Polytopes

Henrique Hepp, ...

June 19, 2014

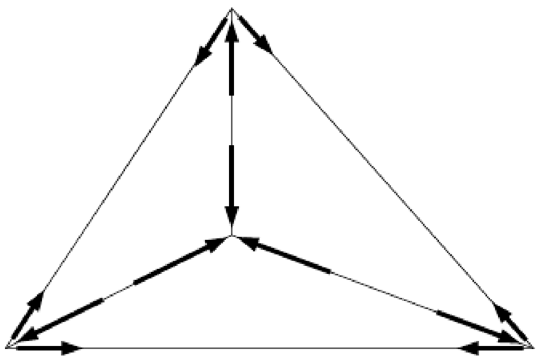
1 Introduction

Theorem 1 (Steinitz, 1922)**.** *A graph G is the edge graph of a 3-polytope if and only if G is simple, planar and 3-connected.*

2 Equilibrium Stresses

Theorem 2 (Maxwell, Whiteley)**.** *Let G be a planar 3-connected graph with 2D drawing $\mathbf{p}$ and designated outer face f_0 . There exists a one-to-one correspondence between*

1. *equilibrium stresses w for G at $\mathbf{p}$; and*
2. *liftings in $\mathbb{R}$, where face f_0 remains in the $z = 0$ plane.*



3 The algorithms

The algorithm based on the article "Embedding Stacked Polytopes on a Polynomial-Size Grid" 2011 can be divided in:

1. Receive the entry.
2. Heavy caterpillar decomposition.
3. Balance the tree.

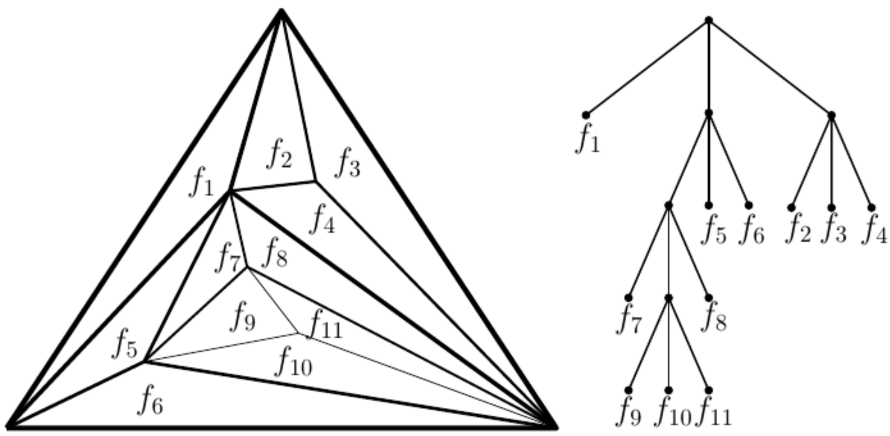
4. Get the coefficients, or weights for the lifting.
5. Lift the graph to a polytope.

The algorithm presented in the definitives version was changed in the two last topics. It can be divided in:

1. Receive the entry.
2. Heavy caterpillar decomposition.
3. Balance the tree.
4. Lift the graph to a polytope.
5. Round the graph to grid points.

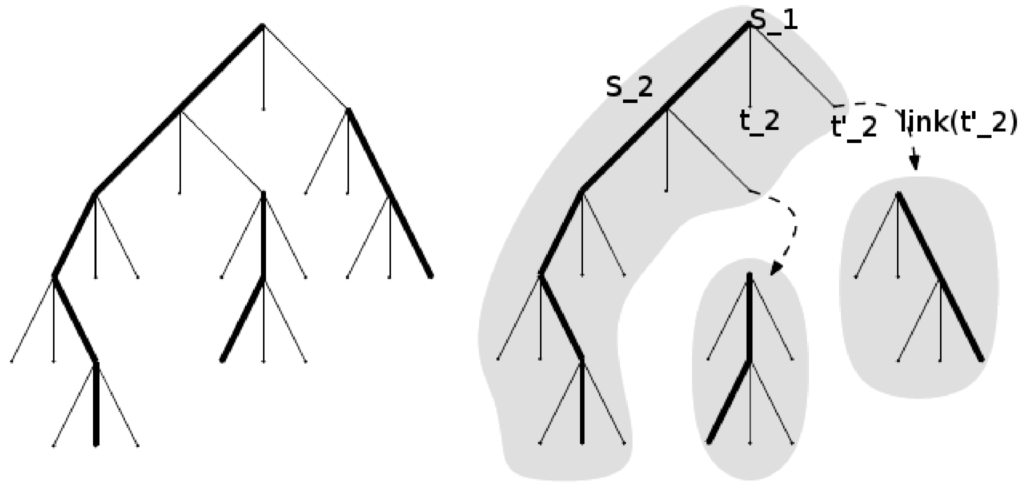
4 The Entry of the algorithm

It is given to the algorithm a 3 connected planar graph and a tree representation of the graph, as showed below.



5 Heavy caterpillar decomposition

The tree representation is decomposed in *heavy paths.* Resulting sub-trees called *caterpillars.* When a node lies on a heavy path it is called a *spine node*, otherwise is a *tree node.* The spine nodes are labelled by s_1 (root), s_2, ..., s_i,...,$s_\perp$. The children from s_i are s_{i+1}, t_{i+1} and t'_{i+1}. In t are stored a pointer link(t) to a other caterpillar.



