Visualizing Lucas's Hamiltonian Paths Through The Convex Edge-Flip Graph

Kacey Thien-Huu La
Ursinus College Mathematics And Computer Science, Collegeville, PA, USA

Jose E. Arbelo
Ursinus College Mathematics And Computer Science, Collegeville, PA, USA

Christopher J. Tralie
Ursinus College Mathematics And Computer Science, Collegeville, PA, USA
https://www.ctralie.com

Abstract

We re-examine the 1987 paper by Joan Lucas[4], who showed that the edge-flip graph of convex polygon triangulations is Hamiltonian. We focus specifically on the first part of her paper on Hamiltonian paths, and we provide a simplified algorithm for that case which elucidates how to assemble a recursive subdivision that she refers to as "stacks." Finally, we provide an interactive web-based visualization of Hamiltonian paths through the stacks.

2012 ACM Subject Classification Human-centered computing → Visualization toolkits; Theory of computation → Randomness, geometry and discrete structures

Keywords and phrases associahedron, hamiltonian paths, visualization, tree rotations, convex polygons

Category Media Exposition

Supplementary Material *Software (Source Code)*: https://associahedron.github.io/viz/

1 Convex Flip Graphs

A **flip graph** for the triangulation of a particular polygon has a vertex for each possible triangulation, and an edge between two vertices if their corresponding triangulations differ by a single **quadrilateral edge flip**. In the special case of convex polygon with $n+2$ vertices, the flip graph can be realized as the 1-skeleton of a convex polytope known as the **associahedron** K_n (Figure 1). This polytope has "Catalan number" $C_n = \frac{1}{n+1}\binom{2n}{n}$ vertices, each of whose i^{th} coordinate in $\mathbb{R}^{n+2}$ is the total area of all triangles adjecent to vertex i [1].

Furthermore, since there is a bijection between the triangulations and all possible rooted binary search trees with n internal nodes and $n+1$ leaf nodes, the flip graph is also called the "rotation graph" [4] (Figure 2). In this context, an edge in K_n corresponds to a single **tree rotation**. Thus, a Hamiltonian path through this structure also gives a way to traverse all possible binary trees with n internal nodes via a single tree rotation at each step.

2 Enumerating And Visualizing A Hamiltonian Path

Remarkably, as Joan Lucas shows in her 1987 paper [4], the flip graph is Hamiltonian. To show this, she uses **codeword encodings** of triangulations introduced by Zerling [6] to design an $O(C_n)$ time algorithm for enumerating Hamiltonian paths and cycles. Specifically, the i^{th} entry of a codeword w counts the number of internal edges incident on vertex i that connect to a vertex with a greater index, where $w[0]$ corresponds to the left of the top edge.

[1] Though this embedding is in $n+2$ dimensions, the geometry can be embedded in $n-1$ dimensions.

40th International Symposium on Computational Geometry (SoCG 2024).
Editors: Wolfgang Mulzer and Jeff M. Phillips; Article No. XX; pp. XX:1–XX:7
Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

