mettez un tigre dans votre TI

en optimisant ses performances

Attendre que votre calculatrice programmable ait fini ses itérations n'est pas toujours une sinécure. Que peut-elle être en train de faire ? Cela va-t-il durer encore longtemps ? Dans ces cas-là, le plus simple est de résoudre le problème à la base : rendons nos programmes les plus performants possibles. Dans cette optique, nous vous proposons quelques trucs pour améliorer la rapidité d'exécution des programmes sur votre TI-58 ou votre TI-59.

Tout d'abord comparons deux programmes qui se contentent d'ajouter 1 en mémoire 00 avec un transfert au pas numéro nnn, l'un exécuté avec GTØ X (étiquette), l'autre avec GTØ nnn (adresse numérique)

Figure 1

Prog. n° 1

000	GTO
	X
...	...
...	...
NNN	LBL
	X
...	OP
...	20
...	RST

Prog. n° 2

000	GTO
...	ON
...	NN
...	...
NNN	OP
...	20
...	RST

Figure 2

nnn	Nombre d'itérations par minute Prog. n° 1	Prog n° 2
010	296	296
020	270	296
100	161	291
300	82	283
500	54	276
800	36	267

Deux programmes ayant le même objet peuvent s'exécuter dans des temps très différents.

Figure 3

Utilisons des instructions NOP (non opérantes) pour préparer les débranchements.

FIGURE 3.1

103	GTO
104	X
105	NOP
...	...
223	LBL
224	X
225	1
226	+
...	...

FIGURE 3.2

103	GTO
104	02
105	23
...	...
223	LBL
224	X
225	1
226	+
...	...

FIGURE 3.3

103	GTO
104	02
105	25
...	...
223	LBL
224	X
225	1
226	+
...	...

(figure 1). Le programme se lance par RST. CMS. R/S ; au bout d'une minute, on arrête le programme en appuyant sur R/S et on lit le nombre de boucles effectuées en rappelant la mémoire 00.

Le tableau de la figure 2 montre que plus nnn est grand, plus la différence de rapidité entre le programme numéro 1 et le numéro 2 est grande (le rapport variant entre 1 et 7). Explication : la TI-59 ne mémorise pas les adresses des labels et, lorsqu'elle rencontre GTØ X, la machine parcourt le programme pas à pas pour trouver la première instruction LBL X (s'il y en a une autre, elle sera ignorée). On comprend alors l'intérêt des GTØ adresse numérique pour les programmes très longs (ou... qui comportent beaucoup d'itérations...) que l'on désire rendre performants.

Bien sûr, la mise au point d'un tel programme est plus difficile car, dès que l'on ajoute ou supprime un pas, le numéro des instructions change.

Méthode simplifiée

Si le programme comporte un sous-programme qu'il exécute un grand nombre de fois, il suffit de placer ce sous-programme en tête : la recherche du LBL X sera alors immédiatement achevée et le transfert très rapide.

Méthode générale

On a intérêt, lors de la mise au point du programme, à n'utiliser que des GTØ X (étiquette) puis, lorsque le programme est au point, à les changer en GTØ nnn (adresse numérique). Mais cela n'est pas si simple car GTØ X prend deux pas tandis que GTØ nnn en prend trois. On ne peut donc pas relever à l'avance le numéro des instructions à adresser. Toutefois on peut éviter ce problème en mettant systématiquement l'instruction NOP lors de la mise au point du programme ; ainsi au lieu d'écrire GTØ X, on écrira GTØ X NOP ; au lieu de DSZ 1 ÷ DSZ 1 ÷ NOP, etc.

Cela pour permettre, une fois le programme au point, de placer les adresses numériques *(figure 3. 2)*.

Pour obtenir la figure 3.2 à partir de la figure 3.1, on pourra faire GTØ 103 LRN puis taper GTØ 223, mais aussi directement GTØ 104 LRN et 2 Lnx, car 23 est le code de la touche LnX. Ceci n'est d'ailleurs pas toujours possible, par exemple pour GTØ 131, car 31 est le code de la touche LRN non programmable (directement) au clavier. Mieux, on peut passer de la figure 3.1 à la figure 3.3. En effet, il est inutile d'aller en 223, puisque LBL X n'est pas exécutable ; il est préférable d'aller en 225 pour gagner deux pas.

