11.3.2 Graph representing a computer network

Description

Here we sketch a very simple example representing the structure of a computer network using class `Graph`. A `Node` in the graph represent a computer (server) and an `Edge` represents a cable.

```
class Network: Graph
   class Node::<
      send(receiver: ref Node,P: ref Packet):
         :::
      receive(send: ref Node,P: ref Packet):
         ...
      :::
      ...
   class Edge::<
      bandWidth: var integer
      send(receiver: ref Node,P: ref Packet):
         ...
      ...
   class Packet(S: var String):
       ...
```

- Class `Network` is a subclass of `Graph`.
- It has further bindings of `Node` and `Edge` from `Graph`.
- It adds a class `Packet` representing the packets being communicated in the network.

Class `Node` is extended with:

- A method `send` that send the `Packet P` to the node represents by `receiver`.
- A method `receive` that receives a a `Packet P` from the `Node` represented by receiver.

Class `Edge` is extended with:

- An `integer bandWidth` representing the bandwidth of the cable represented by the `Edge`.

Th details of send in `Node` may be sketched as follows:

```
send(receiver: ref Node,P: ref Packet):
   map(receiver).send(receiver,P)
```

`Send` uses an ancillary method `map` to find the `edge` representing the cable that has the fastest connection to the `receiver`, and the invokes `send` on this `Edge`.

We do not add further details of this examples. It is major task to implement a computer network. It also involves using parallel programming for listening on input/output ports for packages being transmitted / received. Parallel programming is described in chapter