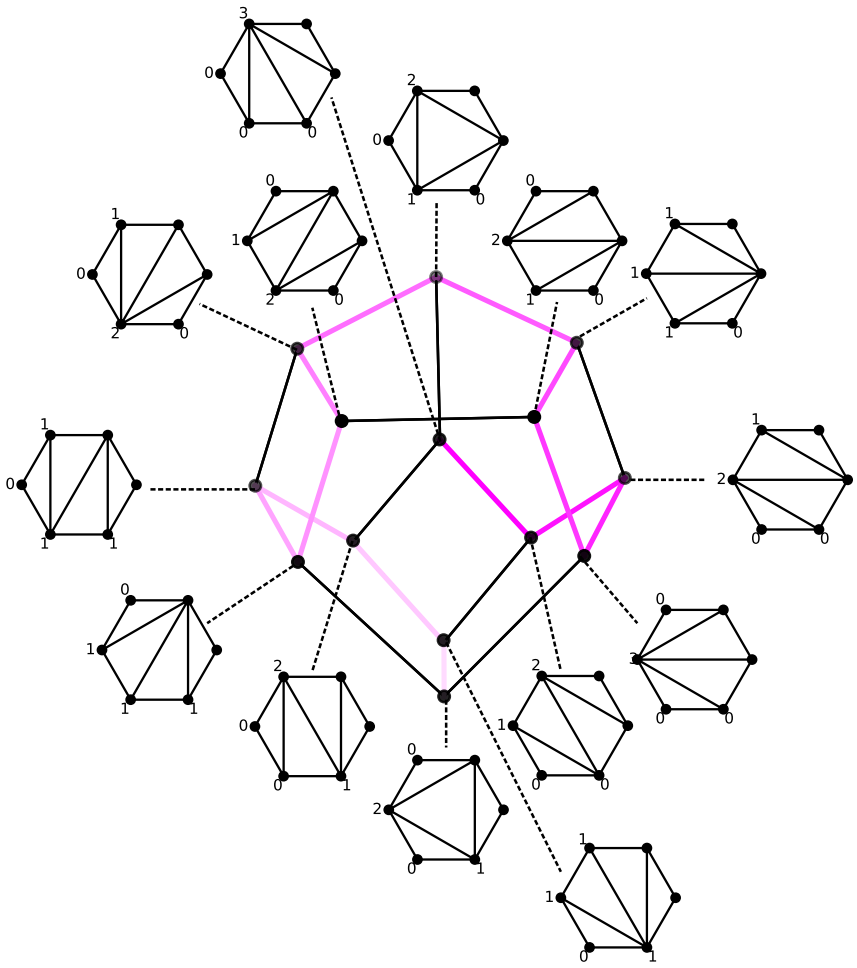

Figure 1 The flip graph is the 1-skeleton of the associahedron. Edges are between triangulations that differ by a single ede flip. In the above, a Hamiltonian path (pink) is shown through all hexagon triangulations in K_4, a 3D polytope. We superimpose Lucas's codewords on each triangulation.

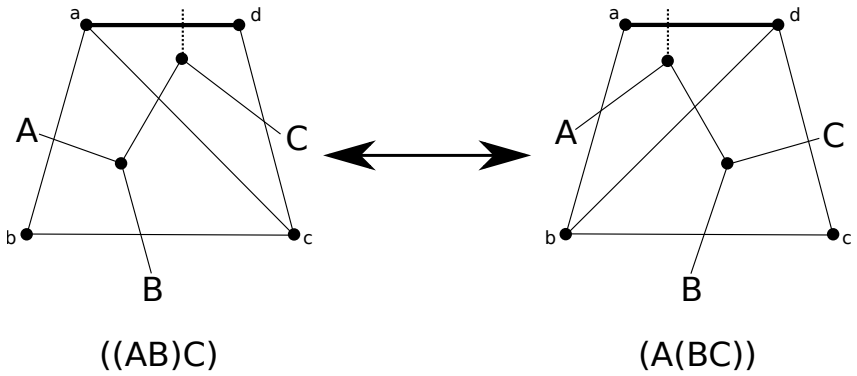

Figure 2 Vertices of the flip graph correspond to triangulations of $n+2$-gons, but also to binary trees with n internal nodes and balanced parentheses on $n+1$ items. Edges are between convex polygons with a quad edge flip, trees with a rotation between them, or parentheses with a single re-association, respectively.

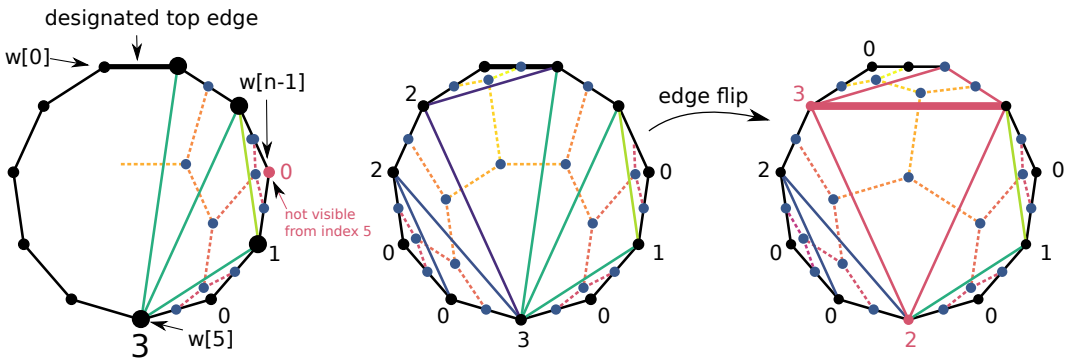

Figure 3 Construct a triangulation from a valid codeword starting at the end and working back in a counter-clockwise fashion, adding $w[i]$ edges to the closest visible vertices at $j > i+1$. Upon completion, extract the corresponding binary tree by putting an internal node in each triangle and connecting nodes belonging to adjacent triangles. Leaf nodes are on the boundary edges of the polygon. An edge flip in a quad decrements one entry of w and increments another.

