LFA
Logiciel de fichiers autodocumentés

J.M. Piriou CNRM/GMAP/MOD

4 février 1999, Version 2

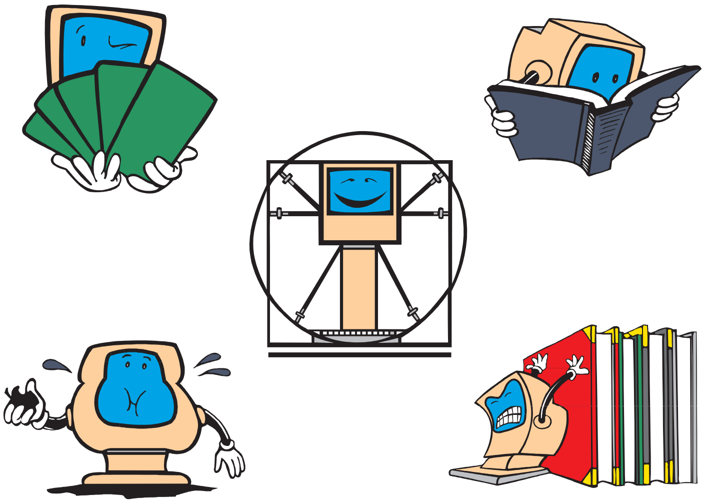


Table des matières

Chapitre 1

Présentation générale

Le présent logiciel permet de lire ou écrire des tableaux de réels, entiers ou caractères sur des fichiers portables (binaires IEEE), et avec un code portable. On accède aux champs en les référençant par leur nom.

L'idée sous-jacente à ce logiciel est de combiner deux propriétés: d'une part l'adressage direct des articles par leur nom, pour un accès plus simple et sécurisé aux données du fichier, et d'autre part l'écriture physique sur des fichiers binaires IEEE pour garantir la vitesse d'exécution et la portabilité.

L'interface usager permet de manipuler les fichiers depuis des codes fortran, mais aussi directement depuis la ligne de commande UNIX.

On peut ainsi bien sûr ouvrir/fermer des fichiers, écrire/lire des articles, mais également effectuer des copies directes d'un fichier sur l'autre, fusionner des fichiers, obtenir la liste des articles, les éléments d'un article (existence, longueur, type), créer un fichier LFA depuis la ligne de commande à partir de fichiers texte, extraire sous forme texte un article donné d'un fichier LFA, etc... Ceci permet de libérer les codes de développement de la "partie basse" de la gestion des fichiers, tout en traitant une large variété de données.

Bref historique, lien avec le logiciel LFI

Ce logiciel a été écrit en octobre 1997. Ses fonctionnalités reprennent celles du logiciel LFI (Logiciel de Fichiers Indexés) de Jean Clochard, avec une interface usager volontairement voisine, mais en y ajoutant l'autodocumentation en type (qui permet notamment la conversion et le traitement automatique de l'ensemble d'un fichier), et la possibilité de traiter des articles caractères. Les fichiers sont de plus des binaires implicites, respectant la norme IEEE.

L'interface usager est voisine pour deux raisons: d'une part, celle de LFI est suffisamment agréable et rationnelle pour qu'on n'ait pas éprouvé le besoin de la modifier, et d'autre part on minimisait ainsi le temps de portage des codes appelants.

Performances

Comment se place le logiciel LFA par rapport aux autres en termes de vitesse d'exécution et taille des fichiers?

On a mesuré le temps d'exécution et la taille des fichiers pour une écriture puis relecture d'un tableau de 150000 réels à la précision 7 chiffres significatifs (*i.e.* codés sur 4 octets), sur une station HP; les temps indiqués sont en secondes CPU (utilisateur + système), et les tailles en octets.

Logiciel	Taille	Temps
non formatté	600008	0.1
LFA	600056	0.2
LFI	605184	0.3
formatté	2550000	15.5

On voit que les formes non formattée, LFI et LFA sont comparables en taille de fichiers (150000 * 4 octets); en termes de temps calcul, l'écriture binaire (non formattée, LFA, LFI) est nettement plus favorable que la transduction en caractères (formattée).

Précision des données

Le logiciel LFA permet d'écrire/lire des réels et des entiers sur 4 ou 8 octets; l'installation du logiciel fournit les différentes bibliothèques correspondant aux précisions possibles de la machine courante. Par précision on entend ici précision des variables passées en argument par l'utilisateur au logiciel LFA. Le nom est explicite, par exemple `liblfa_R8I4.a` pour une bibliothèque destinée à être appelée par un logiciel utilisateur à réels sur 8 octets et entiers sur 4.

Cependant, le choix d'une précision utilisateur pour les passages d'arguments ne vous contraint pas à travailler à cette précision sur fichier: vous pouvez demander à écrire en 4 octets des tableaux qui ont été calculés sur 8 par exemple (afin de sauver de l'espace disque), ou lire sur des tableaux a la précision X des données présentes sur fichier à la précision Y. Le logiciel LFA fait l'interface etre les précisions fichier et celles de l'utilisateur de façon transparente.

