OPERATOR PROPOGATION AND INFERENCE

ROBERT SANDERS

Operators in Complexity Zoology are implemented as partial funtions on the set of all distinct complexity classes. Processing of operators occurs after the processing of equality statements, so we can assume that we have a quotient map $q : \mathscr{N} \to \mathscr{C}$, where $\mathscr{N}$ is the set of all names for complexity classes, $\mathscr{C}$ is the set of all distinct complexity classes, and $q(x) = q(y)$ if and only if x and y are names for the same class. The system's understanding of a complexity class operator op is a partial function $op : \mathscr{C} \rightharpoonup \mathscr{C}$; i.e., a function whose domain is a subset of $\mathscr{C}$ that takes values in $\mathscr{C}$. The function that defines op is necessarily a partial one, because for a given complexity class C, the class $\mathsf{op} \cdot \mathsf{C}$ might not be declared in Complexity Zoology's data set.

Initially, we assume that the following is true of out operator partial functions:

(1) $\mathsf{id}(x) = x$ for all $x \in \mathscr{C}$.
(2) If a class x has a name of the form `op.y`, where $y\mathscr{N}$ and `op` is the name of an operator, then we set $op(q(y)) = x$.

Then, the partial functions are expanded according to the quadratic relations specified in the input file. More specifically, suppose that one such relation is

$$\mathsf{op}_1 \cdot \mathsf{op}_2 = \mathsf{op}_3 \cdot \mathsf{op}_4.$$

Then, the operator partial functions can be expanded according to the following rules:

(1) If $op_1(op_2(x))$ and $y = op_4(x)$ are defined, then define $op_3(y) := op_1(op_2(x))$.
(2) If $op_3(op_4(x))$ and $y = op_2(x)$ are defined, then define $op_1(y) := op_3(op_4(x))$.

These rules are applied iteratively until the partial functions can be expanded no further. This process uses the same task list-based system that is used in the primary inference engine.

While the partial functions are expanding, there is the possibility that a new definition of $op(x)$ contradicts an earlier one. For example, suppose it is established that $op(x) = y$, and then Complexity Zoology learns that $op(x) = z$, where $y \neq z$. This would imply that y and z, which were thought to indicate separate complexity classes, are actually two names for the same class. If this occurs, Complexity Zoology stops and prints an error, as it is expected that all equalities between complexity classes are learned through explicit statements in the data file and the transitivity of equality.