An importance ranking in a network of dependencies

Nir Y. Krakauer

December 30, 2020

Consider a set of N programs indexed 1 to N, where some programs depend on others (directly or indirectly (transitively); a program is never a dependency of itself). This is described by the $N \times N$ boolean dependency matrix (adjacency matrix of the corresponding directed graph) $\mathbf{D}$, where $d_{i,j}$ is 1 if program j depends on program i and 0 otherwise. Assume each program has intrinsic value v_i. We can find a program importance p_i that is the sum of its own value and the importance of all programs which depend on it by solving the linear system

$$\mathbf{Dp} + \mathbf{v} = \mathbf{p},$$

which can be rewritten as

$$(\mathbf{I} - \mathbf{D})\mathbf{p} = \mathbf{v}$$

where $\mathbf{I}$ is the $N \times N$ identity matrix.

This is the same formalism as PageRank, except that there the intrinsic values are all zero and each $d_{i,j}$ is a transition probability from page j to page i.

If a partial dependency matrix is available that has at least all the direct dependencies marked, it can be filled out by applying a transitive closure algorithm, most simply by applying (until a full pass finds nothing more to fill in) the rule that for all i, j, k, if indices (i, j) and (k, i) are dependencies, so is (k, j). This simple rule has order N^3 computational cost though, so more efficient implementations should be considered to bring the cost down to order N^2 where the number of programs considered is large.

Assuming (reasonably) no codependency, the programs and their transitive dependencies form a directed acyclic graph, and a theorem states that we can order the programs such that i is a dependency of program j only if $j > i$ (a so-called topological sort). This would make the matrix $\mathbf{D}$, and with it

$\mathbf{I} - \mathbf{D}$, (upper) triangular, so that we can solve for $\mathbf{p}$ in at most N^2 arithmetic operations.