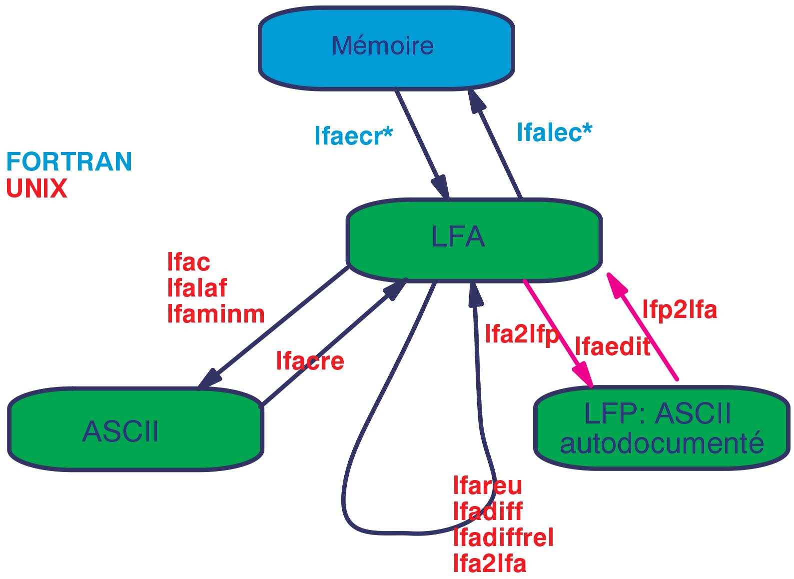
Lorsqu'on ne précise rien de particulier, le défaut est d'écrire à la précision utilisateur. Cette valeur de précision de sortie peut être changée éventuellement avant chaque article écrit, permettant ainsi d'écrire au sein d'un même fichier des données de précision différente (cf lfaprecr et lfapreci).

Portabilité fichier

Les fichiers seront a priori portables entre machines ayant la même représentation interne des données. Beaucoup de machines respectant aujourd'hui la norme IEEE, la portabilité des fichiers est en général possible.

Cas du CRAY: cette machine n'est pas IEEE mais peut produire des fichiers IEEE sur demande, via l'ordre "assign -N ieee"; cet ordre a donc été écrit dans le logiciel LFA, sous clef "#ifdef cray".

Synoptique des fonctions



Les utilitaires LFA qui font l'objet de la présente documentation ont été regroupés sur le schéma ci-dessus: l'interface entre un fichier LFA et la mémoire du calculateur où va s'exécuter votre programme est effectuée par les utilitaires fortran lfalec* (lecture) et lfaecr* (écriture).

En rouge sont représentés les utilitaires LFA appelables DIRECTEMENT depuis la ligne de commande UNIX: voir la liste des articles d'un fichier, modifier un fichier LFA à l'aide de son éditeur de texte habituel, extraire un article de fichier sur sortie standard, effectuer la différence de deux fichiers sont des opérations de routine qui ne nécessiteront donc pas d'écrire un code fortran.

Chapitre 2

Interface fortran usager

On décrit ici les routines que l'utilisateur sera amené à appeler depuis ses codes fortran pour lire, écrire ou gérer des fichiers LFA.

2.1 lfaouv: ouverture

```
subroutine lfaouv(kul,cdnomf,cdtypo)
! --------------------------------------------------------------
! **** *LFAOUV* Ouverture de fichier LFA.
! --------------------------------------------------------------
! **** *LFAOUV* Open a LFA file.
! --------------------------------------------------------------
! En entree:
! kul         unite logique du fichier.
! cdnomf      nom du fichier.
! cdtypo      type d'ouverture: 'R' READ, 'W' WRITE, 'A' APPEND, 'S' SCRATCH.
! En sortie:
! ----------------------------------------------------------------------------
! Input:
! kul         logical unit of LFA file.
! cdnomf      file name.
! cdtypo      opening type: 'R' READ, 'W' WRITE, 'A' APPEND, 'S' SCRATCH.
! Output:
! ----------------------------------------------------------------------------
```

2.2 lfafer: fermeture

```
subroutine lfafer(kul)
! --------------------------------------------------------------
! **** *LFAFER* Fermeture de fichier LFA.
! --------------------------------------------------------------
! **** *LFAFER* Close a LFA file.
! --------------------------------------------------------------
! En entree:
! kul        unite logique du fichier.
! En sortie:
```

```
! -------------------------------------------------------------------------
! Input:
! kul        logical unit of LFA file.
! Output:
! -------------------------------------------------------------------------
```

2.3 lfaprecr: précision des réels en écriture

```
subroutine lfaprecr(kul,kprec)
! -------------------------------------------------------------
! **** *LFAPRECR* Forcage de la precision d'ecriture des reels.
! -------------------------------------------------------------
! **** *LFAPRECR* Force real data writing precision.
! -------------------------------------------------------------
! En entree:
! kul      unite logique du fichier LFA.
! kprec    precision des reels a ecrire ulterieurement, en octets.
! En sortie:
! --------------------------------------------------------------
! Input:
! kul      logical unit of LFA file.
! kprec    precision of real data to write, in bytes.
! Output:
! --------------------------------------------------------------
```

