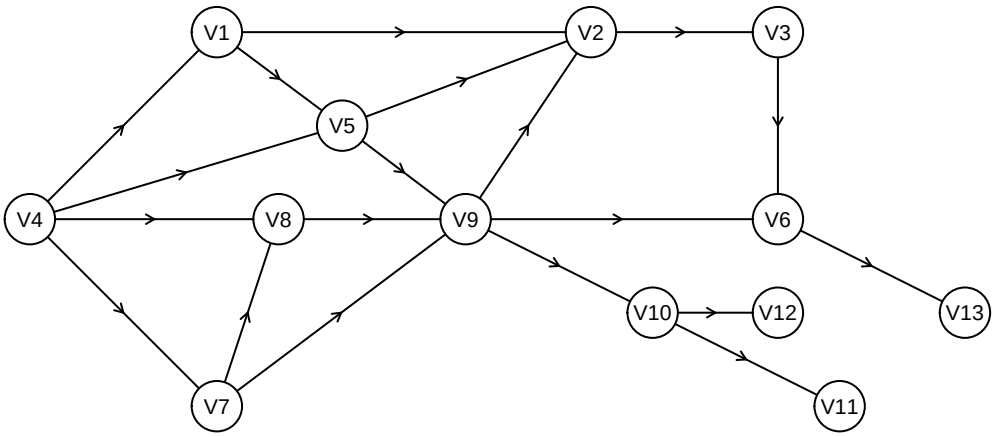
Target Vertices Processing

Guillaume Guénard

2025-01-28

Examplary graph

```
gr_ex2 <- readRDS("gr_ex2.rds")
plot(gr_ex2, cex.min=3.5, cex.lab=0.7)
```



A 13-vertex 19-edge exemplary graph used for this discussion. —

Target calculation

Cases outlined in this document are for a vertex V_T (target) preceded by n_p vertices $V_1^{(P)}$, $V_2^{(P)}$, …, $V_{n_f}^{(P)}$, and followed by n_f vertices $V_1^{(F)}$, $V_2^{(F)}$, …, $V_{n_f}^{(F)}$. The edges linking V_{P_i} and V_T are referred to as $E_1^{(P)}$, $E_2^{(P)}$, …, $E_{n_p}^{(P)}$ and their associated distance are $D_1^{(P)}$, $D_2^{(P)}$, …, $D_{n_p}^{(P)}$, whereas that linking V_T to V_{F_i} are referred to as $E_1^{(F)}$, $E_2^{(F)}$, …, $E_{n_f}^{(F)}$, and their associated distances are $D_1^{(F)}$, $D_2^{(F)}$, …, $D_{n_f}^{(F)}$. From these definitions it follows that a vertex is:

- convergent when $n_p > 1$ or non-convergent when $n_p = 1$,
- divergent when $n_f > 1$ or non-divergent when $n_f = 1$, and
- terminal when $n_f = 0$ or non-terminal when $n_f > 0$.

When evaluating the loading of a target vertex on the phylogenetic eigenfunctions, the later may or may not remain, dictated by the dependence of other vertices upon it. Non-convergent and non-divergent (**NCND**) vertices are always removable (both terminal and non-terminal), whereas convergent or divergent vertices are not. Non-convergent, but divergent vertices may become non-divergent upon removal of other vertices located after them, and thus any non-convergent vertices may eventually become removable. There is no way for convergent vertices to become removable.

When a target vertex is not removable, its `$species` nodal value is set to `FALSE` and its influence coordinates are used to calculate the loadings. A removed vertex does no longer belongs to the graph, but maintains an ancestral reference to one of the graph's vertex, or to an intersection point within one the the graph's edge. Two possibilities exist. A terminal $V^{(T)}$ can be removed along with $E_1^{(P)}$ and its influence coordinates are that of $V_1^{(P)}$ with a distance from the latest ancestor of $D_{la} = D_1^{(P)}$. For a non-terminal $V^{(T)}$, $E_1^{(F)}$ is deleted and $E_1^{(P)}$ is extended to reach $V_1^{(F)}$ (denoted $E_1^{(P-F)}$, its new distance becomes $D_1^{(P-F)} = D_1^{(P)} + D_1^{(F)}$) and its coordinates are located on $E_1^{(P)}$ at distance $D_1^{(P)}$ with a distance from the latest ancestor of $D_{la} = 0$.

An ancestral reference to an edge thus also involves a value of distance between the origin of that edge and the intersection point.

A vertex can be removed if and only if no other vertices is linked to it. Since convergent and divergent vertices have other vertices before and after them, they cannot be removed. Non-convergent vertices that with $n_f \leq 1$ such as terminal and intermediary vertices can be removed. Also, an erstwhile divergent vertex that have become a terminal or intermediary vertex following the removal of one or more of its tributaries can be removed.

Algorithm:

1. Check 1 – verify that all the targets are marked as species, stop with an error message if any is not thus marked.
2. Simplification 1 – going through all the vertices; verify if any vertex that is marked as no being a species can be removed readily. When one is found, it is removed and the procedure is restarted from the beginning. This step ends when going through all the vertices that are not marked as a species and finding none that can be removed. This step is optional, yet recommended, and is accomplished using an external function.
3. From the set of all target vertices, verify if there is a terminal vertex among them if there is one, go to step 5, otherwise go to 4.
4. From the set of all target vertices, verify if there is an intermediary vertex among them, if there is one, go to step 5, otherwise go to 6.
5. Remove the vertex earmarked for removal in step 3 (terminal vertex) or 4 (intermediary vertex), then go to step 3.

   Simplification 2
6. From the set of all non-target vertices, verify if there is a terminal vertex among them, if there is one, go to step 8, otherwise, go to 7.
7. From the set of all non-target vertices, verify if there is an intermediary vertex among them, if there is one, go to step 8, otherwise, to to 9.
8. Remove the vertex earmarked for removal in step 6 (terminal vertex) or 7 (intermediary vertex), then go to step 6.

   Non removables

9. If there are any target vertex remaining, they cannot be removed.

Pass 2: Identify the terminal vertices

- Remove the terminal targets
- When there is no more

C, whereas . On the other hand, strictly divergent (ie. a non-convergent) vertex may

divergent vertices may become an intermediary of a terminal vertex upon removal one or more of its tributaries. This situation calls for an iterative approach for target loading calculation.

The root and convergent vertices cannot be removed

The calculation of the loadings needs to be organized in such a way that terminal and intermediate vertices are treated first

either a surrogate of an other vertex under which it used to located or occurring at some location within an edge. (with some ancestral distance)

When a vertex is collapsed, it either becomes a surrogate of an other vertex

or it is

Case : Intermediary vertex

The latter is arguably simplest case; if one wants to estimate the loading for an intermediary vertex, vertices $V_1^{(P)}$ and $V_1^{(F)}$ can simply be connected directly using a new edge with length $D_1^{(P)} + D_1^{(F)}$. The coordinates are that of $V_1^{(P)}$ at a distance of $D_1^{(P)}$ along the new edge and a latest ancestor distance $D_{anc} = 0$ is applied because the vertex is indeed exactly within the edge. This is the case of strings of any number of intermediary vertices.