SDM'Studio

ANR PLASMA

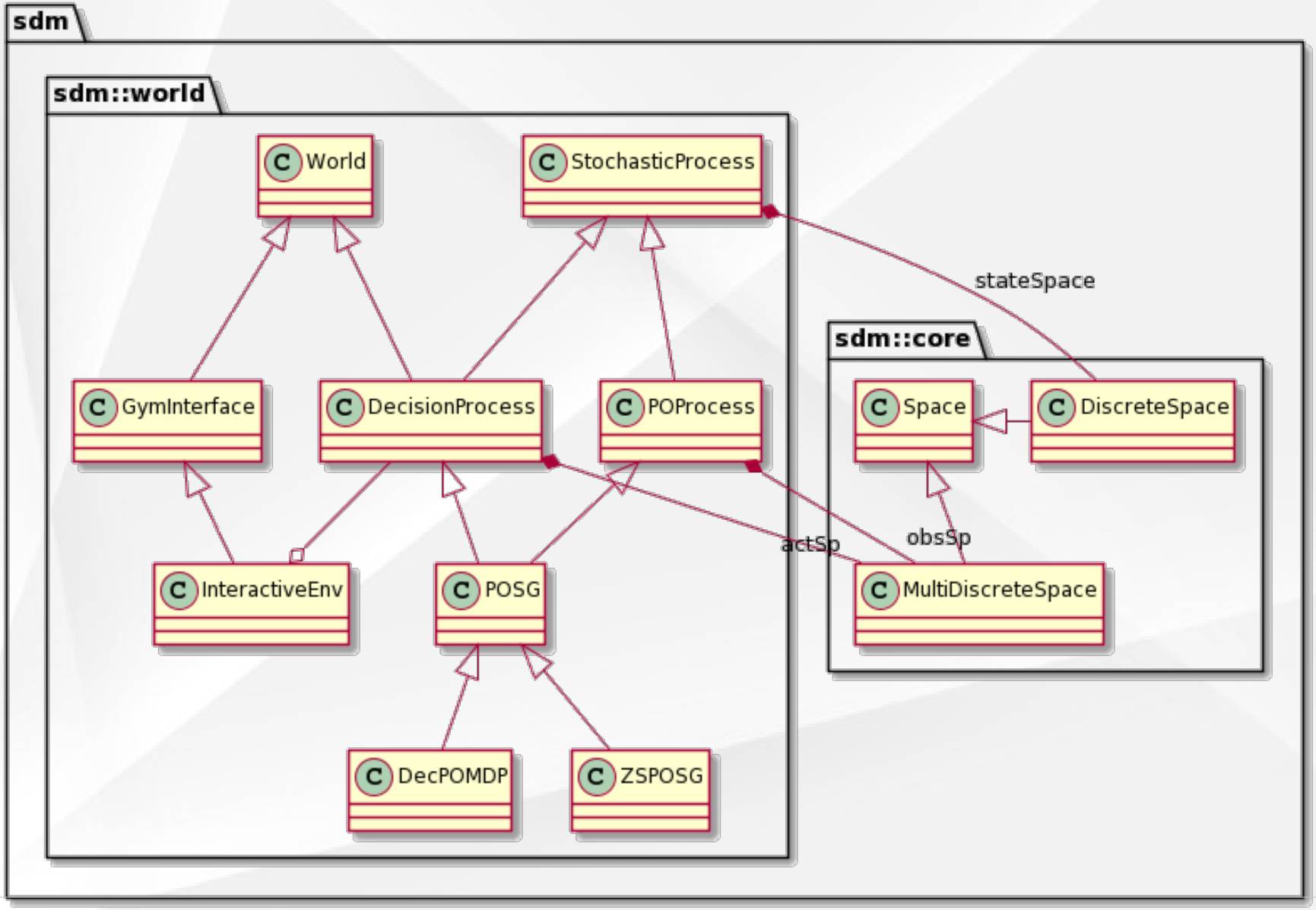
Jilles-Steeve Dibangoye, David Albert

TABLE OF CONTENTS

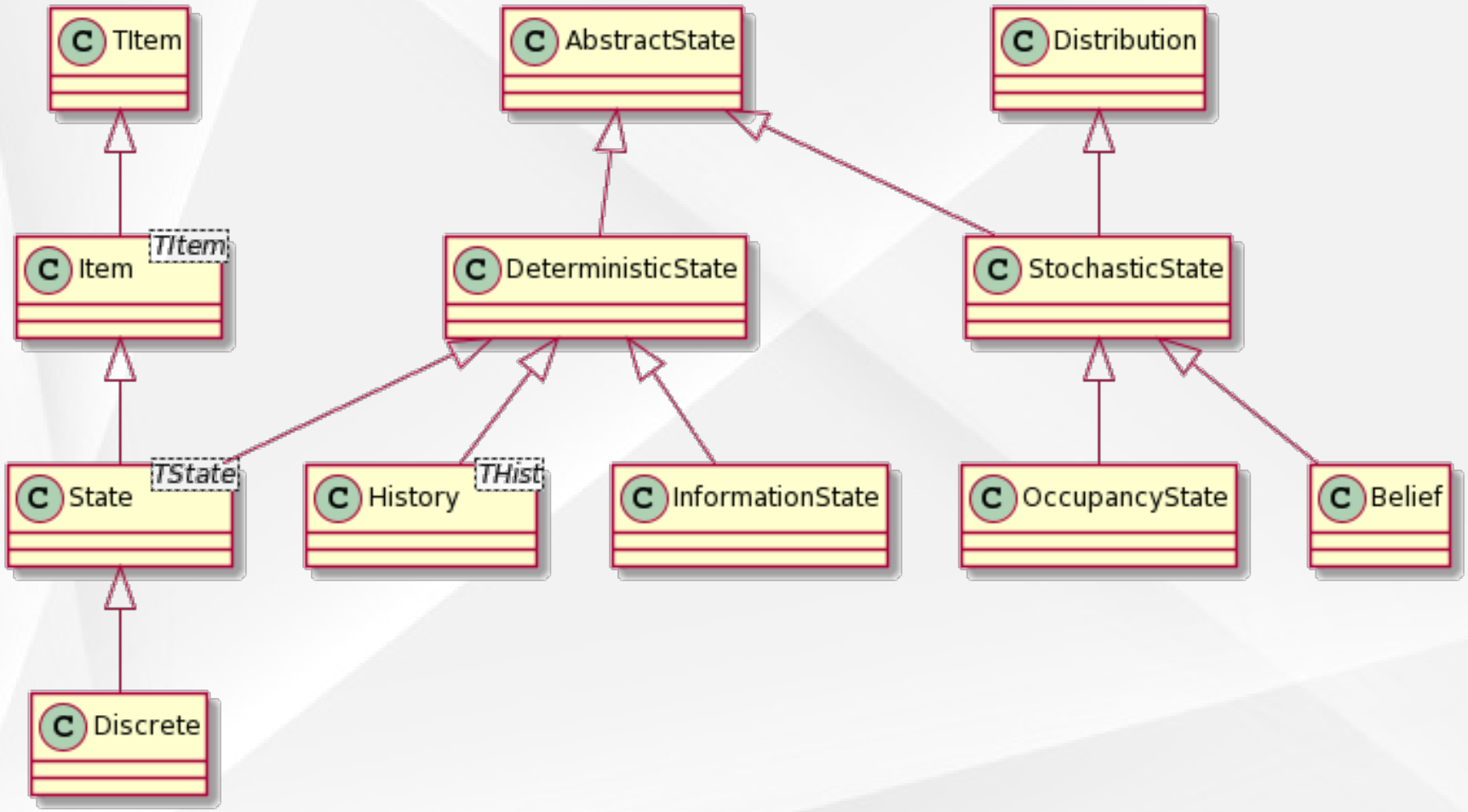
Installation

Basic Usage

Worlds - Class Diagram

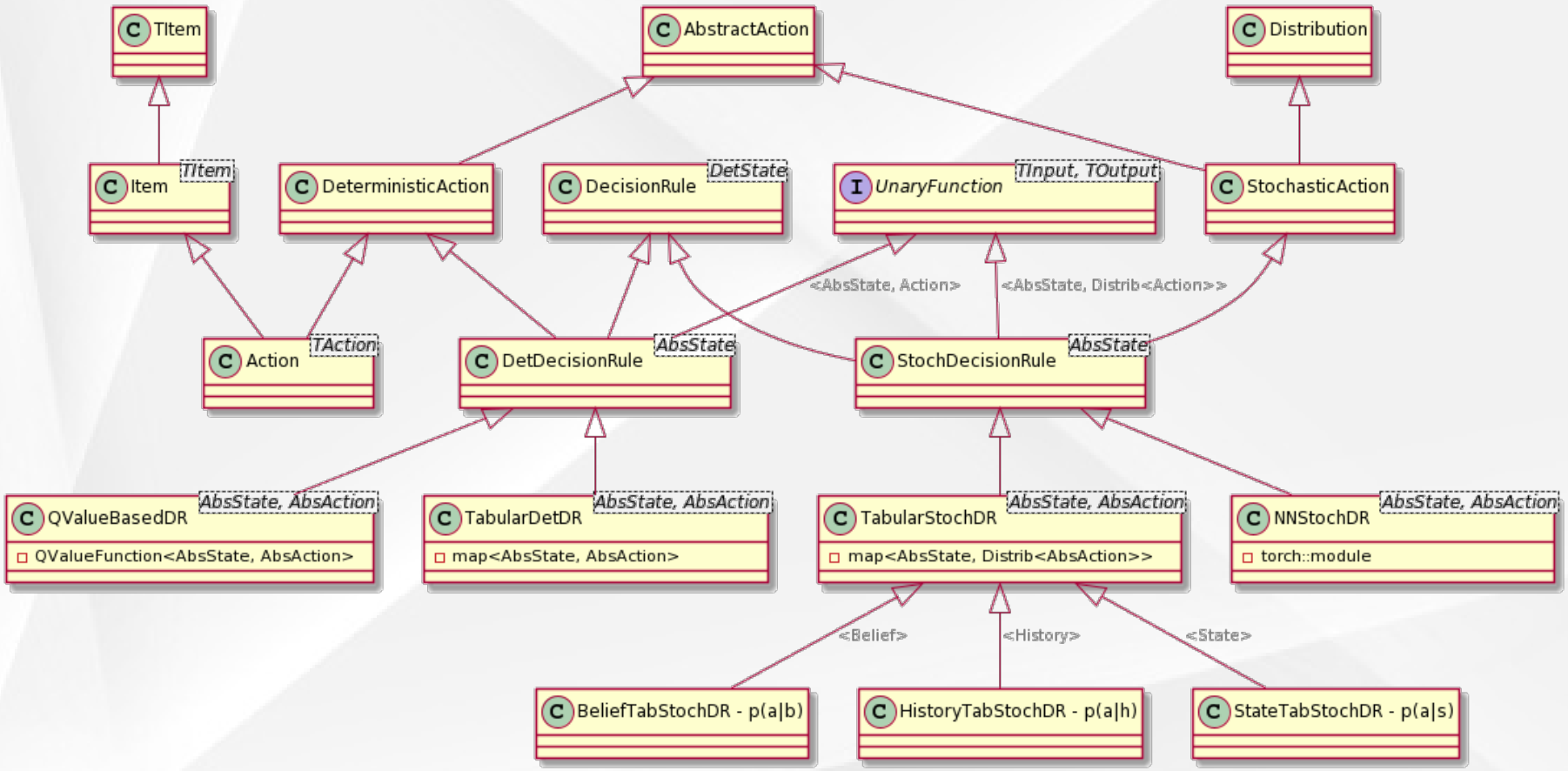


Generic States - Class Diagram



- *AbstractState* $\Leftrightarrow$ *GenericState*
  - *State* $\Leftrightarrow s_t$
  - *History* $\Leftrightarrow$ *GenericHistory*
    - *ObsHistory* $\Leftrightarrow h_t^{indiv} = (z_1, z_2, \ldots, z_t)$
    - *ActObsHistory* $\Leftrightarrow h_t^{indiv} = (a_0, z_1, a_1, \ldots, a_{t-1}, z_t)$
    - *JointObsHistory* $\Leftrightarrow h_t^{joint} = (\mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_t)$
    - *JointActObsHistory* $\Leftrightarrow h_t^{joint} = (\mathbf{a}_0, \mathbf{z}_1, \mathbf{a}_1, \ldots, \mathbf{a}_{t-1}, \mathbf{z}_t)$
  - *InformationState* $\Leftrightarrow \iota_t = (b_0, d_1, ..., d_t)$
  - *Belief* $\Leftrightarrow b_t = p\left(s_t \mid h_t^{indiv}\right)$
  - *OccupancyState* $\Leftrightarrow \xi_t = p\left(s_t, h_t^{joint} \mid \iota_t\right)$