2.4 lfapreci: précision des entiers en écriture

```
subroutine lfapreci(kul,kprec)
! -------------------------------------------------------------
! **** *LFAPRECI* Forcage de la precision d'ecriture des entiers.
! -------------------------------------------------------------
! **** *LFAPRECI* Force integer data writing precision.
! -------------------------------------------------------------
! En entree:
! kul      unite logique du fichier LFA.
! kprec    precision des entiers a ecrire ulterieurement, en octets.
! En sortie:
! --------------------------------------------------------------
! Input:
! kul      logical unit of LFA file.
! kprec    precision of integer data to write, in bytes.
! Output:
! --------------------------------------------------------------
```

2.5 lfaecrr: écriture de réels

```
subroutine lfaecrr(kul,cdna,preel,klong)
! -------------------------------------------------------------
```

```
! **** *LFAECRR* Ecriture de reels sur fichier LFA.
! --------------------------------------------------------------
! **** *LFAECRR* Write real data on LFA file.
! --------------------------------------------------------------
! En entree:
! kul              unite logique du fichier.
! cdna             nom de l'article a ecrire.
! preel(1,klong)   reels a ecrire.
! klong            longueur de l'article a ecrire.
! En sortie:
! --------------------------------------------------------------------------
! Input:
! kul              logical unit of LFA file.
! cdna             name of article to write.
! preel(1,klong)   real data to write.
! klong            length of article to write.
! Output:
! --------------------------------------------------------------------------
```

2.6 lfaecri: écriture d'entiers

```
subroutine lfaecri(kul,cdna,kentier,klong)
! --------------------------------------------------------------
! **** *LFAECRI* Ecriture d'entiers sur fichier LFA.
! --------------------------------------------------------------
! **** *LFAECRI* Write integer data of LFA file.
! --------------------------------------------------------------
! En entree:
! kul                  unite logique du fichier.
! cdna                  nom de l'article a ecrire.
! kentier(1,klong)      entiers a ecrire.
! klong            longueur de l'article a ecrire.
! En sortie:
! --------------------------------------------------------------------------
! Input:
! kul                  logical unit of LFA file.
! cdna                 name of article to write.
! kentier(1,klong)     integers to write.
! klong                length of article to write.
! Output:
! --------------------------------------------------------------------------
```

2.7 lfaecrc: écriture de caractères

```
subroutine lfaecrc(kul,cdna,cdcar,klong)
! --------------------------------------------------------------
! **** *LFAECRC* Ecriture de caracteres sur fichier LFA.
! --------------------------------------------------------------
```

```
! **** *LFAECRC* Write character data on LFA file.
! -----------------------------------------------------------------
! En entree:
! kul              unite logique du fichier.
! cdna             nom de l'article a ecrire.
! cdcar(1,klong)   caracteres a ecrire.
! klong            longueur de l'article a ecrire.
! En sortie:
! --------------------------------------------------------------------------------
! Input:
! kul              logical unit of LFA file.
! cdna             name of article to write.
! cdcar(1,klong)   characters to write.
! klong            length of article to write.
! Output:
! --------------------------------------------------------------------------------
```

2.8 lfalecr: lecture de réels

```
subroutine lfalecr(kul,cdna,kdimb,preel,klong,kerr)
! -----------------------------------------------------------------
! **** *LFALECR* Lecture de reels sur fichier LFA.
! -----------------------------------------------------------------
! **** *LFALECR* Read real data on LFA file.
! -----------------------------------------------------------------
! En entree:
! kul              unite logique du fichier.
! cdna             nom de l'article.
! kdimb            dimension du tableau preel.
! En sortie:
! klong            nombre de reels lus.
! preel(1,klong)   reels lus.
! kerr             indicateur d'erreur:
! +----------+-----------------------------------------------------+
! | Valeur   |             Signification                           |
! +----------+-----------------------------------------------------+
! | kerr=  0 | Tout est OK!                                        |
! | kerr= -1 | Article inexistant                                  |
! | kerr= -6 | Article plus long que le tableau devant le recevoir |
! | kerr= -8 | Mauvais type de donnees (reelles, entieres, car.)   |
! +----------+-----------------------------------------------------+
! --------------------------------------------------------------------------------
! Input:
! kul              logical unit of LFA file.
! cdna             article name.
! kdimb            physical dimension of array preel.
! Output:
! klong            number of real elements read.
! preel(1,klong)   real elements read.
```

```
! kerr             error indicator:
! +----------+----------------------------------------------------+
! | Value    |             Meaning                                |
! +----------+----------------------------------------------------+
! | kerr=  0 | Everything is OK!                                  |
! | kerr= -1 | Article inexistant                                 |
! | kerr= -6 | Article bigger than array supposed to receive it   |
! | kerr= -8 | Wrong data type (real, integer, char.)             |
! +----------+----------------------------------------------------+
! ---------------------------------------------------------------------------
```

