Unstable-Languages-beta: LD-Disambiguation

The PLanCompS Project

`Unstable-Languages-beta/LD/LD-Disambiguation/LD-Disambiguation.cbs`[*]

Language “LD”

A Disambiguation

A.1 Lexical constructs

Lexis SDF

```
lexical syntax
  id = keyword {reject}

lexical restrictions
  id -/- [a-z0-9]
  int -/- [0-9]
```

Syntax SDF

```
context-free syntax
start ::= exp {prefer}
```

A.2 Call-by-value lambda-calculus

Syntax SDF

```
context-free syntax
exp ::= lambda id . exp {longest-match}
exp ::= exp exp {left}
exp ::= let id = exp in exp {longest-match}

context-free priorities
exp ::= exp exp
> {
```

[*]Suggestions for improvement: `plancomps@gmail.com`.
Issues: `https://github.com/plancomps/CBS-beta/issues`.

```
exp ::= lambda id . exp
exp ::= let id = exp in exp
}
```

A.3 Arithmetic and Boolean expressions

Syntax SDF

```
context-free syntax
exp ::= exp + exp {left}
exp ::= exp * exp {left}
exp ::= exp / exp {left}
exp ::= exp <= exp {non-assoc}
exp ::= exp && exp {right}
exp ::= if exp then exp else exp {longest-match}

context-free priorities
exp ::= exp exp
>
{left:
exp ::= exp * exp
exp ::= exp / exp
} >
exp ::= exp + exp
>
exp ::= exp <= exp
>
exp ::= exp && exp
> {
exp ::= lambda id . exp
exp ::= let id = exp in exp
}
```

A.4 References and imperatives

Syntax SDF

```
context-free syntax
exp ::= exp := exp {non-assoc}
exp ::= exp ; exp {right}
exp ::= while exp do exp {longest-match}

context-free priorities
{
exp ::= ref exp
exp ::= ! exp
}>
```

```
exp ::= exp exp

context-free priorities
exp ::= exp && exp
>
exp ::= exp := exp
> {
exp ::= lambda id . exp
exp ::= while exp do exp
} >
exp ::= exp ; exp
>
exp ::= let id = exp in exp
```

A.5 Multithreading

Syntax SDF

```
context-free priorities
{
exp ::= spawn exp
exp ::= join exp
}
>
exp ::= exp ; exp
```