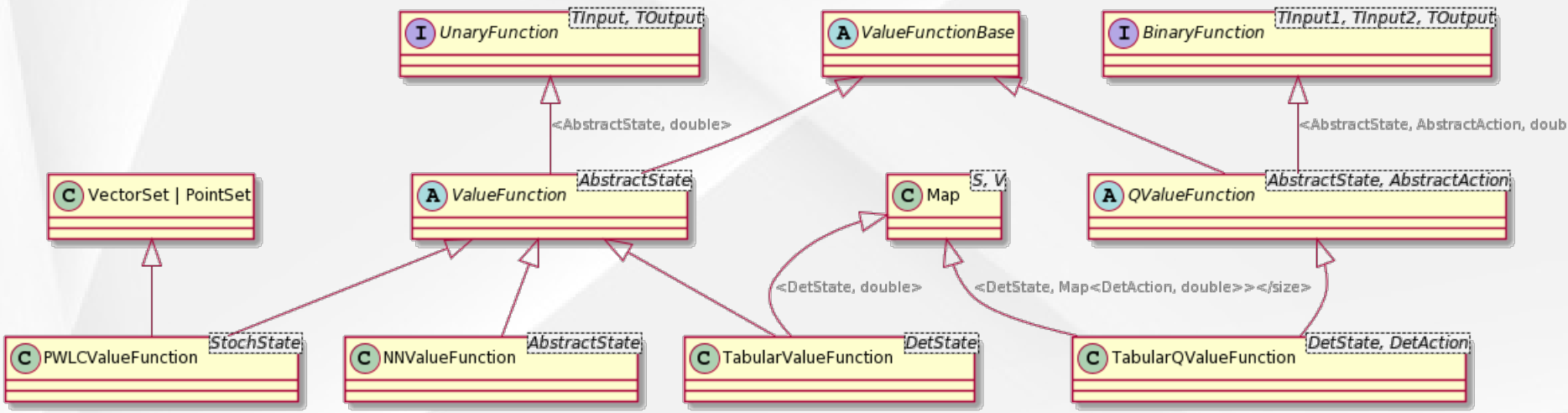
Generic Actions - Class Diagram



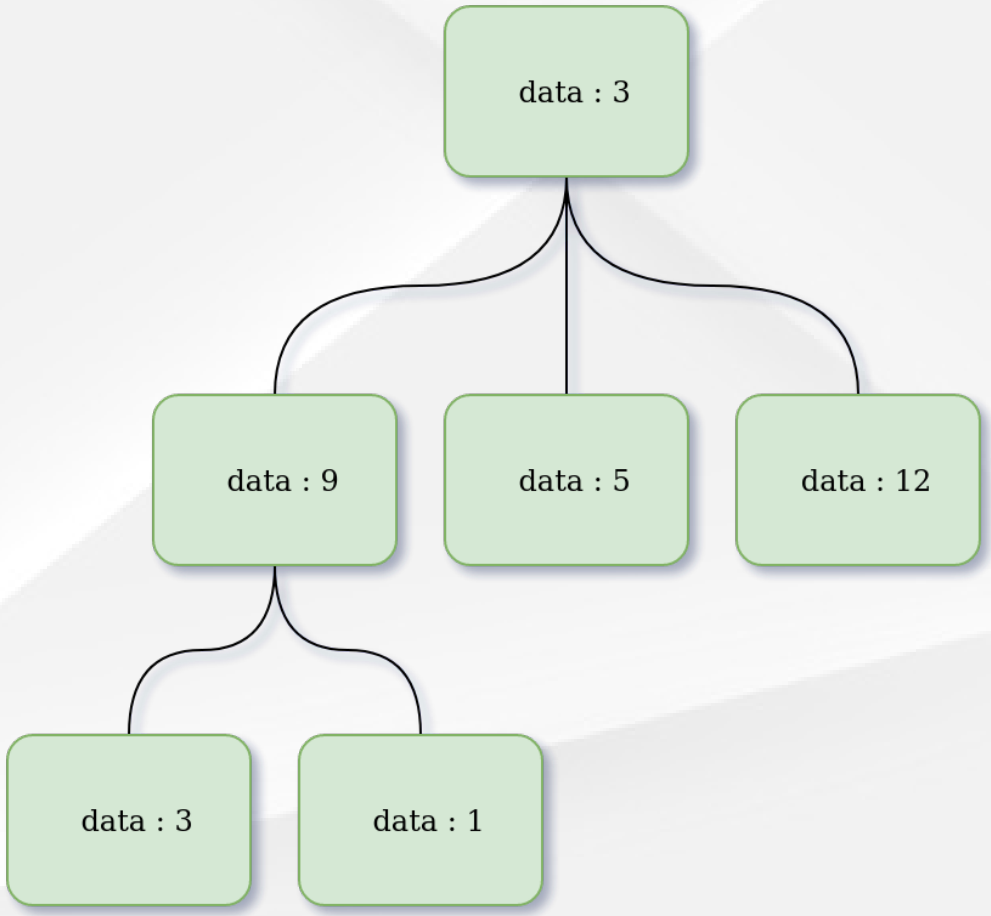
Generic Actions - Class/Theory equivalence

- *AbstractAction* $\Leftrightarrow$ *GenericAction*
  - *Action* $\Leftrightarrow a_t$
  - *DecisionRule* $\Leftrightarrow$ *GenericDecisionRule*
    - *StateBasedDR* $\Leftrightarrow p(a \mid s)$
    - *BeliefBasedDR* $\Leftrightarrow p(a \mid b)$
    - *HistoryBasedDR* $\Leftrightarrow p(a \mid h)$