2.9 lfaleci: lecture d'entiers

```
subroutine lfaleci(kul,cdna,kdimb,kentier,klong,kerr)
! --------------------------------------------------------------
! **** *LFALECI* Lecture d'entiers sur fichier LFA.
! --------------------------------------------------------------
! **** *LFALECI* Read integer data on LFA file.
! --------------------------------------------------------------
! En entree:
! kul              unite logique du fichier.
! cdna             nom de l'article.
! kdimb            dimension du tableau kentier.
! En sortie:
! klong            nombre d'entiers lus.
! kentier(1,klong) entiers lus.
! kerr             indicateur d'erreur:
! +----------+----------------------------------------------------+
! | Valeur   |             Signification                          |
! +----------+----------------------------------------------------+
! | kerr=  0 | Tout est OK!                                       |
! | kerr= -1 | Article inexistant                                 |
! | kerr= -6 | Article plus long que le tableau devant le recevoir |
! | kerr= -8 | Mauvais type de donnees (reelles, entieres, car.)  |
! +----------+----------------------------------------------------+
! ---------------------------------------------------------------------------
! Input:
! kul              logical unit of LFA file.
! cdna             article name.
! kdimb            physical dimension of array kentier.
! Output:
! klong            number of integer elements read.
! kentier(1,klong) integer elements read.
! kerr             error indicator:
! +----------+----------------------------------------------------+
! | Value    |             Meaning                                |
! +----------+----------------------------------------------------+
! | kerr=  0 | Everything is OK!                                  |
! | kerr= -1 | Article inexistant                                 |
```

```
! | kerr= -6 | Article bigger than array supposed to receive it    |
! | kerr= -8 | Wrong data type (real, integer, char.)              |
! +----------+-----------------------------------------------------+
! -----------------------------------------------------------------------------
```

2.10 lfalecc: lecture de caractères

```
subroutine lfalecc(kul,cdna,kdimb,cdcar,klong,kerr)
! -------------------------------------------------------------
! **** *LFALECC* Lecture de caracteres sur fichier LFA.
! -------------------------------------------------------------
! **** *LFALECC* Read character data on LFA file.
! -------------------------------------------------------------
! En entree:
! kul              unite logique du fichier.
! cdna             nom de l'article.
! kdimb            dimension du tableau cdcar.
! En sortie:
! klong            nombre de chaines de caracteres lues.
! cdcar(1,klong)   chaines lues.
! kerr             indicateur d'erreur:
! +----------+-----------------------------------------------------+
! | Valeur   |             Signification                           |
! +----------+-----------------------------------------------------+
! | kerr=  0 | Tout est OK!                                        |
! | kerr= -1 | Article inexistant                                  |
! | kerr= -6 | Article plus long que le tableau devant le recevoir |
! | kerr= -8 | Mauvais type de donnees (reelles, entieres, car.)   |
! +----------+-----------------------------------------------------+
! -----------------------------------------------------------------------------
! Input:
! kul              logical unit of LFA file.
! cdna             article name.
! kdimb            physical dimension of array cdcar.
! Output:
! klong            number of character elements read.
! cdcar(1,klong)   character elements read.
! kerr             error indicator:
! +----------+-----------------------------------------------------+
! | Value    |             Meaning                                 |
! +----------+-----------------------------------------------------+
! | kerr=  0 | Everything is OK!                                   |
! | kerr= -1 | Article inexistant                                  |
! | kerr= -6 | Article bigger than array supposed to receive it    |
! | kerr= -8 | Wrong data type (real, integer, char.)              |
! +----------+-----------------------------------------------------+
! -----------------------------------------------------------------------------
```

2.11 lfatest: le fichier est-il LFA?

```
subroutine lfatest(kul,cdnomf,ldlfa)
! -------------------------------------------------------------
! **** *LFATEST* Teste si un fichier est bien de type LFA.
! -------------------------------------------------------------
! **** *LFATEST* Test if a file is a LFA one.
! -------------------------------------------------------------
! En entree:
! kul         unite logique du fichier;
! .           ce doit etre une unite disponible:
! .           le fichier va etre ouvert sous cette unite logique.
! cdnomf      nom du fichier.
! En sortie:
! ldlfa=.true. si le fichier est de type LFA, .false. sinon.
! --------------------------------------------------------------------------
! Input:
! kul         logical unit of file.
! .           this unit has to be free:
! .           the file will be opened with this logical unit.
! cdnomf      file name.
! Output:
! ldlfa=.true. if the file is a LFA one, .false. else case.
! --------------------------------------------------------------------------
```

2.12 lfames: niveau de messagerie

```
subroutine lfames(kul,kmes)
! -------------------------------------------------------------
! **** *LFAMES* Niveau de messagerie du logiciel LFA.
! -------------------------------------------------------------
! **** *LFAMES* Print out level of LFA software.
! -------------------------------------------------------------
! En entree:
! kul         unite logique du fichier.
! kmes        niveau de messagerie:
! si 0 aucun message sorti par le logiciel LFA.
! si 1 messages d'ATTENTION et d'ERREUR sorties.
! si 2 LFA est bavard (a reserver au debug de LFA...).
! En sortie:
! --------------------------------------------------------------------------
! Input:
! kul         logical unit of LFA file.
! kmes        print out level:
! if 0 no message print out.
! if 1 WARNING or ERROR messages print out.
! if 2 many comments print out (LFA debug mode only...).
! Output:
! --------------------------------------------------------------------------
```