D'ailleurs LBL X ne sert plus à rien et si le programme ne tient pas dans la machine, il faudra gratter tous les « fonds de tiroirs » et donc supprimer tous les LBL X superflus du fait de l'adressage numérique.

L'exemple fig. 3.2 deviendra

```
103    GTO
104     02
105     23
...    ...
223      1
...      +
...    ...
```

mais, bien sûr, tous les pas ont alors changé de n°...

Mais pour ceux qui possèdent l'imprimante PC 100 A, B ou C, le problème peut se résoudre aisément ; pour les autres il faudra beaucoup de sueur...

On commence par lister le programme figure 4.1. On dessine l'organigramme (c'est-à-dire tous les transferts), puis on remplace tous les GTØ X par GTØ X 00 en insérant un pas après X. On aura intérêt à commencer par la fin du programme car ainsi les premiers pas n'auront pas changé de numéro. Pour obtenir la figure 4.2, il suffit de faire :

GTØ 45 LRN DEL DEL LRN
GTØ 34 LRN INS LRN
GTØ 10 LRN DEL DEL LRN
GTØ 2 LRN INS LRN

Puis on reliste le programme, on retrace l'organigramme à l'aide du premier listing ; il ne reste plus qu'à écrire les étiquettes comme expliqué précédemment pour le passage de la figure 3.2 à la figure 3.3.

Pour que la recherche d'un sous-programme appelable au clavier (touches A, B,... A', B'... ou SBRX, etc.) se fasse très vite, il faut avoir recours à une petite astuce. Supposons que l'on ait deux sous-programmes A et B *(figure 5.1)*.

Il suffit d'écrire au début LBL A GTØ 300, et LBL B GTØ 500 *(figure 5.2)*. Ainsi, lorsqu'on appuie sur la touche A ou B, le premier label A ou B est trouvé quasi instantanément et l'instruction GTØ 300 se charge du transfert rapide à la bonne étiquette.

Lorsque la place vient à manquer il est préférable d'écrire le programme sous la forme figure 5.3.

Cette méthode est d'ailleurs applicable à tout un programme. Toutefois si le nombre de labels est trop élevé, la place peut arriver à manquer (5 pas en plus par label utilisé) ; de plus cette méthode n'est pas vraiment aussi rapide que la méthode générale, surtout s'il y a beaucoup de labels.

L'instruction RST au lieu de GTØ 000 ne prend qu'un pas au lieu de trois. C'est un autre moyen d'obte-

Figure 4

FIGURE 4.1

```
000    GTO
...      X
...    ...
010    LBL
011      X
012     OP
013     20
014    RST
...    ...
...    ...
030    X=T
031      ÷
032    GTO
033      X
...    ...
...    ...
045    LBL
046      ÷
047      1
048      +
049      3
050      )
```

FIGURE 4.2

```
000    GTO
...      X
...     00
...    ...
011     OP
012     20
013    RST
...    ...
...    ...
029    X=T
030      ÷
031     00
032    GTO
033      X
034     00
...    ...
...    ...
046      1
047      +
048      3
049      )
```

FIGURE 4.3

```
000    GTO
...     00
...     11
...    ...
011     OP
012     20
013    RST
...    ...
...    ...
029    X=T
030     00
031     46
032    GTO
033     00
034     11
...    ...
...    ...
046      1
047      +
048      3
049      )
```

Figure 5

Sous-programmes appelés depuis le clavier.

FIGURE 5.1

```
...    ...
...    ...
300    LBL
         A
         3
       RTN
...    ...
500    LBL
         B
         5
       RTN
```

FIGURE 5.2

```
000    LBL
         A
       GTO
        03
        00
       LBL
         B
       GTO
        05
        00
...    ...
300    LBL
         A
         3
       RTN
...    ...
500    LBL
         B
         5
       RTN
```

FIGURE 5.3

```
000    LBL
         A
       GTO
        03
        00
       LBL
         B
       GTO
        05
        00
...    ...
300      3
       RTN
       ...
       ...
...    ...
500      5
       RTN
       ...
       ...
```

Figure 6

Remplaçons GT0000 par RST

```
000        GTO     000        GTO
            08                 08
            00                 00
...        ...     ...        ...
800         OP     800         OP
            20                 20
           RST                GTO
                               00
                               00
```

