**Computation Niche model simulation code**

**List of Functions**

© 2018, Richard J.Carter

|  |  |  |
| --- | --- | --- |
| **Function Name** | **Used In**  CN – computation niche  novel – unconstrained multi-state computation niche | **Description** |
| calcInputDistributionv5.m | CN, novel | Determine the cumulative input received at a membrane automata’s incoming edges |
| checkDuplicates.m | novel | Sometimes the minimisation algorithm can leave duplicate transitions in a finite state automata. This simply removes any duplicate transitions. |
| checkInteraction.m | novel | Check whether the interaction of two automata generates a valid automaton |
| checkIsomorphic.m | novel | Check that a newly produced automata is isomorphic |
| checkLanguageCoverage.m | novel | Checks that a newly produced automata (Tc) can read the same language as the Ta automata that produced it. |
| checkMachineDim.m | novel | Count the number of states in an automata (machine) |
| checkNullTypes.m | novel | Housekeeping function |
| checkStronglyConnected.m | novel | Checks that an automata is strongly connected (part of validating that it is an epsilon-machine) |
| checkUnifilarity.m | novel | Checks that an automata is unifilar i.e. there are no duplicate transitions per state of an automata |
| checkWaitingList.m | novel | Part of the Hopcroft minimisation algorithm implementation in MATLAB |
| compareLists.m | novel | Used by the ‘findLists.m’ function. |
| composeMachinesRevised.m | CN, novel | Performs composition of two automata. |
| convertL2D.m | CN, novel | Converts an automata list to a digraph. The digraph is used to calculate the structural complexity of an automata. |
| convertList2Y.m | CN, novel | Converts an automata description from the list format (e.g. [1 1 1]) to an outgoing probability distribution (e.g. [1 0]) |
| convertPartition2List.m | novel | Part of the Hopcroft minimisation algorithm. Converts partitioned equivalence classes to conventional automata representation as a list. |
| convertPopCell2Matrix.m | novel | Converts the ‘popDynamics’ cell array to a |T| x Z matrix where Z is the number of iterations of the simulation |
| createSigmaSet.m | CN, novel | Creates four binary vectors each of length |T|. Each vector represents a symbol pair ‘0|0’,’0|1’,’1|0’,’1|1’ and each entry in the vector with a ‘1’ represents that the automata type Ti has a transition of that type. |
| findList.m | CN, novel | Searches the list of all automata types (T) currently in the population. Returns a ‘0’ if not found, or ‘1’ otherwise. |
| findPartitionSet.m | novel | Part of the Hopcroft minimisation algorithm. |
| getNextState.m | novel | Called by the stateTransition.m function. |
| initCNv2.m | CN, novel | Initialise the membrane network. |
| initialiseCY.m | CN, novel | Initialise the output range of each membrane automata. |
| inverseList.m | novel | Part of the Hopcroft minimisation algorithm. |
| minList.m | novel | The Hopcroft minimisation algorithm that also incorporates the necessary validation checks to ensure that the resulting, minimal automaton is an epsilon-machine. |
| nkCheck.m | novel | Part of the tests for a valid epsilon-machine. |
| performCompositionRevised.m | novel | Performs the interaction of two automata and the subsequent minimisation of the resultant automata. |
| produceMachinev5\_unconstrained.m | novel | Produce machines without any constraint. Used by the open-ended multi-state population simulations. |
| produceMachinev5.m | CN, novel | Produce machines but do not allow new automata types to be introduced. |
| pruneTc.m | novel | Remove unreachable states from a newly produced automata. |
| rebuildCNv5.m | novel | Re-construct the membrane network as a new automata type has been generated. |
| reLabel.m | novel | The minimisation algorithm can remove transitions and states and so the remaining states/transitions need to be relabelled. |
| seedAutomata.mat | CN, novel | The MATLAB data file containing the interaction network for the seed population. |
| setActive.m | CN, novel | Determine which membrane automata are activated on this time-step. |
| stateTransition.m | CN, novel | As a multi-state membrane automata is activated it transitions to a different state. |
| updateCNv5\_unconstrained.m | novel | Update the membrane network e.g. add new membrane automata nodes, update edge weightings |
| updateCNv5.m | CN, novel | Update the membrane network e.g. update edge weightings only |