2.13 lfaerf: niveau d'erreur admis

```
subroutine lfaerf(kul,lderf)
! -------------------------------------------------------------
! **** *LFAERF* Niveau d'erreur tolere par le logiciel LFA.
! -------------------------------------------------------------
! **** *LFAERF* Choose error level of LFA software.
! -------------------------------------------------------------
! En entree:
! kul               unite logique du fichier.
! lderf             .true. si toute erreur doit etre fatale,
! .false. si aucune ne doit l'etre.
! En sortie:
! lgerf             .true. si toute erreur est fatale,
! .false. si aucune ne l'est.
! --------------------------------------------------------------------------
! Input:
! kul               logical unit of LFA file.
! lderf             .true. if any error has to be fatal.
! .false. si none has to be.
! Output:
! lgerf             .true. if any error has to be fatal.
! .false. si none has to be.
! --------------------------------------------------------------------------
```

2.14 lfalaf: liste des articles sur output standard

```
subroutine lfalaf(kul,kulout)
! -------------------------------------------------------------
! **** *LFALAF* Liste des articles d'un fichier LFA.
! -------------------------------------------------------------
! **** *LFALAF* Article list of a LFA file.
! -------------------------------------------------------------
! En entree:
! kul             unite logique du fichier.
! kulout          unite logique sur laquelle sortir la liste.
! En sortie:
! --------------------------------------------------------------------------
! Input:
! kul             logical unit of LFA file.
! kulout          logical unit on which print out the list.
! Output:
! --------------------------------------------------------------------------
```

2.15 lfalaft: liste des articles sur tableau de caractères

```
subroutine lfalaft(kul,cdlis,kdlis,knlis)
! -------------------------------------------------------------
! **** *LFALAFT* Liste des articles d'un fichier LFA sur tableau de caracteres.
```

```
! ------------------------------------------------------------
! **** *LFALAFT* Article list of a LFA file, on an array.
! ------------------------------------------------------------
! En entree:
! kul             unite logique du fichier.
! kdlis           dimension physique du tableau cdlis.
! En sortie:
! knlis           nombre d'articles du fichier.
!                 Ce nombre est egalement le nombre d'elements ecrits sur cdlis
! cdlis(1, ..., knlis) nom des articles du fichier.
! --------------------------------------------------------------------------
! Input:
! kul            logical unit of LFA file.
! kdlis          physical dimension of array cdlis.
! Output:
! knlis          number of articles on the file.
!                This number is also the number of elements written on cdlis.
! cdlis(1, ..., knlis) article names.
! --------------------------------------------------------------------------
```

2.16 lfaminm: extrema de tous les articles

```
subroutine lfaminm(kul)
! ------------------------------------------------------------
! **** *LFAMINM* Extrema de tous les articles d'un fichier LFA.
! ------------------------------------------------------------
! **** *LFAMINM* Extrema of all articles of a given LFA file.
! ------------------------------------------------------------
! En entree:
! kul unite logique du fichier LFA d'entree.
! En sortie:
! Extrema sur output standard.
! -------------------------------------------------------------
! Input:
! kul logical unit of LFA file.
! Output:
! Extrema on standard output.
! -------------------------------------------------------------
```

2.17 lfacas: renseignements sur un article

```
subroutine lfacas(kul,cdna,cdtype,klong,kerr)
! ------------------------------------------------------------
! **** *LFACAS* Renseignements sur un article de fichier LFA.
! ------------------------------------------------------------
! **** *LFACAS* Get documentation about a LFA article.
! ------------------------------------------------------------
! En entree:
```

```
! kul               unite logique du fichier.
! cdna: si cdna=' ' on recherche l'article suivant.
! .         cdna est alors en entree/sortie,
! .         et en sortie il vaudra le nom de l'article suivant
! .         (si cet article existe).
! .         kerr...retour de recherche: 0 si OK,
! .                1 si fin de fichier.
! .     si cdna<>' ' cdna est le nom de l'article cherche.
! .          Il est alors en entree seulement.
! .         kerr...retour de recherche: 0 si OK,
! .                1 si article inexistant.
! En sortie:
! cdtype            type d'article: 'R4', 'I8', 'C '.
! klong             nombre d'elements de cet article.
! --------------------------------------------------------------------------
! Input:
! kul               file logical unit.
! cdna: if cdna=' ' on looks for nbext article.
! .         cdna is then in input/output
! .         and in output it will receive next article name
! .         (if this article exists).
! .         kerr...return from search: 0 if OK,
! .                1 if end of file.
! .     if cdna<>' ' cdna is the name from required article.
! .          It is then in input olny.
! .         kerr...return from search: 0 if OK,
! .                1 if non-existant article.
! Output:
! cdtype            article type: 'R4', 'I8', 'C '.
! klong             numbre of elements in this article.
! --------------------------------------------------------------------------
```

2.18 lfaavan: saut de l'article courant

```
subroutine lfaavan(kul)
! -------------------------------------------------------------
! **** *LFAAVAN* Saut de l'article courant dans un fichier LFA.
! -------------------------------------------------------------
! **** *LFAAVAN* Step over current article in an LFA file.
! -------------------------------------------------------------
! En entree:
! kul               unite logique du fichier.
! En sortie:
! --------------------------------------------------------------------------
! Input:
! kul               logical unit of the LFA file.
! Output:
! --------------------------------------------------------------------------
```

2.19 lfarew: rebobinage du fichier

Cet appel sert dans le cas rare suivant: vous avez lu certains articles du fichier, puis vous voulez lire tous les articles du fichier séquentiellement via lfacas. lfacas fournissant le nom de l'article suivant, il faut au préalable rebobiner le fichier par lfarew.

