Tout d'abord, en PCF, un terme demande un environnement pour s'interpréter. Outre l'accumulateur, la pile et le code, notre machine abstraite doit donc avoir un quatrième registre : *l'environnement*. Cette machine doit également avoir une instruction **Extend** `x` qui étend l'environnement en lui ajoutant la définition `x = V` où `V` est le contenu de l'accumulateur et une instruction **Search** `x` qui va chercher la valeur associée à `x` dans l'environnement et la met dans l'accumulateur.

Quand on exécute le code produit par la compilation de plusieurs applications imbriquées, l'environnement change plusieurs fois, et il faut, quand on termine une telle exécution, retrouver l'environnement initial. La machine abstraite devra donc avoir des instructions **Pushenv** et **Popenv** pour mettre le contenu du registre environnement sur le sommet de la pile et le récupérer. Ces opérations consistant à empiler et dépiler des environnements sont souvent décomposées en plusieurs opérations où l'on empile et dépile les éléments de l'environnement un à un, mais nous ne les décomposerons pas de cette manière ici.

4.3.2 Les fermetures

Ensuite, il est nécessaire, en PCF, d'avoir des fermetures comme valeurs. Outre l'instruction **Ldi** `n`, nous aurons une instruction **Mkclos**`(f,x,t)`, qui prend en argument deux variables `f` et `x` et un terme `t`, qui fabrique la fermeture $\langle$`f, x, t, e`$\rangle$ où `e` est le contenu du registre environnement et met cette fermeture dans l'accumulateur.

4.3.3 Les différentes constructions de PCF

Compiler un terme de la forme `fun x -> t` ou `fixfun f x -> t` n'est pas difficile puisqu'il suffit de le compiler en l'instruction **Mkclos**`(f,x,t)` qui fabrique la fermeture qui est la valeur de ce terme.

De même, compiler un terme de la forme `x` n'est pas difficile puisqu'il suffit de le compiler en l'instruction **Search** `x` qui va chercher la valeur correspondant à `x` dans l'environnement.

Voyons maintenant comment compiler un terme de la forme `t u`. Nous avons vu que la règle définissant la sémantique opérationnelle à grands pas des termes de la forme `t u` est

$$\frac{\begin{array}{c}\mathtt{e} \vdash \mathtt{u} \hookrightarrow \mathtt{W} \\ \mathtt{e} \vdash \mathtt{t} \hookrightarrow \langle \mathtt{f},\ \mathtt{x},\ \mathtt{t'},\ \mathtt{e'} \rangle \\ (\mathtt{e'},\ \mathtt{f} = \langle \mathtt{f},\ \mathtt{x},\ \mathtt{t'},\ \mathtt{e'} \rangle,\ \mathtt{x} = \mathtt{W}) \vdash \mathtt{t'} \hookrightarrow \mathtt{V}\end{array}}{\mathtt{e} \vdash \mathtt{t}\ \mathtt{u} \hookrightarrow \mathtt{V}}$$

Pour interpréter le terme `t u` dans l'environnement `e`, on commence par interpréter le terme `u` dans l'environnement `e`, ce qui donne la valeur `W`. On interprète ensuite le terme `t` dans l'environnement `e`, ce qui donne la fermeture $\langle$`f, x, t', e'`$\rangle$, on interprète enfin le terme `t'` dans l'environnement `(e', f = ⟨f, x, t', e'⟩), x = W`, ce qui donne le résultat.