2^{2h} + d_x2^h + d_y$, where each bit string is read as a binary number with its first symbol

as the most significant bit. The checker computes an integer Λ such that (4.1) holds with $\lambda = \Lambda/2^{64}$.

5.2. The checker. The checker rebuilds the transitions with its own implementation of steps (1)–(5) and of rep: a direct dynamic programming computation on the window that marks which cells are defined, stops if a cell read in steps (3) or (4) is undefined, and asserts that all increments lie in $\{0, 1\}$ and that an accepted input causes exactly one diagonal step. It checks that the list is strictly increasing and consists of integers below 2^{4h} , that it contains q_0 , that every accepted reduced successor of every listed state is listed, and that every entry of v is a positive normal floating-point number. For (4.1) it uses the integer weights $\hat{w}_j = \lceil 2^{64} w_z(T_{c,l}) \rceil$ for $j = |l|$, which a separate script computes from the exact rational z ; the same z is used for the final bound. It writes each entry of v exactly as $m2^e$ with an integer m and evaluates, for every state q , the sum of the terms $\hat{w}_{|l|} v(\delta_{\text{red}}(q, T_{c,l}))$ over the accepted inputs in 128-bit integer arithmetic, scaled to the exponent of $v(q)$; terms that are shifted to the right are rounded up. With $\Lambda_q = \lceil (\text{sum})/m_q \rceil$ it reports $\Lambda = \max_q \Lambda_q$. If a row cannot be bounded in this way, the checker reports $\Lambda = 2^{64} - 1$, which gives no useful bound; otherwise $A_z v \leq (\Lambda/2^{64})v$ holds exactly, and no rounding occurs except the upward roundings described. Finally $2 \ln(2z)/\ln(\Lambda/2^{64})$ is evaluated in interval arithmetic and rounded up to six decimals.

A second checker, written independently from the same definitions, regenerated the state sets for $h = 9, 10, 11$ and 12 by its own breadth-first search and verified these four certificates; in each case the smallest admissible Λ that it computes equals the value reported by the first checker. It also passes the certificate for $h = 8$, and it rejects perturbed certificates: for $h = 9, 10, 11$ and 12 , $\Lambda - 1$ in place of Λ and a slightly larger z ; for $h = 9$ and $h = 11$, a slightly smaller entry of v and a list with one state removed.

5.3. Search. The vectors v come from a power iteration in double precision; any positive vector that passes the check is a valid certificate, and the iteration only serves to make Λ small. The certificate for $h = 12$ in Table 1 is the output of a version of the iteration that starts at the all-ones vector, runs a fixed number of iterations and adds in a fixed order; compiled without contraction of multiply-adds, it rounds every operation separately, and its output does not depend on the processor or the number of threads. The state sets are the sets of states reachable from q_0 ; for $h = 12$ the breadth-first search finds 22,670,087,101 reduced states and 418,139,991,128 accepted transitions. For $h = 12$ the transition table is stored compactly, each target as its rank among the states with the same pair (x, y) , which takes 881 GB; the computation ran on a machine with 288 virtual cores and 2.2 TB of memory, and the exact check took 45 minutes after loading and preprocessing the input. The programs used for $h = 12$ use a faster, bit-parallel transition function (Section 5.4); the checker does not use any code of the search programs.

5.4. Validation of the automaton. For $h = 2$ our implementation reproduces the transition table of the 7-state automaton printed in Lueker’s description of his programs [8]. For $h = 2, \dots, 8$ the numbers of reduced states are 7, 98, 1020, 10160, 89960, 780368 and 6419109, as in [7, Table II], and for $h = 9$ and $h = 10$ they are 51,425,006 and 400,253,317, as in [6]. Our transition tables for $h = 9$ and $h = 10$, written in the format of [6], have the SHA-256 checksums published there. The transition function of the checker and the direct and bit-parallel transition functions of the search programs agree on every input at every reachable state for $h \leq 10$.

5.5. Results. Table 1 lists the certificates. The checker does not verify that the listed states are reduced, and this is not needed: the set R of states reachable from q_0 under δ_{red} consists of reduced states, lies in the list by the closure check, and is closed under δ_{red} , and the inequality (4.1) holds at each of its states. Lemma 4.2, applied to R and the restriction of v , and Theorem 4.1 give the bound of each row, and Theorem 1.1 follows from the row $h = 12$.