Ce cas est rare: en général, soit on lit des articles en y accédant directement par leur nom, auquel cas la gestion du pointeur fichier est effectuée de façon transparente par le logiciel LFA, soit on veut lire tout le fichier séquentiellement, et on le fait dès son ouverture, et il n'y a donc pas lieu de rebobiner!...

```
subroutine lfarew(kul)
! --------------------------------------------------------------
! **** *LFAREW* Rebobinage d'un fichier LFA.
! --------------------------------------------------------------
! **** *LFAREW* Rewind a LFA file.
! --------------------------------------------------------------
! En entree:
! kul: unite logique du fichier LFA.
! En sortie:
! ---------------------------------------------------------------
! Input:
! kul: logical unit of LFA file.
! En sortie:
! ---------------------------------------------------------------
```

2.20 lfacop: copie d'article de fichier à fichier

```
subroutine lfacop(kule,cdnae,cdnas,kuls)
! --------------------------------------------------------------
! **** *LFACOP* Copie d'un article d'un fichier LFA a un autre.
! --------------------------------------------------------------
! **** *LFACOP* Copy one article from a LFA file to another.
! --------------------------------------------------------------
! En entree:
! kule unite logique du fichier LFA d'entree.
! cdnae nom de l'article a lire.
! cdnas nom sous lequel l'article est recopie.
! kuls unite logique du fichier LFA de sortie.
! En sortie:
! Le fichier d'unite logique kuls est augmente d'un article.
! ---------------------------------------------------------------
! Input:
! kule logical unit of input LFA file.
! cdnae article name to be read.
! cdnas article name to be written out.
! kuls logical unit of output LFA file.
! Output:
! The file which logical unit is kuls receives one more article.
! ---------------------------------------------------------------
```

Chapitre 3

Interface UNIX usager

Les fichiers LFA pouvant contenir ds données très diverses, il est utile, lorsqu'on dispose de fichiers LFA, d'effectuer sur ces fichiers un certain nombre d'opérations simples directement depuis la ligne de commande du système d'exploitation: savoir la liste des articles qu'ils contiennent, quels sont les extrêmes de tel article, pouvoir créer un fichier "à la volée" a partir de n fichiers textes, etc... Afin que chacun ne soit pas amené à récrire ces utilitaires, leurs sources sont mis à disposition avec la bibliothèque LFA, et leurs exécutables sont générés lors de l'installation du logiciel.

Leurs descriptions synoptiques et modes d'emploi sont proposés ci-dessous. Ces élements peuvent être obtenus également depuis la ligne de commande: taper une commande LFA (par exemple: lfalaf) sans argument provoque la sortie à l'écran du synoptique de la commande.

3.1 lfalaf: liste des articles

```
 Sortie sur output standard de la liste des articles d'un fichier lfa.

 Utilisation: lfalaf nomf
```

3.2 lfaminm: sortie des extrema et moyenne

```
Sortie des extrema et moyenne des articles d'un (plusieurs) fichier(s) LFA.

Utilisation: lfaminm LFA1 [LFA2 ... LFAn]
```

3.3 lfaedit: édition sous forme texte

```
Edition d'un(plusieurs) fichier(s) LFA.
Le but est ici de pouvoir visualiser ou modifier à l'aide
de son éditeur habituel toutes les valeurs d'un fichier LFA.
```

```
Utilisation: lfaedit F1 [F2 ... Fn]

Principe: les fichiers sont transformés en leur forme LFP (ASCII texte),
puis on appelle l'éditeur. Les fichiers en sortie de l'éditeur,
s'ils ont été modifiés, sont retransformés en leur forme LFA.
L'éditeur appelé est celui défini par la variable
d'environnement EDITOR.
```

3.4 lfac: extraction d'un article

```
Extraction sur output standard d'un article de fichier lfa.

Utilisation: lfac nomf nomc
avec
    nomf nom du fichier lfa d'entree.
    nomc nom du champ a extraire.
```

3.5 lfareu: réunion de deux fichiers

```
Reunion de deux fichiers lfa.

Utilisation: lfareu F1 F2 Fres
       En entree: F1 et F2, en sortie: Fres.
       F2 est prioritaire sur F1, i.e. si un article
       est present dans F1 et F2, c'est celui de F2 qui sera copie.
```

3.6 lfamoy: moyenne de n fichiers

```
Moyenne de n fichiers LFA.

Utilisation: lfamoy FMOY F1 F2 [F3 ... Fn]
avec
       F1 F2 [F3 ... Fn] les n fichiers d'entrée.
       FMOY le fichier de sortie, recevant la moyenne.

Remarque: la moyenne est opérée sur les articles
communs aux n fichiers.
```