Value Functions - Class Diagram



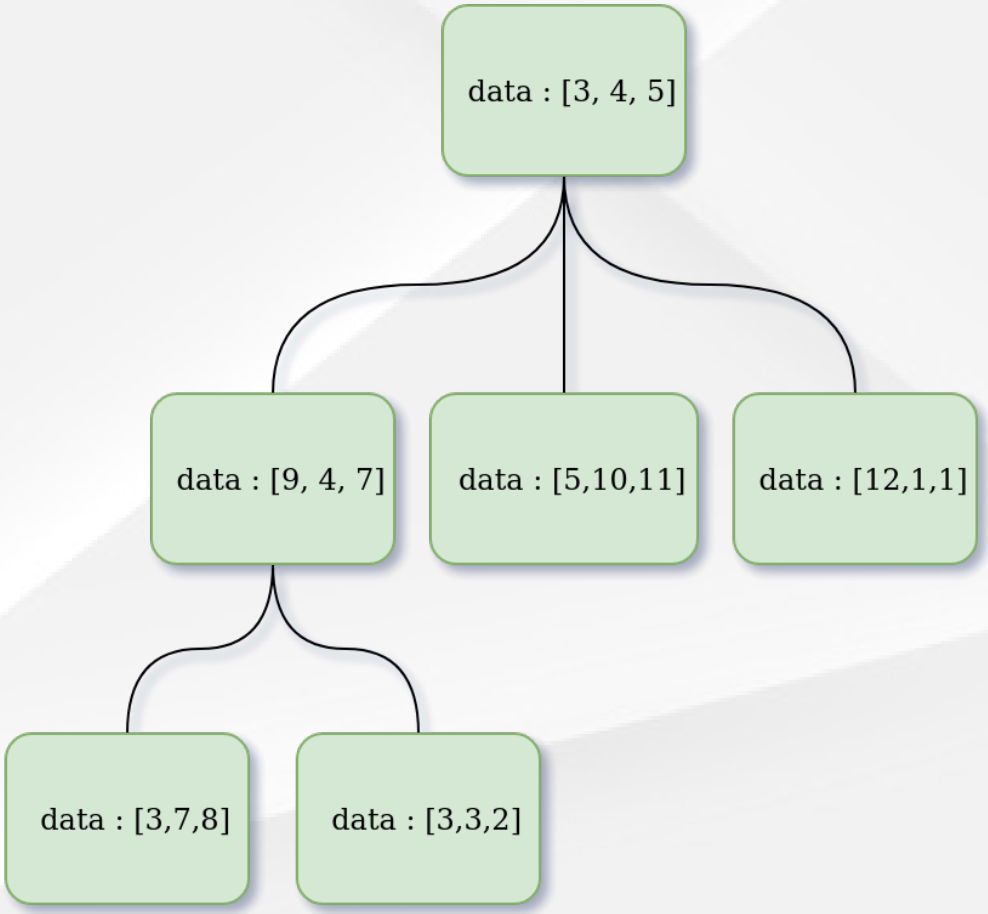
HistoryTree<int>



HistoryTree<Joint<Observation>>

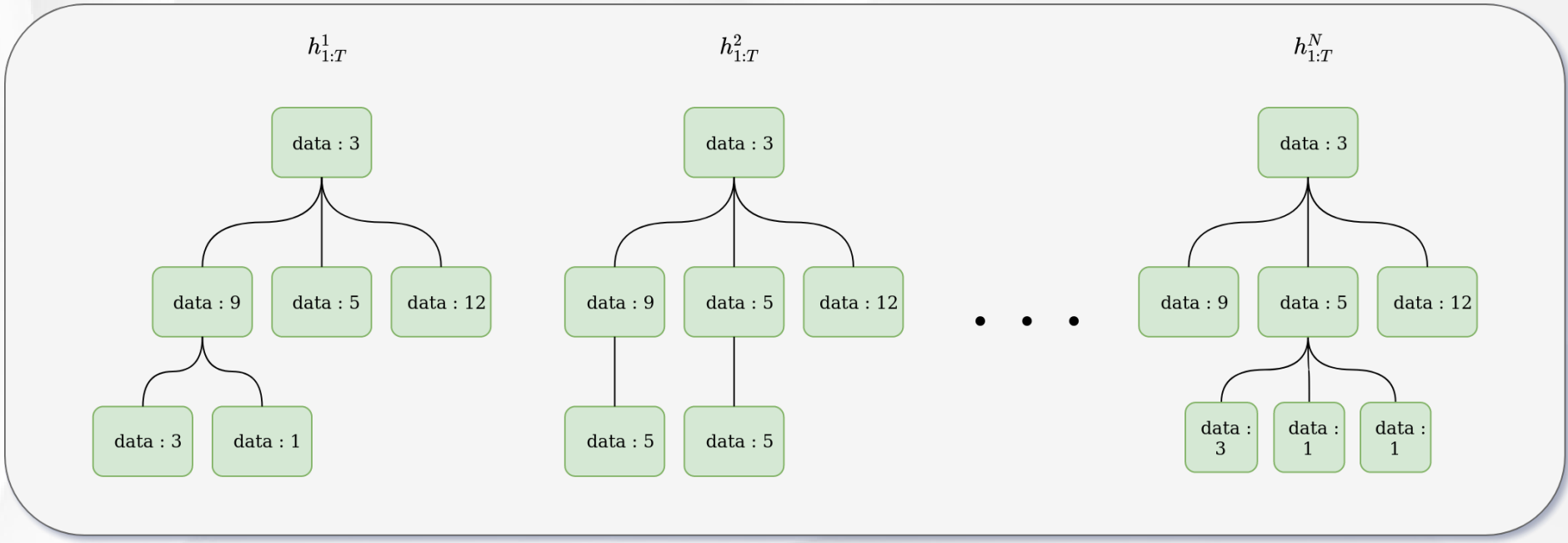
(+) Fast access to $\mathbf{o}_t = \left(o_t^1, o_t^2, .., o_t^n\right)$