6 Balance the tree

The balacing of the Tree is given by the pseudo-code below:

function BALANCE(C)
  Input: A caterpillar C from the heavy caterpillar decomposition of $\mathcal{T}(G)$. All weights are equal 1.
  Output: Weights for the nodes of $\mathcal{T}(G)$.
  for all t_i, t'_i in C **do**
    BALANCE(link(t_i))
    BALANCE(link(t'_i))
    if $w(t_i) > w(t'_i)$ **then**
      relabel $t_i \leftrightarrow t'_i$
    end if
    $w(t_i) = w(t'_i)$
    add $(w(t_i) - w(t'_i))$ to the weight of $s_\perp$ in link (t_i)
    add $(w(t'_i) - w(t_i))$ to the weight of $\text{link}^{-1}(C)$
  end for
end function

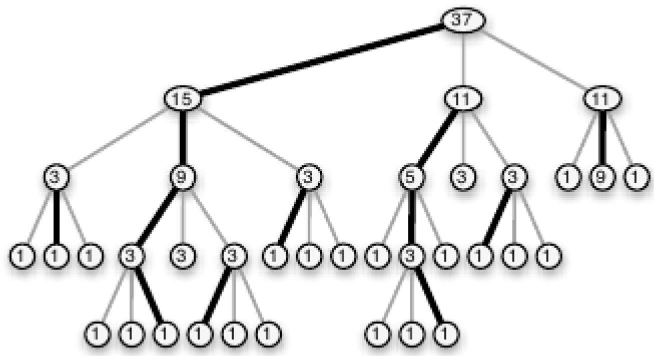
A balanced Tree will the have the properties:

$$w(s_{i-1}) = w(s_i) + w(t_i) + w(t'_i)$$

$$w(s_i) \geq w(t_i), w(t'_i)$$

$$w(t_i) = w(t'_i)$$

A example of a balanced tree is show next:



7 First algorithm

7.1 Get the coefficients

If v_i is the vertex stacked on the face $v_j v_k v_l$ then:

$$\alpha_{ijkl} = \frac{1}{w(t_u)}$$

Where $w(t_u)$ is the weight of the subface $v_j v_k v_l$ obtained by the function BALANCE.

The weights are now updated with the coordinate values. First the coordinates from the vertices are rounded down. $r_i = (\lfloor x_i \rfloor, \lfloor y_i \rfloor)$.

The weights are:

$$\dot{w}_{ij} = \sum_{\{i,j,k,l\} \in S} \lfloor Y \alpha_{ijkl} \rfloor w_{ij}^{kl}$$

I have to find out how to make that sum

The coefficient α_{ijkl} is scaled by multiplying it with $Y = 4n^2$, n is the number of vertices.

$$w_{ij}^{kl} = [i,k,l][j,k,l]$$

$$[i,j,k] = \det \begin{pmatrix} x_i & x_j & x_k \\ y_i & y_j & y_k \\ 1 & 1 & 1 \end{pmatrix}$$

7.2 The lifting

Each face i gains a parameter a_i in that way:

For the firs face is set:

$$a_0 = (0,0,0)^T$$

Having the parameter of the rigth face (the first is a_0) then we can find all the others with:

$$a_l = \dot{w}_{ij}(\mathbf{q}_i \times \mathbf{q}_j) + a_r$$

Where (i, j) is the common edge of f_i and f_l and $\mathbf{q}_i$ is defined as:

$$\mathbf{q}_i = (x_i, y_i, 1)^T$$

Now to calculate the height from a point $\mathbf{p}_i$ we have to make the inner product from $\mathbf{p}_i$ with a_k. Who is the parameter from face k whose one of the vertices is $\mathbf{p}_i$.

$$h_i = \langle \mathbf{p}_i, a_k \rangle$$

8 Second algorithm

8.1 Calculate the shifts

As the vertex v_i is stacked on a face f_D is new height is:

$$z = \zeta_i + z_D$$

Where z_D is the height of the point in the face D and the shift ζ_i is:

$$\zeta_i = A_i \cdot B_i$$

Where A_i and B_i are the two possible weights of the three new faces formed by the vertex v_i. Remember $w(t_i) = w(t_i')$.

But before we calculate the new heights (z_i) we round the coordinates in the embedding, and then also the heights.

8.2 Round the graph to grid points

The coordinates on the embedding are all rounded so that they are multiples of 1/pert, where:

$$\text{pert} = 240n^{\frac{3}{2}}$$

Now we lift, rounding, the points so that the heights $z = \zeta_i + z_D$ are multiples of $1/\text{pert}_z$, where:

$$\text{pert}_z = 3n$$

After these points are rounded they are still real numbers, not intengers. To make them integer we multiply they with pert.

To make the rounding we calculate:

$$\lfloor x \cdot \text{pert} \rfloor \div \text{pert}$$

The final value in integer is:

$$\lfloor x \cdot \text{pert} \rfloor \div \text{pert} \cdot \text{pert} = \lfloor x \cdot \text{pert} \rfloor$$