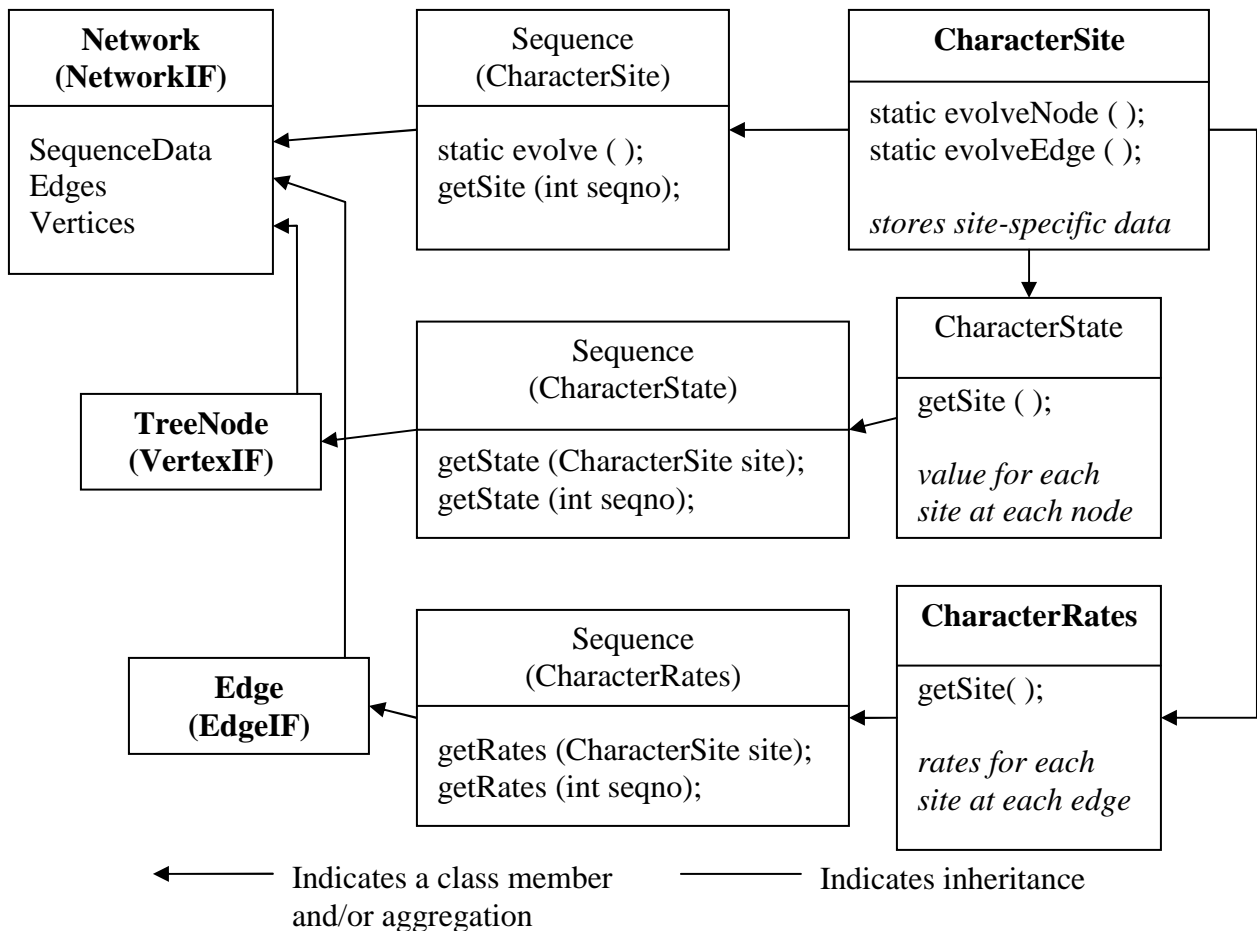


Class Diagrams for Network Evolution



Existing objects are in **bold face**
 Planned objects are in normal type

NetworkIF, EdgeIF, VertexIF are network data structures

CharacterSites: stores character and site specific data such as character type: lexical, phonological, morphological, number of states, evolution law along edges, etc...

CharacterRates: two sets of numerical parameters (vertical and horizontal transmission)

CharacterStates: two types of values (homoplastic, non homoplastic)

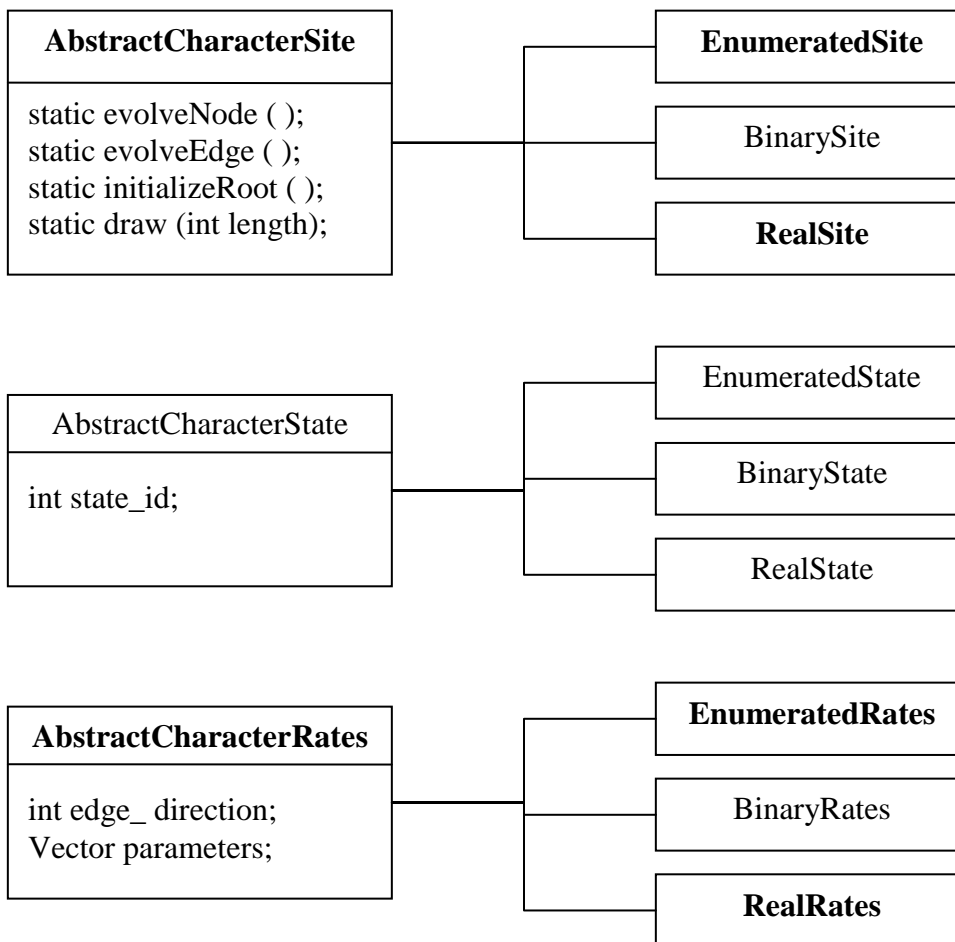
Sequence: Sites, States and Rates are ordered in sequences. Normally the site is the index allowing retrieval. As a speed-up we can also use an integer seqnum. In that case, all 3

indexes must be consistent so that the same seqnum implies same reference Site. There will be a consistency check during index construction.

EnumeratedCharacter: Lexical and (morphological ?), an infinite number of states, one homoplastic state, transmission/borrowing on contact edges. 5 parameters: $p(n,h)$, $p(n,m)$, $p(n,n)$, $p(h,n)$, $p(h,h)$.

BinaryCharacter: (Phonological ?), two states, homoplasy, transmission/borrowing on contact edges.

RealCharacter: floating point valued character, no transmission/borrowing. Brownian motion. 1 parameter: amplitude (linked to variance).



← Indicates a class member and/or aggregation

— Indicates inheritance

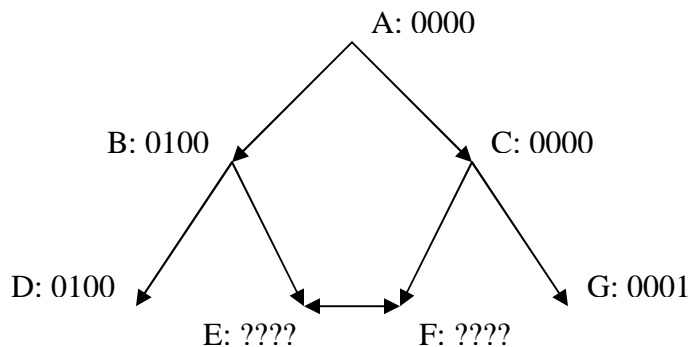
Existing objects are in **bold face**
 Planned objects are in normal type

Resolving Cycles in the Network

As long as the network is acyclic, evolution can easily be computed from the root: the network represents a dependency graph, such that we can compute the evolutionary value of every node in the graph by respecting a temporal precedence relationship.

Cycles pose a problem. They represent a deadlock in the dependency graph such that there is no order the nodes can be computed in that respects the precedence relationship expressed by the oriented edges network.

Unfortunately every contact edge creates a cycle of size 2: in the graph illustrated here, there is a bi-directional contact edge between nodes E and F. This introduces the obvious cycle (E > F > E) such that there is no possible ordering of the pair (E, F) that is consistent with the precedence relationship expressed by the oriented edges of the network.



At this point, I can only envision two ways of simulating evolution for E and F. For now, I am implementing Solution 1, but I am taking great care that this can easily be changed (at least as long as there are no cycles of size greater than 2).

Solution 1: Independent Character Evolution

Have a random draw to determine a *single* direction for the edge (E, F). This would reflect the assumption that any given character can be transmitted either left to right or right to left but not both. This assumption would not work for sequence evolution in

general, since we can imagine that on the same sequence evolution, character c_1 goes left to right while c_2 goes right to left on the edge (E, F). However the sequence evolution can be computed character by character, by having each character evolve independently.

Solution 2: Statistical Model of Sequence Transmission

Have a statistical process of simultaneous evolution/transmission for E and F and come up with a model predicting the resulting equilibrium and allowing us to compute the evolution at E and F, given sequences B and C and edge-lengths (B,E) and (C,F) (those lengths could be independent, equal or zero). We could for example, have a single length $\|(B, E)\| = \|(C, F)\|$ and come up with a Markov chain, where the atomic event is the exchange of one sequence character in a random direction, with a Poisson draw. Randomly draw two equilibrium states for E and F (but wouldn't that just be a hybridization producing two hybrids?).