(-) Slow access to $h_{1:T}^i = \left(o_1^i, o_2^i, .., o_T^i\right)$



Joint<HistoryTree<Observation>>

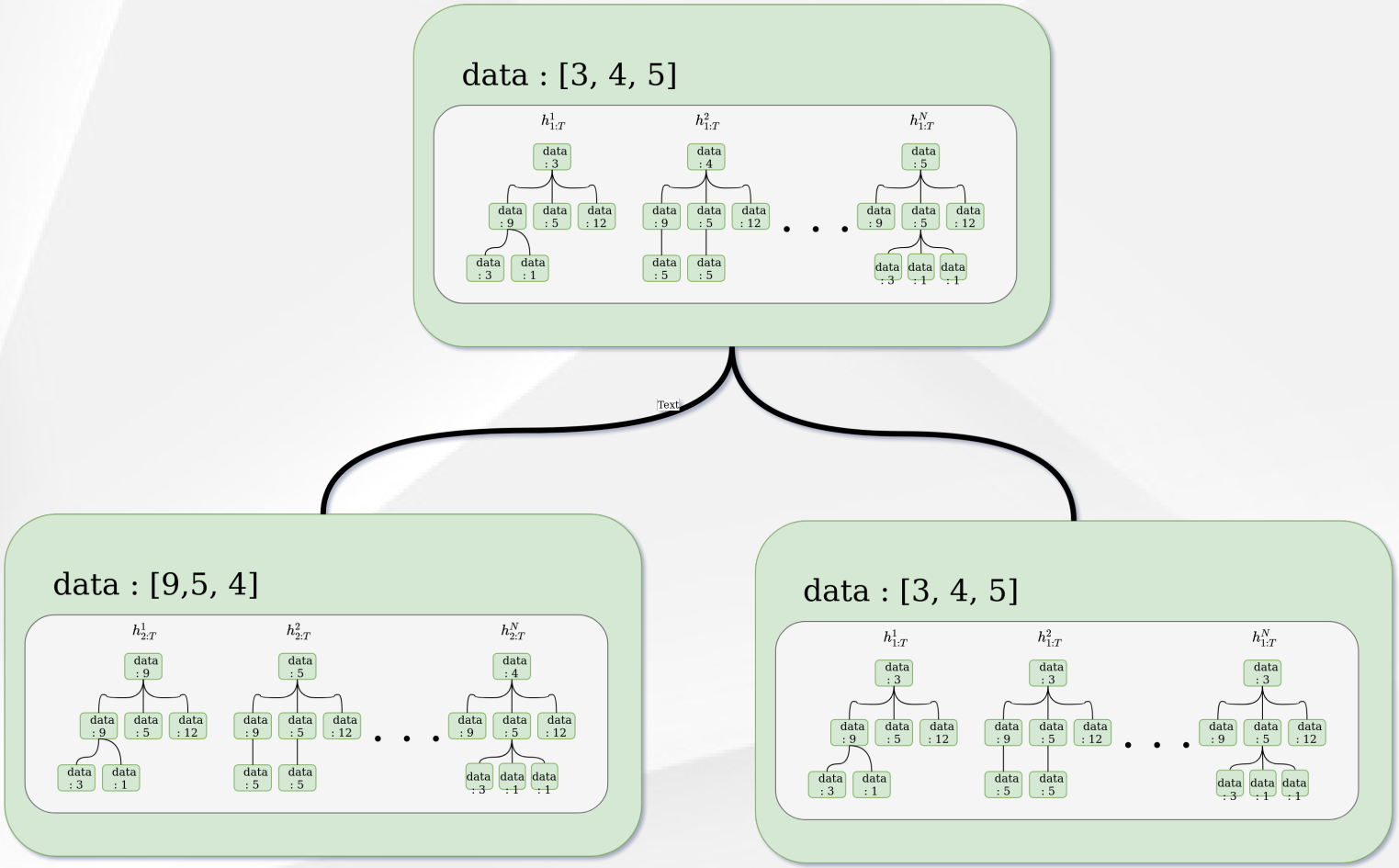
(+) Fast access to $h^i_{1:T} = \left(o^i_1, o^i_2, .., o^i_T\right)$
(-) Slow access to $\mathbf{o}_t = \left(o^1_t, o^2_t, .., o^N_t\right)$



