Syntax: Object Algebras

April 2, 2015

1 Object Algebra Interface

1.1 Inheritance $\times$

1.1.1 Template

`BEFORE:  ` **`sig`** $N_{AI}[\overline{T_{AI}}]$ **`where`** $\overline{N_{CS} : T_{CS}}$`;`

`AFTER:   ` **`type`** $N_{AI}[\overline{T_{AI}}]$ `=` $\{\overline{N_{CS} : T_{CS}}\}$`;`

1.1.2 Example: **`ExpAlg[E]`**

```
BEFORE:  sig ExpAlg[E] where
            lit : Int -> E,
            add : E -> E -> E;
```

```
AFTER:   type ExpAlg[E] = {
            lit : Int -> E,
            add : E -> E -> E
         };
```

1.2 Inheritance $\surd$

1.2.1 Template

`BEFORE:  ` **`sig`** $N_{AI}[\overline{T_{AI}}]$ **`extends`** $\overline{N_{AI_2}[\overline{T_{AI_2}}]}$ **`where`** $\overline{N_{CS} : T_{CS}}$`;`

`AFTER:   ` **`type`** $N_{AI}[\overline{T_{AI}}]$ `=` $\&(\overline{N_{AI_2}[\overline{T_{AI_2}}]})$ `&` $\{\overline{N_{CS} : T_{CS}}\}$`;`

1.2.2 Example: **`StatAlg[E, S]`**

```
BEFORE:  sig StatAlg[E, S] extends ExpAlg[E] where
            seq : S -> S -> S,
            asn : String -> E -> S;
```

```
AFTER:   type StatAlg[E, S] = (ExpAlg[E]) & {
            seq : S -> S -> S,
            asn : String -> E -> S
         };
```

2 Object Algebra

2.1 Inheritance $\times$

2.1.1 Template

BEFORE: **algebra** N_A **implements** $\overline{N_{AI}[\overline{T_A}]}$ **where** $\overline{t@(N_{CS}\ \overline{x}) = E}$;

AFTER: $[\overline{T_A}/\overline{T_{AI}}]$ **let** N_A = $\{\overline{N_{CS} = \lambda(\overline{x} : \overline{T_{CS}}).\ \{t = E\}}\}$;

2.1.2 Example: **EvalExpAlg**

```
BEFORE:  type IEval = { eval : Int };
         algebra EvalExpAlg implements ExpAlg[IEval] where
            eval@(lit x) = x,
            eval@(add x y) = x.eval + y.eval;
```

```
AFTER:   type IEval = { eval : Int };
         let EvalExpAlg = {
            lit = \(x : Int) -> { eval = x },
            add = \(x : IEval) -> \(y : IEval) -> { eval = x.eval + y.eval }
         };
```

2.2 Inheritance $\surd$

2.2.1 Template

2.2.2 Example: **PrintStatAlg**

3 Datatype

3.1 Template

3.2 Example: **List[A]**

4 Creating a Structure

4.1 Template

4.2 Example:

5 Instantiation

5.1 Template

5.2 Example: