NetLogo NW Extension (Beta 0.02) — Cheat Sheet

For download and complete documentation, see:
https://github.com/NetLogo/NW-Extension

General Primitive

nw:set-snapshot *turtleset linkset*

Builds a static internal representation of the network formed by all the turtles in *turtleset* and all the links in *linkset* that connect two turtles from *turtleset*. This network snapshot is the one that will be used by other primitives until a new snapshot is created. Example:

```
nw:set-snapshot turtles links
```

Centrality Primitives

nw:betweenness-centrality, nw:eigenvector-centrality, nw:closeness-centrality

These primitives calculate different centrality measures for a turtle. Example:

```
nw:set-snapshot turtles links
ask turtles [ set size nw:betweenness-centrality ]
```

Distance and Path-Finding Primitives

nw:distance-to *target-turtle*
nw:weighted-distance-to *target-turtle weight-variable-name*

Finds the shortest path to the target turtle and reports the total distance for this path, or false if no path exists in the current snapshot. The `nw:distance-to` version of the primitive assumes that each link counts for a distance of one. The `nw:weighted-distance-to` version accepts a *weight-variable-name* parameter, which must be a string naming the link variable to use as the weight of each link in distance calculations. The weights cannot be negative numbers.
Example:

```
nw:set-snapshot turtles links
ask turtle 0 [ show nw:distance-to turtle 2 ]
ask turtle 0 [ show nw:weighted-distance-to turtle 2 "weight" ]
```

nw:path-to *target-turtle*
nw:turtles-on-path-to *target-turtle*
nw:weighted-path-to *target-turtle weight-variable-name*
nw:turtles-on-weighted-path-to *target-turtle weight-variable-name*

Finds the shortest path to the target turtle and reports the actual path between the source and the target turtle. The `nw:path-to` and `nw:weighted-path-to` variants will report the list of links that constitute the path, while the `nw:turtles-on-path-to` and `nw:turtles-on-weighted-path-to` variants will report the list of turtles along the path, including the source and destination turtles. As with the link distance primitives, the `nw:weighted-path-to` and `nw:turtles-on-weighted-path-to` accept a *weight-variable-name* parameter, which must be a string naming the link variable to use as the weight of each link in distance calculations. The weights cannot be negative numbers. If no path exist between the source and the target turtles, all primitives will report an empty list. Examples:

```
nw:set-snapshot turtles links
ask turtle 0 [ show nw:path-to turtle 2 ]
ask turtle 0 [ show nw:turtles-on-path-to turtle 2 ]
ask turtle 0 [ show nw:weighted-path-to turtle 2 "weight" ]
ask turtle 0 [ show nw:turtles-on-weighted-path-to turtle 2 "weight" ]
```

nw:turtles-in-radius *radius*
nw:turtles-in-out-radius *radius*
nw:turtles-in-in-radius *radius*

Returns the set of turtles within the given distance (number of links followed) of the calling turtle in the current snapshot. The `turtles-in-radius` form works with undirected links. The other two forms work with directed links; out or in specifies whether links are followed in the normal direction (`out`), or in reverse (`in`). Example:

```
nw:set-snapshot turtles links
ask turtle 0 [ show sort nw:turtles-in-radius 1 ]
```

nw:mean-path-length
nw:mean-weighted-path-length *weight-variable-name*

Reports the average shortest-path length between all distinct pairs of nodes in the current snapshot. If the `nw:mean-weighted-path-length` is used, the distances will be calculated using *`weight-variable-name`*. The weights cannot be negative numbers. Reports false unless paths exist between all pairs. Examples:

```
nw:set-snapshot turtles links
show nw:mean-path-length
show nw:mean-weighted-path-length "weight"
```

Clusterers and Clique Finder Primitives

nw:k-means-clusters *nb-clusters max-iterations convergence-threshold*

Partitions the turtles in the current snapshot into nb-clusters different groups. It uses the *x y* coordinates of the turtles to group them together, not their distance in the network. Example:

```
nw:set-snapshot turtles links
let clusters nw:k-means-clusters 10 500 0.01
```

nw:bicomponent-clusters

Reports the list of bicomponent clusters in the current network snapshot. The result is reported as a list of lists of turtles. One turtle can be a member of more than one bicomponent at once. Example:

```
nw:set-snapshot turtles links
let clusters nw:bicomponent-clusters
```

nw:weak-component-clusters

Reports the list of "weakly" connected components in the current network snapshot. The result is reported as a list of lists of turtles. One turtle cannot be a member of more than one weakly connected component at once. Example:

```
nw:set-snapshot turtles links
let clusters nw:weak-component-clusters
```

nw:maximal-cliques

A clique is a subset of a network in which every node has a direct link to every other node. A maximal clique is a clique that is not, itself, contained in a bigger clique. The result is reported as a list of lists of turtles. One turtle can be a member of more than one maximal clique at once. The primitive uses the Bron–Kerbosch algorithm and only works with undirected links. Example:

```
nw:set-snapshot turtles links
let cliques nw:maximal-cliques
```

nw:biggest-maximal-clique

The biggest maximal clique is, as its name implies, the biggest clique in the current snapshot. The result is reported a list of turtles in the clique. If more than one cliques are tied for the title of biggest clique, only one of them is reported at random. Example:

```
nw:set-snapshot turtles links
let biggest-clique nw:biggest-maximal-clique
```

Generator Primitives

nw:generate-preferential-attachment *turtle-breed link-breed nb-nodes [commands]*
nw:generate-random *turtle-breed link-breed nb-nodes connection-prb [commands]*
nw:generate-small-world *turtle-breed link-breed rows cols exp toroidal? [commands]*
nw:generate-lattice-2d *turtle-breed link-breed rows cols exp toroidal? [commands]*
nw:generate-ring *turtle-breed link-breed nb-nodes [commands]*
nw:generate-star *turtle-breed link-breed nb-nodes [commands]*
nw:generate-wheel *turtle-breed link-breed nb-nodes [commands]*
nw:generate-wheel-inward *turtle-breed link-breed nb-nodes [commands]*
nw:generate-wheel-outward *turtle-breed link-breed nb-nodes [commands]*

The generators are amongst the only primitives that do not operate on the current network snapshot. Instead, all of them take a turtle breed and a link breed as inputs and generate a new network using the given breeds. Examples:

```
nw:generate-preferential-attachment turtles links 100 [ set color red ]
nw:generate-random turtles links 100 0.5 [ set color green ]
nw:generate-small-world turtles links 10 10 2.0 false [ set color blue ]
nw:generate-wheel turtles links 100 [ set color yellow ]
```

Import/Export Primitives

nw:save-matrix *file-name*

Saves the current network snapshot to *file-name*, as a text file, in the form of a simple connection matrix. At the moment, nw:save-matrix does not support link weights. Every link is represented as a "1.00" in the connection matrix. Example:

```
nw:set-snapshot turtles links
nw:save-matrix "matrix.txt"
```

nw:load-matrix *file-name turtle-breed link-breed [commands]*

Generates a new network according to the connection matrix saved in *file-name*, using turtle-breed and link-breed to create the new turtles and links. Please be aware that the breeds that use use to load the matrix may be different from those that you used when you saved it. Example:

```
nw:load-matrix "matrix.txt" turtles links
```

nw:save-graphml *file-name*

Saves the current snapshot in the GraphML format, including every attribute of the turtles and links included in the snapshot. Example:

```
nw:set-snapshot turtles links
nw:save-graphml "graph.xml"
```