As such, each internal edge contributes to exactly one entry. As Lucas proves [4], a codeword is a valid representation of a convex polygon triangulation if and only if:

$$\sum_{i=0}^{n-1} w[i] = n-1 \tag{1}$$

and

$$w[i] \leq n - i - \sum_{k=i+1}^{n-1} w[k], 0 < i < n \tag{2}$$

The first condition follows since triangulations of polygons with $n+2$ vertices have exactly $n-1$ internal edges, and the second condition prevents internal edge crossings. Given a valid codeword, one can construct the corresponding triangulation by starting at vertex $i-1$ and moving backwards one vertex at a time to vertex 0, adding $w[i]$ internal edges at each step to the closest visible vertices with indices $> i+1$ (Figure 3).

A necessary condition for a single edge flip between codewords w_1 and w_2 is

$$w_1[a] = w_2[b] \pm 1, w_2[b] = w_1[a] \mp 1, a < b, \text{ and } w_1[k] = w_2[k], k \notin \{a, b\} \tag{3}$$

Intuitively, the vertices at indices a, b are the base of a quadrilateral where an edge flip happens, and they exchange exactly one edge. However, this is not sufficient; the quad should also be free of other internal edges, and the polygonal region below ab should be fully triangulated. Formally, the following two additional conditions are sufficient ([4] lemma 2):

$$\sum_{\ell=k}^{b-1} w_1[\ell] < b-k, a < k < b \tag{4}$$

$$\sum_{\ell=a}^{b-1} w_1[\ell] \geq (b-a) + \delta_{w_1[a]+1, w_2[a]} - \delta_{a,0}, \text{ where } \delta_{x,y} = \left\{ \begin{array}{ll} 1 & x = y \\ 0 & x \neq y \end{array} \right\} \tag{5}$$

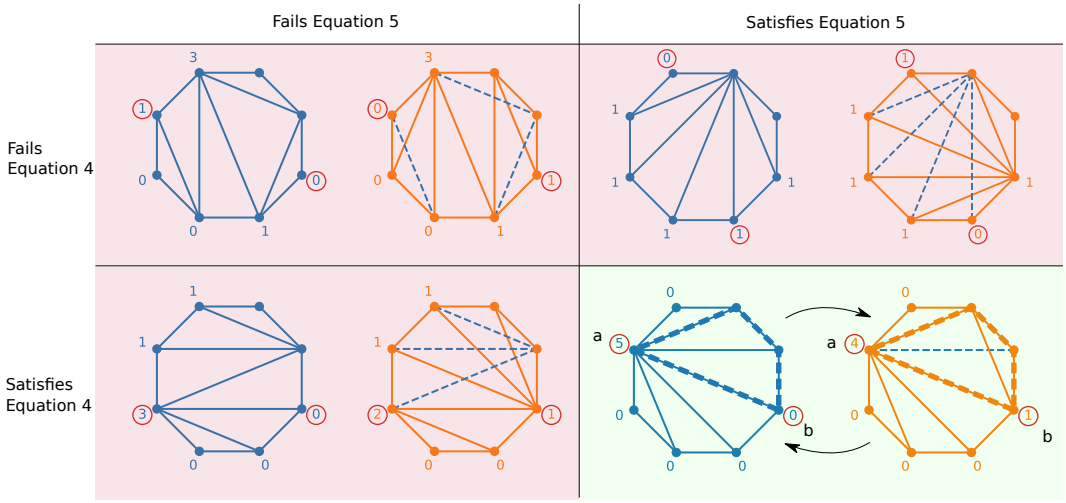

Figure 4 All of the above examples increment one entry and decrement a different entry to move from w_1 to w_2, but not all describe a single valid edge flip in a quadrilateral.