JointHistoryTree<Observation>

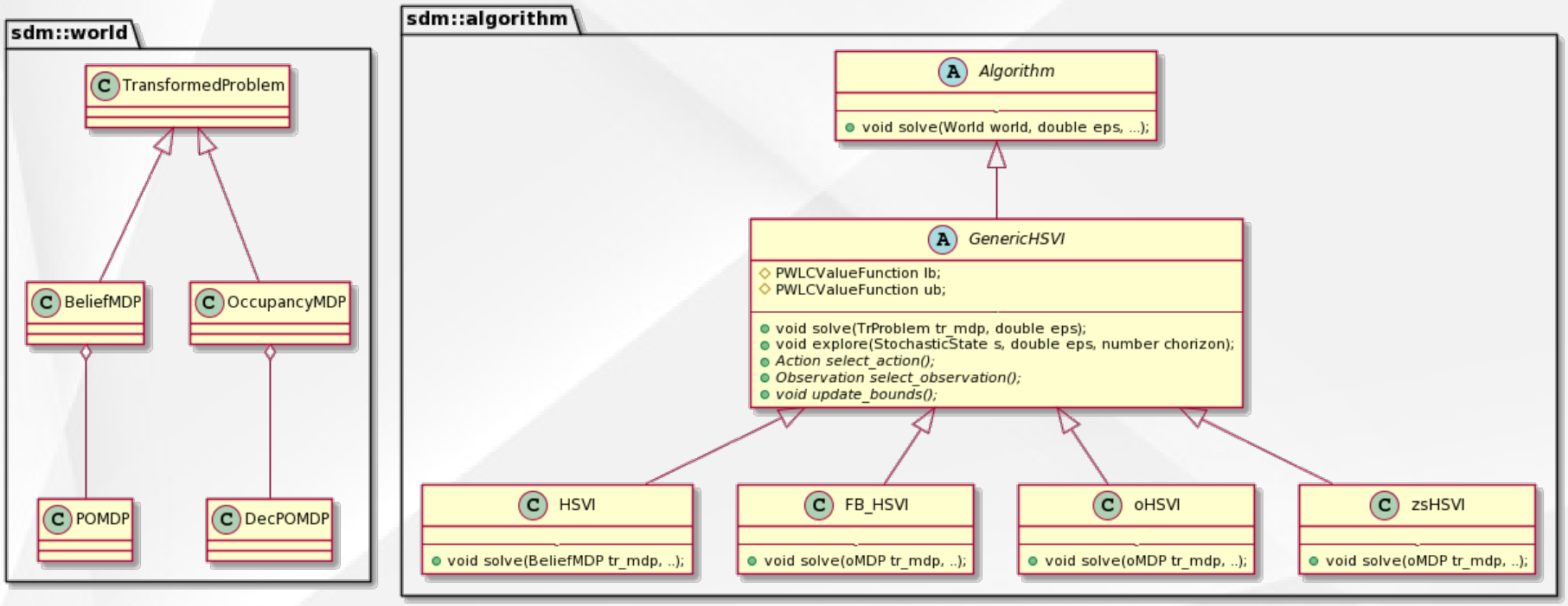
(+) Fast access to $h^i_{1:T} = \left(o^i_1, o^i_2, . . , o^i_T\right)$

(+) Fast access to $\mathbf{o}_t = \left(o^1_t, o^2_t, . . , o^N_t\right)$

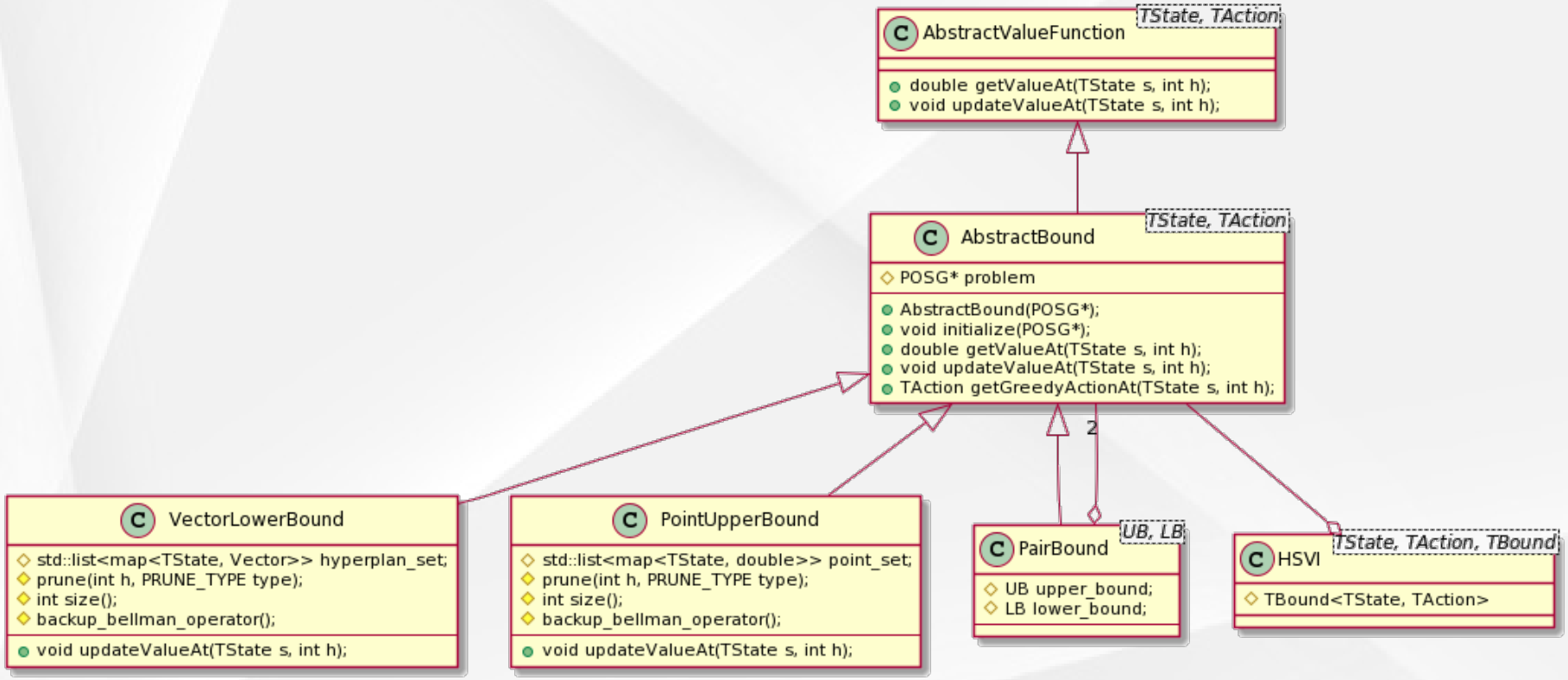


```
RecursiveMap<type1, type2, ...., typeN> rmap;
rmap.recursive_emplace(v1, v2, ..., vN);
typeN val = rmap(v1, v2, ..., vN-1);
RecursiveMap<type3, type4, ..., typeN> rmap2 = rmap(v1, v2);
std::cout « rmap « std::endl;
```

HSVI


sdm::world
TransformedProblem
BeliefMDP
OccupancyMDP
POMDP
DecPOMDP
sdm::algorithm
Algorithm
void solve(World world, double eps, ...);
GenericHSVI
PWLCValueFunction lb;
PWLCValueFunction ub;
void solve(TrProblem tr_mdp, double eps);
void explore(StochasticState s, double eps, number chorizon);
Action select_action();
Observation select_observation();
void update_bounds();
HSVI
void solve(BeliefMDP tr_mdp, ..);
FB_HSVI
void solve(oMDP tr_mdp, ..);
oHSVI
void solve(oMDP tr_mdp, ..);
zsHSVI
void solve(oMDP tr_mdp, ..);

HSVI


AbstractValueFunction
TState, TAction
double getValueAt(TState s, int h);
void updateValueAt(TState s, int h);
AbstractBound
TState, TAction
POSG* problem
AbstractBound(POSG*);
void initialize(POSG*);
double getValueAt(TState s, int h);
void updateValueAt(TState s, int h);
TAction getGreedyActionAt(TState s, int h);
2
VectorLowerBound
std::list<map<TState, Vector>> hyperplan_set;
prune(int h, PRUNE_TYPE type);
int size();
backup_bellman_operator();
void updateValueAt(TState s, int h);
PointUpperBound
std::list<map<TState, double>> point_set;
prune(int h, PRUNE_TYPE type);
int size();
backup_bellman_operator();
void updateValueAt(TState s, int h);
PairBound
UB, LB
UB upper_bound;
LB lower_bound;
HSVI
TState, TAction, TBound
TBound<TState, TAction>

Contribute to SDMS