3.7 lfacre: création d'un fichier depuis la ligne de commande

```
Création d'un fichier lfa à partir de la ligne de commande
et(ou) de fichier(s) texte.

Utilisation:

lfacre LFA [nom_1 type_1 valfic_1] ... [nom_n type_n valfic_n]
n doit valoir au plus  20
En sortie, le fichier LFA contiendra les n articles
nom_1 à nom_n, dont le type sera type_1 à type_n (type: R, I, ou C),
et dont le contenu sera valfic_1 à valfic_n:
        - Si valfic_i est un fichier, alors le contenu
          de ce fichier sera le contenu de l'article nom_i.
        - Si valfic_i n'est pas un fichier, alors c'est la valeur
          du contenu de l'article de longueur 1 nom_i.

Exemple:

cat <<EOF > gol
gol1
gol2
EOF
lfacre LFA IO  R 1370. indice C gol i8 I 8
créera le fichier LFA, contenant trois articles, l'article réel IO
(longueur 1), l'article caractère indice (longueur 2),
et l'article entier i8 (longueur 1).
```

3.8 lfadiff: différence de deux fichiers

```
Différence de deux fichiers lfa.

Utilisation: lfadiff F1 F2 FDIFF
avec
       F1 et F2 les deux fichiers d'entrée.
       FDIFF le fichier de sortie, recevant F1-F2.

Remarque: la différence est opérée sur les articles
communs aux deux fichiers.
```

3.9 lfadiffrel: différence relative de deux fichiers

```
Différence relative de deux fichiers LFA.

Utilisation: lfadiffrel F1 F2 FDIFF
avec
```

```
       F1 et F2 les deux fichiers d'entrée.
       FDIFF le fichier de sortie, recevant (F2-F1)/rcm(F1).

rcm(F1) est la racine du carré moyen de l'article de F1.
Remarque: la différence est opérée sur les articles
communs aux deux fichiers.

Lorsque rcm(F1)=0, le résultat est nul si F2=0, et égal à  999.999
 sinon.
```

3.10 lfadiffart: différence d'articles entre deux fichiers

```
Différence entre les listes d'articles de deux fichiers LFA.

Utilisation: lfadiffart F1 F2
avec
       F1 et F2 les deux fichiers d'entrée.
```

3.11 lfa2lfp: obtention d'une version ASCII texte du fichier

```
Conversion d'un fichier LFA en un fichier LFP.

Sujet:

Les fichiers binaires ne sont lisibles que par un logiciel.
Or il serait souvent utile de naviguer directement dans les
données à l'aide d'un simple éditeur, pour voir
les valeurs individuelles, pouvoir les imprimer, etc...
Le présent convertisseur transforme un fichier LFA (binaire IEEE)
en un fichier ASCII texte comportant in extenso les noms d'articles,
leur taille, type, et tous leurs éléments. Un tel fichier
peut egalement transiter par le courrier électronique.

Utilisation: lfa2lfp nom_fic_entree nom_fic_sortie
```

3.12 lfp2lfa: passage d'une version ASCII texte à un LFA

Conversion inverse de la précédente.

```
Conversion d'un fichier LFP en un fichier LFA.

Utilisation: lfp2lfa nom_fic_entree nom_fic_sortie
```

3.13 lfa2lfa: modification de la précision des données

```
Conversion d'un fichier LFA en un autre fichier LFA,
en forçant la précision des réels et des entiers.

Utilisation: lfa2lfa [-i] [-r] nom_fic_entree nom_fic_sortie

avec
       -i8 si on veut en sortie des entiers sur 8 octets.
       -i4 si on veut en sortie des entiers sur 4 octets.
           défaut:  4
       -r8 si on veut en sortie des réels   sur 8 octets.
       -r4 si on veut en sortie des réels   sur 4 octets.
           défaut:  4

Exemple:
       lfa2lfa -r8 -i4 LFA LFARES
```

Cet utilitaire permet par exemple de gagner de la place fichier en convertissant a posteriori en 4 octets des données inutilement écrites sur 8.

Chapitre 4

Exemples d'utilisation

4.1 Simple écriture / lecture

La routine LFAPPDEMO du source lfa.F écrit sur un fichier LFA des données entières, réelles et caractères, sort la liste des articles écrits, puis relit ces données.

4.2 Lecture séquentielle de tout un fichier

Un des intérêts du logiciel LFA est que les articles sont autodocumentés en nom, taille et type de sorte qu'il est possible de lire tout un fichier sans a priori sur son contenu. Exemple avec le programme lecture_sequentielle.f qui donne les moyenne et écart-type de tous les articles réels d'un fichier LFA donné.

4.3 Lecture hors du cadre LFA

Lorsqu'on fournit de façon ponctuelle un fichier LFA à quelqu'un ne disposant pas du logiciel -et ne souhaitant pas l'installer pour lire un seul fichier!-, on joint avec le logiciel le programme lecture_directe_lfa.f, qui saute les articles d'autodocumentation pour fournir les seuls articles de données réelles et entières.

Chapitre 5

Présentation détaillée: principes de fonctionnement interne

5.1 Structure des fichiers

Les fichiers en entrée/sortie sont des fichiers séquentiels non formattés, respectant la norme IEEE; leur structure est la suivante:

1. En-tête du fichier: on écrit un entier fixe. Cet entier permet au logiciel LFA de vérifier lors de chaque ouverture en lecture d'un fichier, que celui-ci avait bien auparavant été écrit via le logiciel LFA. La valeur de cet entier permet également de voir la version du logiciel qui a écrit le fichier, afin d'assurer la compatibilité ascendante lors de futures évolutions.

