UNPACKING ALGORITHM

Additional DORA's parameters

- binary 'tokenize' parameter
    - tokenize=True: to unpack roles into different analogs with a separate copy of the object in each OR
    - tokenize=False: to unpack all roles into one analog with one copy of the object
- binary 'remove_compressed' parameter which allows to choose whether to delete or to leave the compressed structure in LTM

basicRunDORA.py

- **do_unpacking()** function
    - starts with finding compressed predicates in the driver -- the ones connected to higher-order semantics
    - for each compressed predicate the firingOrder list is created: the predicate itself plus the object it is bound to (POs)
    - POs in firingOrder fire one at a time
    - each PO in firingOrder is active for several iterations (until local inhibitor fires)
    - **unpacking_routine()** function is called several times for each PO while that unit is active
        - hoSemCount is calculated equal to the number of higher-order semantics connected to the current compressed predicate
        - on the first run unpacking_routine() recruits hoSemCount number of new RB units (made_RBs)
        - for the compressed predicate in firingOrder hoSemCount number of new predicate units are recruited
        - while compressed predicate PO is active, each newly recruited predicate unit iteratively learns weighted connections to the regular semantics connected to one of the PO's higher-order semantics
        - each newly recruited predicate learns connection to one of the made_RBs
        - binary 'tokenize' parameter is used to decide whether one copy of the object will be bound to the unpacked predicates or whether a separate copy will be bound to each unpacked predicate
        - for object PO in firingOrder hoSemCount number of new object units are recruited, copies of the PO
        - while object PO is active, the newly recruited PO units iteratively learn weighted connections to the PO's semantics
        - each newly recruited object learns connection to one of the made_RBs
    - **bind_others_to_unpacked ()** function collects the rest of the analog in the driver
        - the rest of the PO units which did not participate in unpacking_routine() recruit new units to create copies of themselves
            - using **infer_PO()** function
        - these copies are put together with the made_RB unit into a full proposition

        - the copies of the POs learn weighted connections with regular semantic units iteratively
- the unpacked part is also bound to the newly recruited Ps (if the original proposition had a P unit)
- new analog/analogs -- depending on the 'tokenize' parameter -- is created and populated with all the newly recruited units