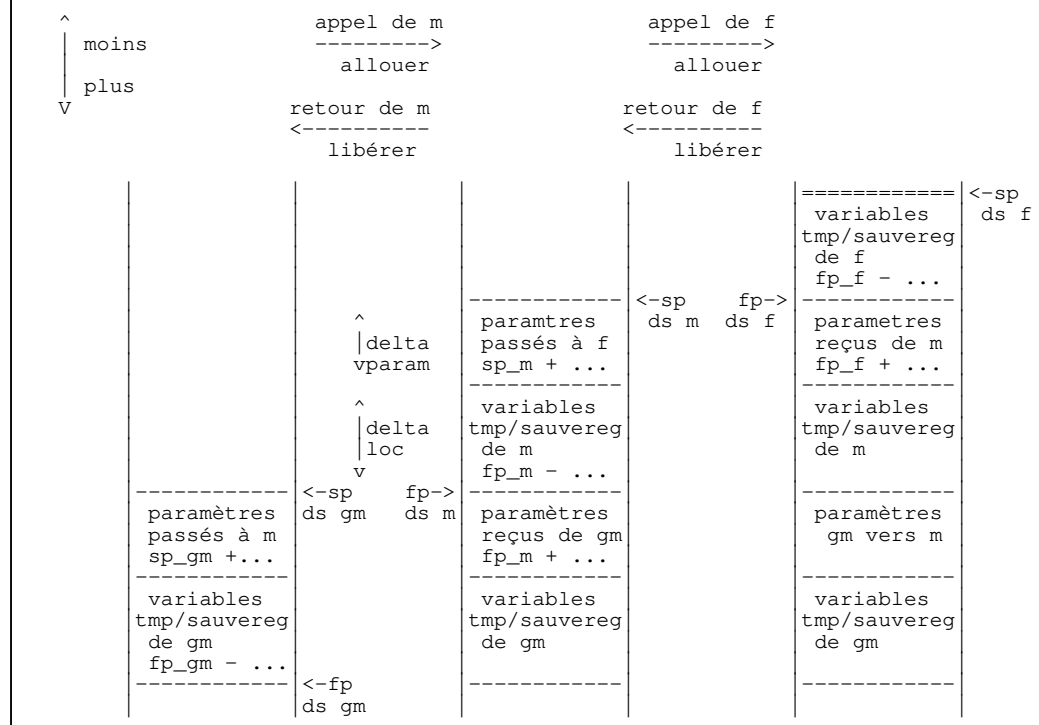
III) Stockage des locaux et des paramètres dans une pile

Registre sommet de pile (sp) : repère limite plain/vide
Registre pointeur de cadre (fp) : repère bloc précédent (appelante)




Remarque : réutilisation de l'espace mémoire.

Variantes de schéma de pile : fd, fa, ed, ea
  full descending, full ascending, empty desceding, empty ascending
  --> pile croit vers adresses hautes/basses
  --> sommet de pile pointe dernière case pleine/première case vide

ARM : fd  (très répandue)

a appelle b puis a appelle c : zone allouée par b réallouée à c.

IV) Squelette de codage

```
fonction:     @ prologue
              @ sauvegarder fp  (en sp - ....)
          @ fp = sp
          @ sauvegarder autres registres (en fp - ....)
          @ sp = sp - delta_loc - delta_param (0 si appel terminal)
          @ ceci réalise l'allocation de mémoire

          @ corps
          @ acces aux locaux en fp - ..., paramètres reçus en fp + ...
                @ appel de g
          @ écrire paramètres de g en sp + ...
          @ bl g

          @ épilogue
          @ restaurer tous les registres
          @ --> rétablit au passage anciens fp et sp --> libération
          @ mv pc,lr
```

Remarque : sachant que l'écart entre fp et sp est constant, on peut tout gérer uniquement avec sp, mais les deltas changent avec l'évolution de sp.

V) Optimisations

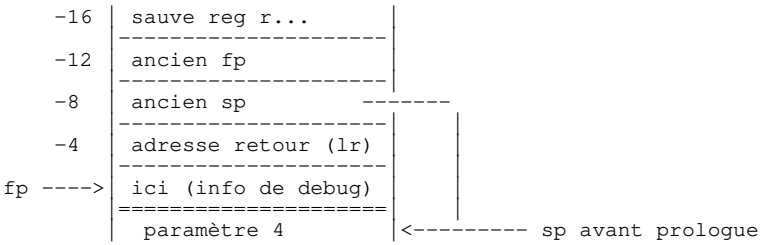
Pour économiser des accès à la pile, convention répandue :
* n premiers arguments d'appels dans les registres (ARM : 4 dans r0 à r3)  *
résultat d'une fonction stocké à la place du premier argument
* adresse de retour :
   + RISC : dans un registre (ARM :lr)/sauvée par appelée
   + CISC : empilée par instruction d'appel dans l'appelante (jsr de 68000)

VI) Code standard de gnu/ARM

  Particularité : on fait fp <- sp - 4
  Les registres sont sauvés par l'appelée, excepté ip.

  Code standard du prologue :

```
prologue:  mov   ip,sp
           stmfd ip!,{fp,ip,lr,pc}
           sub   fp,ip,#4
ici:       sub   sp, sp, #DELTA
           str   rx, [fp, #-Delta_sauve_rx]
           ...
           str   rz, [fp, #-Delta_sauve_rz]
```




  Code standard de l'épilogue

```
           str   rz, [fp, #-Delta_sauve_rz]
           ...
           str   rx, [fp, #-Delta_sauve_rx]
           ldmea fp, {fp, sp,pc}
           @ restaure anciens fp et sp, restaure lr dans pc (fait mov pc,lr)
```

  A noter : lors d'un appel, penser à sauver r0 à r3 qui contenaient les
  paramètres reçus et contiendront les paramètres passés.