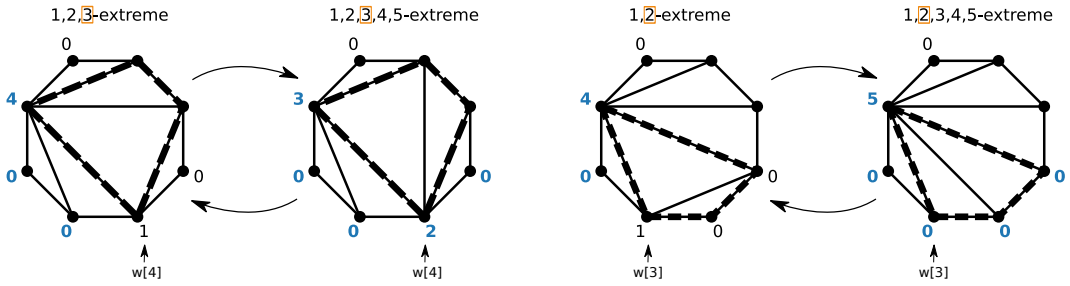

Figure 5 A d-extreme codeword remains d-extreme after an edge flip involving index $d+1$.

```python
def make_stack_rec(w, d, stack_index, wsum):
    n = len(w)
    h = n-d-wsum+1 # where wsum = sum(w[d+1:])
    vals = list(range(h)) # Entry in w[d] for each substack
    if stack_index[d]%2 == 1: # Reverse every other d-dim substack
        vals = reversed(vals)
    stack_index[d] += 1
    for val in vals:
        w[d] = val
        if d == 1: ## Base case
            w[0] = n-1-(w[d]+wsum) # Equation 1 re-arranged
            print(w) # This will print the codewords in Hamiltonian order
        else:
            make_stack_rec(w, d-1, stack_index, wsum+val)
def print_hamiltonian(n): # Print a Hamiltonian traversal of Kn
    make_stack_rec([0]*n, n-1, [0]*n, 0) # Start on an (n-1)-extreme node
```

Figure 6 A simplified $O(C_n)$ algorithm to enumerate a Hamiltonian path through K_n.

The first Kronecker δ case accounts for an edge flip moving a diagonal out of a, and the second δ accounts for the top boundary edge which may be edge ad of the quad $abcd$.

A codeword w is in **d-extreme**, $1 \leq d \leq n-1$, if $w[i] = 0$ or $w[i] = n - i - \sum_{k=i+1}^{n-1} w[k]$ for all $1 \leq i \leq d$. Lucas proves that if an edge flip is possible using index $d+1$ in a d-extreme codeword w_1, then the resulting codeword w_2 is also d-extreme (Figure 5, lemma 3 in [4]).

Finally, Lucas defines a **d-dimensional stack** as the set of valid codewords that share the last $n-d-1$ entries. "Dimension" is apt here, as such a codeword has d degrees of freedom; there are $d+1$ unspecified entries, but $w[0]$ is linearly dependent on the rest given Equation 1. Furthermore, define the **height h of a stack** as the number of possible entries $w[d]$ can take on in a particular d-dimensional stack S. Based on Equation 2, this is

$$h = \left(n - d - \sum_{k=d+1}^{n-1} w[k] \right) + 1 \tag{6}$$

We now sketch Lucas's inductive proof that a Hamiltonian path exists through any stack S (theorem 1 in [4]): *given a d-extreme codeword w_s in a d-dimensional stack, there exists a Hamiltonian path through the stack starting at w_s and ending on a d-extreme codeword w_t.* The base case is $d = 1$, where there are exactly h codewords in S that, when arranged in order of $w[1]$, have adjacent pairs differ by exactly one edge flip (a path in K_n), with 1-extreme codewords at the beginning and end with $w_s[1] = 0$, $w_t[1] = h - 1$. Now consider a stack S with $d > 1$. Partition S into h **substacks** $\{S(i) = \{w \in S | w[d] = i\}\}_{i=0}^{h-1}$, and arrange them in order by i. Let $S(0)$ start on w_s, which is d-extreme, so it is also $d-1$-extreme, and so, by the inductive hypothesis, a Hamiltonian path exists through $S(0)$ ending on a $d-1$-extreme node w_{t0}. One can show that w_{t0} connects to an extreme codeword w_{s1} in $S(1)$ [4]. Start at w_{s1} in $S(1)$ and repeat this process to chain together Hamiltonian paths through all of the substacks, ending on w_t in $S(h-1)$, which is a Hamiltonian path through all of S. Moreover, since w_s and w_t are $d-1$-extreme, and $w_s[d] = 0$ and $w_t[d] = h - 1$, they are d-extreme $\square$.

This theorem can be used inductively to assemble an $n-1$ dimensional stack of height 2 in Hamiltonian order, which contains all vertices of K_n. Figure 6 shows a complete implementation in 16 lines of pure python, which is slightly simpler than Lucas's provided code. The key trick is to reverse adjacent stacks of the same dimension to ensure that the $d-1$-extreme codewords that connect their boundaries are the ones that differ by exactly

one flip. Figure 7 provides more intuition via several example stacks. We also provide a Javascript web app using d3 [1] to animate the Hamiltonian path one step at a time.

References

1 Michael Bostock, Vadim Ogievetsky, and Jeffrey Heer. D^3 data-driven documents. *IEEE Transactions on Visualization and Computer Graphics*, 17(12):2301–2309, 2011.

2 Sean Cleary and Katherine St John. A linear-time approximation for rotation distance.

3 Sean Cleary and Katherine St John. Rotation distance is fixed-parameter tractable. *Information Processing Letters*, 109(16):918–922, 2009.

