TODOs for GBG-Framework – WK/05/2017

From TR-GBG.pdf (to be deleted there):

* The n-tuple agent developed for C4 (Connect Four) needs to be ported to GBG.
* Allow only trained agents to be saved.
* Clarify: Is the parameter data flow safe, if we issue a ’play’ or ’compete’

for 2 agents of same type but with different parameters?

Further things:

* Read the section in TR-GBG.pdf on game value / game score and function for accessing it. Can the interface be made simpler? (less functions which are very similar and thus hard to distinguish?)
* Prepare TR-GBG.pdf for publication on CIplus Server
* Saving (serializing) of things (logs, agents) is still not safe, if something changes in the classes being serialized. Is it possible (with minimal effort) to read older versions of serialized objects as well?
* Bug: The current LoadAgent version ‘hangs’ if an agent with incompatible serialID is read (TTT agents TDS & TD-NTuple) 🡪 Fix this first.
* Hex: extend to logging with subdirs
* Types.ACTIONS: replace enum (which needs to be extended whenever we need more ACTIONS) by some class construct
* OK: Bug fix in TDAgent.java: We needed to replace the hard-coded   
  private int inpSize[] = { 6, 6, 10, 19, 13, 19, 0, 0, 0, 9 };  
  by the flexible m\_feature.getInputSize(m\_feature.getFeatmode()).
* OK TD-pars tab: Constrain ‘Feature set’ (= value for featmode) to the set of allowed featmode values for the current game (m\_feature.getAvailFeatmode()) and set the initial value to a sensible value 🡪 KG fixed this and generated a pull request
* During TDNTupleAgt-training: Replace the Minimax-evaluation by proper general evaluation (replace JFreeChart plot title)
* OK: BUG1 + BUG2 fix for MCAgent: In some versions of Hex, the MC agent does not choose the best action, but **instead the worst** action (!!) This happens for (3x3, 1.), (4x4, 2.), (5,5, 1.), … and so on. The same behavior for TTT (3x3, 1.) – This was due to 2 errors: (1) in RandomSearch there could be actions on newSob even if newSob is already a game-over-state (BUG1) and (2) newSob.getGameScore has to be called with the referingState sob to get the sign right in every case (BUG2)
* OK Improve TR-GBG around game score and game value: Make it simpler!
  + Concentrate Chap. 3.3 on so.getGameScore() and so.getGameScore(sob). Explain the difference in pictures. Give examples showing explicit values for both functions.
  + Move pa.getGameScore(so) to Chap. 3.4 (Game Value), because it is a game value, not a game score (think about renaming it to pa.getGameValue). Move the mind-buggling complicated so.getGameValue() and pa.estimateGameValue(so) to an appendix.
* Think about the game logic for so.getGameScore(referingState) in 3-player games (might depend on whether the game is a win-terminating or loose-terminating game)
* If MCTS has several actions with the same value (all are a ‘Win’), then it does not take the shortest path to victory. Instead it takes a random among all winning moves. 🡪 KG will think about an add-on to value function which breaks ties in favor of a shorter path.