- quatre symboles de terme `+`, `-`, `*` et `/` à deux arguments, dont les deux arguments sont des termes, ne liant pas de variables dans leurs arguments,
- un symbole de terme `ifz` à trois arguments, dont les trois arguments sont des termes, ne liant pas de variables dans ses arguments,
- un symbole de terme `fix` à deux arguments, dont le premier est un type et le second un terme, liant une variable de terme dans son second argument,
- un symbole de terme `let` à trois arguments, dont le premier est un type et les deux autres des termes, liant une variable de terme dans son troisième argument,
- une constante de type `nat`,
- un symbole de type `->` à deux arguments, dont les deux arguments sont des types, ne liant pas de variable dans ses arguments.

Autrement dit, la syntaxe de PCF avec des types est inductivement définie par

```
A = X
  | nat
  | A -> A
t = x
  | fun x:A -> t
  | t t
  | n
  | t + t | t - t | t * t | t / t
  | ifz t then t else t
  | fix x:A t
  | let x:A = t in t
```

5.1.2 La relation de typage

On peut maintenant définir inductivement la relation `t : A` « le terme `t` a le type `A` ». Plus précisément, comme dans le cas de l'interprétation, on définit inductivement une relation à trois places `e ⊢ t : A` où `t` est un terme qui peut contenir des variables libres et `e` un *environnement de typage* qui associe un type à chacune de ces variables. Cette définition est une définition inductive, elle ressemble donc beaucoup à la définition de la sémantique opérationnelle à grands pas de PCF. On peut imaginer que c'est la sémantique opérationnelle d'un langage qui a la même syntaxe que PCF mais où l'interprétation d'un terme produit, non sa valeur, mais son type — c'est ce que l'on appelle une *interprétation abstraite* d'un terme.

$$\frac{}{\texttt{e} \vdash \texttt{x} : \texttt{A}} \text{ si } \texttt{e} \text{ contient } \texttt{x} : \texttt{A}$$

$$\frac{\texttt{e} \vdash \texttt{u} : \texttt{A} \qquad \texttt{e} \vdash \texttt{t} : \texttt{A -> B}}{\texttt{e} \vdash \texttt{t u} : \texttt{B}}$$