INPUT FILES AND SYNTAX

ROBERT SANDERS

Complexity Zoology reads its initial data from two plain text files: one consisting of complexity classes and their inclusions and oracle separations (`classes.txt`), and another consisting of complexity class operators and their relations (`operators.txt`). Both classes and operators must be declared in their respective files before they can be used. If an undeclared class or operator is used in some inclusion, separation, or relation, Complexity Zoology will halt and print an error. A declaration consists of a line of text having the following form:

```
NAME : description :  keyword1, keyword2, keyword3
```

Here, `NAME` is the name by which the class or operator is referenced both internally and in the output. Any alphanumeric characters, as well as hyphens, can be used for names. The `description` is a short phrase used to indicate the nature of the class or operator to a human reader; the program itself does not use the description. Like names, descriptions should consist of alphanumeric characters and hyphens, although whitespace is also allowed. Finally, keywords can optionally be included in a declaration. Keywords follow the same naming rules as class and operator names, and multiple keywords must be separated by commas, which can be surrounded by any amount of whitespace. If there are no keywords, the second colon must be omitted.

Keywords are used to provide additional information about the class or operator being declared. In general, a keyword is shorthand for commonly arising relations. For example, the class keyword `symmetric` is equivalent to including the line `C = co.C`, where `C` is the name of the declared class. The following keywords are defined for complexity classes:

- `hidden` – The class is suppressed in the final output, but it is still included for the purposes of calculation and deduction.
- `ignore` – The class is not included in calculation or output; any relations involving the class are effectively commented out.
- `preferred` – If this class is equal to another, this class should be the preferred name.
- `preferred[#]` – Here, the symbol `#` should be replaced with a positive integer and indicates the *preference rank* of the declared class. When Complexity Zoology chooses a name for equal complexity classes, it favors those with the smaller preference rank. The `preferred` keyword is equivalent to `preferred[1]`.
- `symmetric` – The class is *symmetric* in the sense of being equivalent to its complement: for a complexity class C, this means that $\mathsf{C} = \mathsf{co} \cdot \mathsf{C}$.

For operators, there is currently only one keyword:

- `idempotent` – Applying the operator to a class a second time has the same effect as applying it once. This keyword is equivalent to the relation `op.op = op`.

Aside from class and operator declarations, the input files also include *relations* describing what is (initially) known about the classes and operators. For complexity classes, relations are either statements of equality, statements of inclusion, or statements of oracle separation.

Suppose that C_1 and C_2 are classes declared with the names `C1` and `C2`, respectively. Then we have these valid relations:

- `C1 = C2` – $\mathsf{C}_1^A = \mathsf{C}_2^A$ for every oracle A.
- `C1 < C2` – $\mathsf{C}_1^A \subseteq \mathsf{C}_2^A$ for every oracle A.
- `C1 r< C2` – $\mathsf{C}_1^A \subseteq \mathsf{C}_2^A$ with probability 1 for a random oracle A.
- `C1 a< C2` – $\mathsf{C}_1^A \subseteq \mathsf{C}_2^A$ for every algebraic oracle A.
- `C1 t< C2` – $\mathsf{C}_1 \subseteq \mathsf{C}_2$ relative to the trivial oracle.
- `C1 x< C2` – $\mathsf{C}_1^A \subseteq \mathsf{C}_2^A$ for some algebraic oracle A.
- `C1 o< C2` – $\mathsf{C}_1^A \subseteq \mathsf{C}_2^A$ for some oracle A.
- `C1 osep C2` – $\mathsf{C}_1^A \not\subseteq \mathsf{C}_2^A$ for some oracle A.
- `C1 rsep C2` – $\mathsf{C}_1^A \not\subseteq \mathsf{C}_2^A$ with probability 1 for a random oracle A.
- `C1 xsep C2` – $\mathsf{C}_1^A \not\subseteq \mathsf{C}_2^A$ for some algebraic oracle A.
- `C1 tsep C2` – $\mathsf{C}_1 \not\subseteq \mathsf{C}_2$ relative to the trivial oracle.
- `C1 asep C2` – $\mathsf{C}_1^A \not\subseteq \mathsf{C}_2^A$ for every algebraic oracle A.
- `C1 sep C2` – $\mathsf{C}_1^A \not\subseteq \mathsf{C}_2^A$ for every oracle A.

For operators op_1, op_2, op_3 and op_4 with declared names `op1`, `op1`, `op1`, and `op1`, respectively, we have these relations:

- `op1.op2 = op3.op4` – $\mathsf{op}_1 \cdot \mathsf{op}_2 = \mathsf{op}_3 \cdot \mathsf{op}_4$. Omitting one of the operators on either side of the equation is allowed; e.g., `op1.op2 = op3`, which is interpeted as $\mathsf{op}_1 \cdot \mathsf{op}_2 = \mathsf{op}_3$. Complexity Zoology implements this by replacing the missing operator with the identity operator id.
- `op1 z= op2` – op_1 and op_2 commute. This is equivalent to including the line `op1.op2 = op2.op1`.
- `op1 p= op2` – op_2 absorbs op_1 on the left and right. This is equivalent to including the lines `op1.op2 = op2` and `op2.op1 = op2`.
- `op1 < op2` – $\mathsf{op}_1 \leq \mathsf{op}_2$; i.e., $\mathsf{op}_1 \cdot \mathsf{C} \subseteq \mathsf{op}_2 \cdot \mathsf{C}$ for every complexity class C.

For all relations, with the exception of the quadratic operator relations of the form `op1.op2 = op3.op4`, it is possible to include multiple classes or operators separated by commas:

```
C1,C2 = C3,C4,C5
```

This example is equivalent to including six lines: `C1 = C3`, `C1 = C4`, `C1 = C5`, `C2 = C3`, `C2 = C4`, and `C2 = C5`.

Lastly, there are two ways to include text that Complexity Zoology will ignore: *comments* and *citations*. Comments consist of the character `#` followed by all text up to the end of the current line. Citations consist of text surrounded by square brackets. In the current version of the input files, citations are used both to refer to this documentation's bibliography and to annotate common arguments.