267 itérations par minute — *202 itérations par minute*

Figure 7

Remplaçons 1 SUM 00 par OP 20

```
000        GTO
            08
            00
...        ...
800          1
           SUM
            00
           RST
```

244 itérations par minute

Figure 8

Néanmoins, si vous trouvez que votre calculatrice va vraiment trop vite, nous vous offrons le programme le plus lent possible

```
000        LBL
             A
           GTO
             B
...        ...
800        LBL
             B
             1
           SUM
            00
           GTO
             A
```

35 itérations par minute

nir un gain de temps non négligeable *(figure 6)*.

Le remplacement de 1 SUM 00 ou (1 +/— SUM 00 ou 1 INV SUM 00) par OP 20 (OP 30) donne aussi un gain de temps *(figure 7)*.

Sur la figure 8 est présenté le programme le plus lent possible : 35 itérations seulement, soit 7,6 fois plus lent que le programme figure 6.

Figure 9

Pour calculer P(x(= $x^3 + 3x^2 + 4x + 1$ pour x = 22, faites ces bons choix...

FIG 9.1	FIG 9.2	FIG 9.3	FIG 9.4	FIG 9.5
RCL	RCL	RCL	RCL	RCL
0	0	0	0	0
Y^x	x	+	x	↑
3	X^2	3	(	↑
+	+	)	CE	↑
3	3	x	x	3
x	x	RCL	(	+
RCL	RCL	0	CE	x
0	0	+	+	4
X^2	X^2	4	3	+
+	+	)	)	x
4	4	x	+	1
x	x	RCL	4	+
RCL	RCL	0	)	
0	0	+	+	
+	+	1	1	
1	1	=	=	
=	=			

Figure 10

et vous obtiendrez les gains de temps suivants

NUMERO DE PROG.	Nombre d'itérations par minute
1	54
2	61
3	66
4	73
5	126

Il faut aussi éviter de rappeler et de stocker des nombres en mémoire sans raison. La touche CE permet d'éviter dans certains cas ces rappels inutiles.

Prenons un exemple : calcul pour x = 22 du polynôme du troisième degré P (x) = $x^3 + 3x^2 + 4x + 1$ *(figure 9.1)*. Pour x^3 il faut éviter de calculer x↑3 car, hormis le fait que le calcul est lent, x↑3 provoque une erreur pour x négatif (sur TI-59) ; par contre x^2 se calcule instantanément grâce à la touche x^2 qui ne fait pas appel aux logarithmes.

Sur TI-59 8↑2 = 64 - 1,2 $* 10^{-10}$ alors que $8^2 = 64$.

En calculant x^3 par X * X^2 on obtient la figure 9.2.

On peut faire mieux en utilisant le schéma de Hörner qui minimise le nombre de multiplications !

P (x) = ((X + 3) * X + 4) * X + 1 *(figure 9.3.)*

On peut encore faire mieux par ma méthode (?) qui n'est autre qu'un schéma de Hörner modifié *(figure 9.4.)*.

P (x) = X * (X *(X + 3) + 4) + 1

Grâce à l'emploi de la touche CE, on n'utilise qu'un rappel de mémoire. Remarque : CE équivaut un peu au ENTER↑de chez HP sans en avoir les avantages...

Le tableau 10 permet de constater les progrès de la rapidité d'exécution suivant les programmes 1, 2, 3, 4 et 5. Le programme numéro 5 a été exécuté sur HP 41C qui semble ne pas poser de problème en ce qui concerne la rapidité des transferts, tout au moins une fois que le programme a été compressé.

Il semble d'ailleurs que la précision des calculs sur HP41C ne soit pas extraordinaire : pourquoi les arrondis de HP sont-ils définitifs ?

Exemple : 7 1/x 1/x donne 7 + 2 $* 10^{-12}$ sur TI-59 (donc 7 à l'affichage avec 10 chiffres).
mais 7 - 2 $* 10^{-9}$ sur HP41C soit 6.99999999.

En conclusion, il est fort probable que l'on puisse compléter ces quelques exemples simples par de nombreuses autres « trouvailles ».

Nous espérons simplement pour l'instant que ces trucs ont pu vous aider.

Pierre Canal