4 Joan M Lucas. The rotation graph of binary trees is hamiltonian. *Journal of Algorithms*, 8(4):503–535, 1987.

5 Jean Pallo. On the rotation distance in the lattice of binary trees. *Information Processing Letters*, 25(6):369–373, 1987.

6 David Zerling. Generating binary trees using rotations. *Journal of the ACM (JACM)*, 32(3):694–701, 1985.

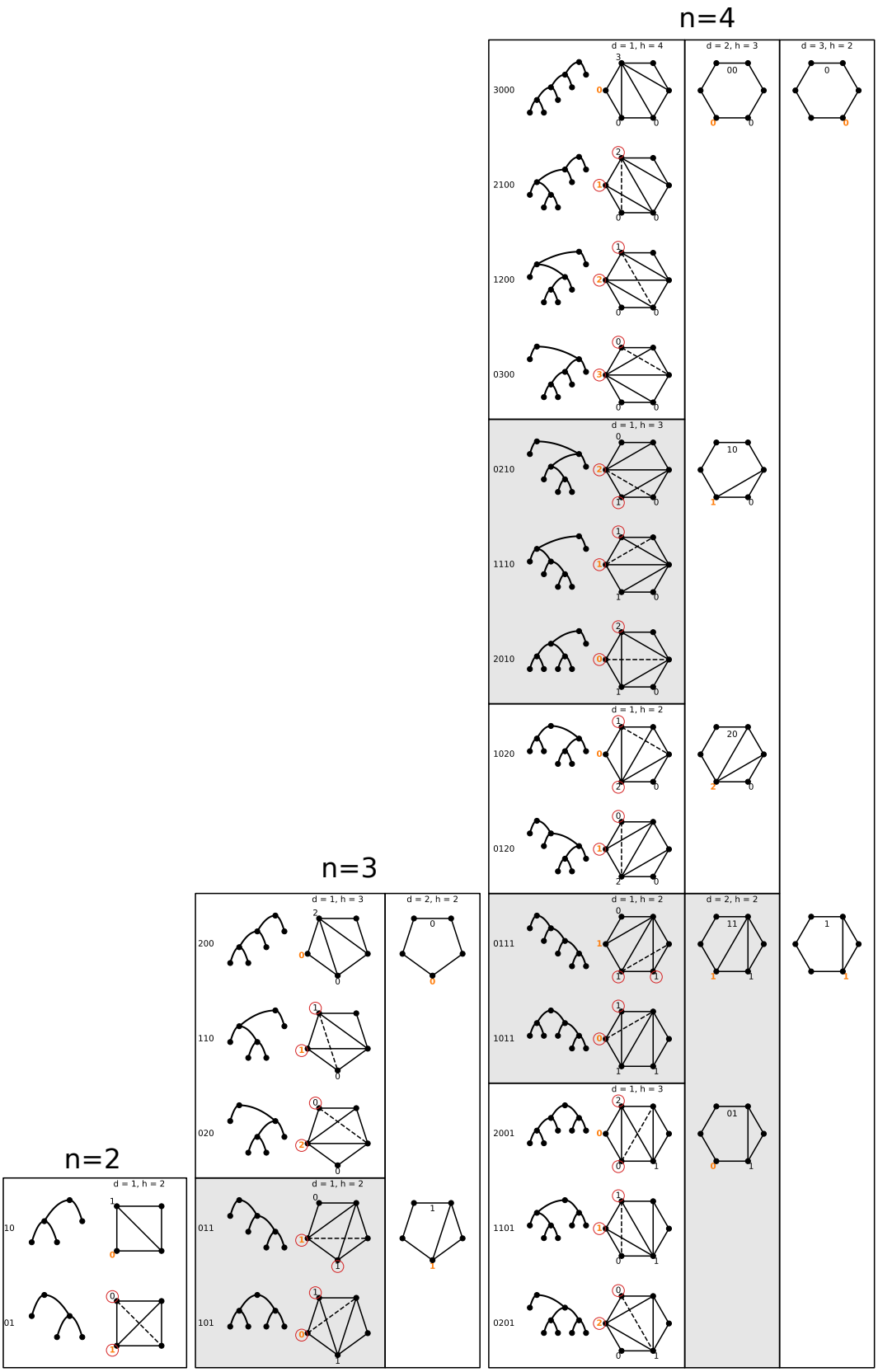

Figure 7 All stacks for $n = 2, 3, 4$ laid out vertically in Hamiltonian order by the algorithm in Figure 6. Flipped edges from adjacent polygons in order are depicted as dashed lines, with changed codeword indices circled. Every other substack is presented in the reverse order (gray) so they fit together at adjacent extreme codewords.