TABLE 1. Certified upper bounds on γ_2 from Lueker’s automaton with history length h . The numbers of states and accepted transitions refer to the reduced automaton. The value $\lambda = \Lambda/2^{64}$ is rounded up; the value $2 \ln(2z)/\ln \lambda$, computed with the exact λ , is truncated, and the last column is that value rounded up.

h	states	accepted transitions	z	$\lambda \leq$	$2 \ln(2z)/\ln \lambda$	$\gamma_2 \leq$
8	6,419,109	91,508,397	0.3209	0.341832258719	0.8262791785	0.826280
9	51,425,006	790,078,170	0.3298	0.364708361170	0.8251002729	0.825101
10	400,253,317	6,574,257,580	0.3376	0.385532597280	0.8241195406	0.824120
11	3,045,043,879	53,133,194,463	0.3445	0.404566031012	0.8232896731	0.823290
12	22,670,087,101	418,139,991,128	0.3505	0.421581925965	0.8225783634	0.822579

6. REMARKS

Convergence. By [7, Theorems 3.13 and 3.26] the bounds obtainable from M_h converge to γ_2 as $h \rightarrow \infty$, and for history length h they can be made at most $\gamma_2 + O((\log h/h)^{1/3})$. The improvements in Table 1 decrease slowly, from 0.001179 between $h = 8$ and $h = 9$ to 0.000711 between $h = 11$ and $h = 12$, while the number of states grows by a factor between 7.4 and 8.1 for each step of h .

The choice of z . For each h the parameter z was chosen near the value that minimizes the bound. For $h = 11$, power iterations at $z = 0.3425$, 0.3445 and 0.3465 indicate that the optimal z lowers the bound by less than 10^{-7} , which does not change the rounded value. For $h = 12$ we used only $z = 0.3505$; a floating-point evaluation of the smallest ratio $(A_z v)_q/v_q$ for an iterate of the power iteration, which bounds the Perron root of A_z from below, indicates that no positive vector gives a bound below 0.822578 at this z .

Periodic words. Bukh and Cox [1, Section 5] found a binary word P of length 28 such that the expected length of a longest common subsequence of a uniformly random binary string of length n and the n -letter periodic word with period P is $\gamma_P n + O(\sqrt{n})$ with $\gamma_P \geq 0.82118$. An upper bound $\gamma_2 < 0.82118$ would show that this periodic word is closer to a random string, in this sense, than an independent random string is; they note that the bound $\gamma_2 \leq 0.826280$ does not suffice. Theorem 1.1 does not suffice either.

DATA AVAILABILITY

The checking programs, the search programs, the weights and the logs of the checks are archived at doi:10.5281/zenodo.23517991 and available at <https://github.com/tejasvi/chvatal-sankoff-certificates>. The archive also contains the certificates for $h = 8, 9, 10$ and 11 , compressed losslessly to 28.3 GB. The certificate for $h = 12$ consists of two files of 181 GB each and is not archived; the archived programs regenerate it. A breadth-first search from the start state gives the list of states, whose SHA-256 checksum begins with `bef385e5`, and 240 iterations of the deterministic power iteration with $z = 0.3505$ give the vector, whose checksum begins with `915ac393`; for these two files the checker reports $\Lambda = 7776813894375449830$, and any vector for which it reports $\Lambda \leq 7776819092424481736$ gives the same bound 0.822579. The README of the archive gives the commands, the full checksums and the resources; regenerating and checking the certificate takes less than two and a half hours on a machine with 288 virtual cores and 2.2 TB of memory.

USE OF AI TOOLS

Generative AI tools were used to find and prove the results, to write the programs, and to draft the text. The author reviewed the arguments and the programs and takes responsibility for the content.

REFERENCES

- [1] B. Bukh and C. Cox, *Periodic words, common subsequences and frogs*, Ann. Appl. Probab. **32** (2022), 1295–1332.
 - [2] V. Chvátal and D. Sankoff, *Longest common subsequences of two random sequences*, J. Appl. Probab. **12** (1975), 306–315.
 - [3] V. Dančák, *Expected length of longest common subsequences*, PhD thesis, University of Warwick, 1994.
 - [4] V. Dančák and M. Paterson, *Upper bounds for the expected length of a longest common subsequence of two binary sequences*, Random Structures Algorithms **6** (1995), 449–458.
 - [5] D. Davis, P. Ivanišvili, T. Tao and contributors, *Optimization constants in mathematics*, GitHub repository, <https://github.com/teorth/optimizationproblems>, page `constants/31a.md` at commit `7d762bbb`, 2026.
 - [6] L. Jack, *An improved upper bound on the binary Chvátal–Sankoff constant*, GitHub repository <https://github.com/leandrejack-afk/chvatal-sankoff-bound>, commit `472c18de`, 2026.
 - [7] G. S. Lueker, *Improved bounds on the average length of longest common subsequences*, J. ACM **56** (2009), no. 3, Article 17.
 - [8] G. S. Lueker, *Overview of the code*, with the Java programs for the bounds of [7], <https://www.ics.uci.edu/~lueker/papers/lcs/code/>.
 - [9] D. Soiffer, A. Salls, C. Miller, D. Reichman, G. Sárközy and G. Heineman, *Improved lower bounds on the expected length of longest common subsequences*, in: Proc. 2025 IEEE International Symposium on Information Theory (ISIT), 2025, 1–6; arXiv:2407.10925.
- ORCID: <https://orcid.org/0000-0003-4668-3639>