7 st-ordering & incremental orthogonal drawing

Topological ordering of a *directed* graph is an ordering so that for each $(u, v) \in E$, u comes before v in the ordering.

st-ordering is an ordering of the vertex set s.t. each v_j for $2 \le j \le n-1$ has at least one neighbour v_i with $i < j$ and a neighbour v_k with $k > j$.

Thm Let G biconnected. Then, for given $s, t \in V$, an st-ordering can be computed in linear time. *Proof:*

1. Orient edges of G to create an $s-t$-graph G_{st} by calculating an oriented open ear decomposition.
2. The topological numbering of G_{st} is then an st-ordering.

Open ear decomposition $D = (P_0, \ldots, P_r)$ of an undirected graph is a partition of E into an ordered collection of edge-disjoint paths s.t.

- P_0 is an edge
- $P_0 \cup P_1$ is a simple cycle
- both end vertices of P_i are in $P_0 \cup \ldots \cup P_{i-1}$
- no internal vertex of P_i is in $P_0 \cup \ldots \cup P_{i-1}$

Computation: Perform a DFS. Ears will be s, t, and backwards edges and suffixes of the spanning tree path leading up to the backwards edge. Orient edges like other edge at "source" node of backwards edge.

alg (Incremental orthogonal drawing) Prerequisites: max degree 4.

8 Straightline drawings & canonical ordering

Canonical Ordering (cf. st-ordering) of a triangulated graph is an ordering of the vertex set V, s.t.

- v_1, v_2, v_n are on the outer face
- for any v_j, $j \in \{3, \ldots, n-1\}$, it is adjacent to

  1. at least two vertices with a lower index, and
  2. at least one vertex with a higher index.

Computation: (*"peeling order"*) For $k = n, \ldots, 3$, pick v_k from $\partial(G_k) \backslash \{v_1, v_2\}$ so that v_k not incident to a chord in $\partial(G_k)$ *(because else if incident to cord, i.e. triangle with two edges in $\partial(G_k)$ $\rightarrow$ removing leaves only one edge in $\partial(G_k)$)*. **Existence**: minimal chord (extreme case: triangle) has at least one vertex in $\partial(G_k)$ that is not part of a chord.

Thm. A canonical ordering of a triangulated planar graph can be computed in linear time. *i.e. can pick chord-free vertices in linear time*. *Proof:*

- counter CHORDS(v) for $v \in \partial(G_k)$
- list CHORDFREE to consider

Pop vertex from CHORDFREE, update counter of neighbours: total runtime $\mathcal{O}(\sum deg(v)) = \mathcal{O}(2m)$. Insert vertex into CHORDFREE if counter is 0.

Let G_j be the graph induced by $\{v_1, \ldots, v_j\}$.

Prop. v_{j+1} is on the outer face of G_j. *Proof:* There ex. path P from v_{j+1} to v_n whose internal vertices come after v_{j+1} in the ordering, in particular P does not contain vertices of G_j! — v_n is on the outer face of G and thus of G_j. — Assume v_{j+1} is in an inner face: In any embedding, P has to cross the boundary of G_j, and this cannot happen in a vertex. Hence, we have an edge crossing, contradiction to planary of G.

Prop Edges connecting v_j to G_j are consecutive among the edges connecting G_j to $G \backslash G_j$ *(i.e. there is no edge in between that is incident to $w \in G \backslash G_j$)*. *Proof:* Follows from the above proof.

Prop. If G is triangulated, the boundary $\partial(G_j)$ is a simple cycle. *Proof:* Induction.

Applications

- Kandinsky Drawings
- Straighline drawings for planar graphs

8.1 Straightline drawings for planar graphs

SHIFT-Algorithm of Fraysseix-Pach-Pollack.

Basic Idea

- Suffices to consider triangular graphs (else triangulate, layout, then remove added edges)
- Construct drawing step-by-step by incrementally adding a single vertex and its incident edges, potentially shifting the rest of the graph s.t. no crossings appear.

Alg (SHIFT-Algorithm)

1. Begin with one triangle, place v_1 on $(0, 0)$, v_2 on $(2, 0)$ and v_3 on $(1, 1)$. *(slopes on outer face always -1, 1)*
2. According to canonical ordering, add a new vertex v_k. Let v_l be its leftmost (first on P) adjacent neighbour, v_r the rightmost. Add v_k in center s.t. slopes on outer face are unity.
3. To avoid edge overlaps, move v_l, v_r and all nodes left (resp. right) further away. *(Consequently, v_k will be placed higher up)*

- Edges from v_k to $w \in G_{k-1}$ can not create crossings with edges in $\partial(G_{k-1})$ *Proof:* Slope of $f \in \partial(G_{k-1}) \leq 1$ always and slope of other $e \geq 1$. Because x-coordinates are monotonous, there cannot be such an edge because there cannot be an endpoint at the implied position.
- v_k is on a grid point if $x + y$ is even (because slope 1).
- We have shown that newly added part does not introduce crossing. Now still have to show that shifting does not introduce new crossings on the inside, i.e. in G_{k-1}. We distinguish cases defined by the leftmost vertex to be shifted.
  - For vertices $v \neq v_k$, it must necessarily be some $v \in \partial(G_{k-1})$ by construction of the algorithm (do use nodes that are adjacent to v_k except for v_l and v_r). Can directly apply IH.
  - For $v = v_k$, using it as target for shift is the same as using the second-to-leftmost adjacent v'. But this is the same case as for G_{k-1}.
  - Shifting edges incident to v_k does not introduce crossings because they and their neighbourhood move rigidly with v_k.

Prop. Manhattan distance between two vertices on $\partial(G_k)$ is even. *Proof:* Distance remains or increases by two

Prop. SHIFT runs in linear time.

- y-coordinates can be computed from canonical ordering.
- for x-coordinates (of v_k) maintain a forest of *rigid edges* (middle edges that never appear in the boundary) *(such a tree is always shifted as one (per definition))* and boundary edges of current G_j.