   `integer*4 iversion`

2. On rencontre ensuite, autant de fois que d'articles:

   Une autodocumentation: c'est une ligne d'entiers, contenant successivement le type de la donnée (réel, entier ou caractère), sa longueur (logique, pas la longueur physique en octets), la longueur L en caractères du nom de l'article, puis la suite des L codes ASCII des caractères du nom de l'article. L'écriture fortran est du type:
   `write(kul) ktype,klong,ilna,(ichar(cdna(jna:jna)),jna=1,ilna).` avec

   `integer*4 ktype,klong,ilna,ichar`

   Des données: il s'agit d'un write implicite de réels, d'entiers ou de caractères. Les réels et entiers peuvent être sur 4 ou 8 octets, ceci étant autodocumenté par la valeur de `ktype` ci-dessus.

5.2 Position du pointeur

Le tableau lgpoint du common lfacom contient la position du pointeur du fichier séquentiel, sous la forme suivante: s'il est vrai, le pointeur est avant un article de données, et sinon il est avant un article d'autodocumentation. C'est à partir de cette information que le logiciel cherche un article demandé à partir de n'importe quelle position du pointeur dans le fichier. Cette information est *interne* au logiciel, elle n'a pas à être connue de l'utilisateur, qui gère lui de façon transparente un appel à un article par son nom.

Chapitre 6

Versions du logiciel

Version 1 (1997-10)

Version disposant des aspects autodocumentation, gestion transparente des aspects physiques du fichier, utilitaires appelables depuis le système d'exploitation, etc...

Version 1.1 (1997-11, iversion=8)

Les chaînes de caractères sont maintenant écrites de façon implicite, alors qu'avant on écrivait la suite des entiers des codes ASCII des caractères. On gagne donc un facteur 4 sur la taille des articles caractères. La comptabilité en lecture des anciens fichiers est assurée.

Version 1.2 (1998-02, iversion=9)

- Correction d'une petite bug concernant l'impression sur output standard en cas d'article non trouvé dans un fichier LFA: la longueur du nom d'article y était parfois erronée.
- Correction d'une bug active dans le seul cas de la machine Cray, et dans le cas suivant: vous ouvrez un fichier avec le logiciel LFA, et ce fichier n'est pas un LFA; vous obtenez alors un code réponse d'erreur. Vous ouvrez alors le fichier avec un autre logiciel. Le logiciel LFA n'ayant pas exécuté l'assign inverse (pour supprimer la transduction FORMAT_IEEE/FORMAT_CRAY), les programmes fortran suivants vont être affectés par la transduction. La correction de bug a consisté ici à effectuer l'ordre assign inverse dans tous les cas de figure du logiciel LFA.
- Enrichissement des codes de réponse d'erreur.

Version 2.0 (1998-03, iversion=9)

- Il est souvent utile d'écrire ou lire des réels ou entiers à une précision différente de celle a laquelle ils sont calculés par le programme utilisateur. C'est ce que permet la version 2 de LFA: le logiciel permet maintenant de lire et d'écrire des données réelles ou entières en 32 ou 64 bits, suivant le choix de l'utilisateur, et ce quel que soit la taille des tableaux utilisateur passés en argument au logiciel. On peut ainsi calculer en 64 bits puis passer le tableau 64 bits au logiciel LFA qui l'écrira en 32 ou 64 bits suivant le choix que vous aurez fait via lfaprecr ou lfapreci. L'appel à lfaprecr ou lfapreci peut être répété plusieurs fois, de sorte que que l'on peut successivement au sein

d'un même fichier écrire des données en 32 et 64 bits. La taille est autodocumentée dans le fichier de sorte que la relecture s'effectuera de façon transparente. Il sera ainsi transparent de lire des fichiers LFA ayant été écrits sur des machines IEEE différentes, avec des précisions différentes.
- La lecture des anciens fichiers par le nouveau logiciel est possible s'ils ont été écrits en entiers 32 bits, et réels 64 bits.
- La lecture des nouveaux fichiers par les anciennes versions du logiciel LFA est impossible.
- La conservation par transitivité de la précision à laquelle chaque article du fichier a été écrit est assurée par les différents utilitaires UNIX lfaedit, lfareu, etc... Ce qui veut dire que les manipulations que vous effectuez via les unitaires UNIX de LFA conserveront la précision que vous aviez initialement choisie pour vos données. L'utilitaire lfa2lfa permet, lui, de modifier cette précision en précisant celle voulue sur le fichier de sortie.
- L'installation a été simplifiée, puisqu'aucun paramètre n'est plus nécessaire pour installer le logiciel: le type de machine est reconnu par le script d'installation, qui crée alors les différentes librairies associées aux précisions entières ou réelles disponibles sur ladite machine, avec un nom explicite, par exemple pour une bibliothèque LFA destinée à être appelée par un logiciel utilisateur à réels sur 8 octets et entiers sur 4: `liblfa_R8I4.a`.
- Le logiciel devient bilingue, en affichant ses messages en français ou anglais suivant la valeur de la variable d'environnement LANG: si LANG a ses deux premiers caractères égaux à "fr", LFA affiche en français; il affiche en anglais dans les autres cas.
- Ajout de l'utilitaire lfadiffart.

Version 2.1 (1998-11, iversion=21)

Modification de l'interprétation de l'autodocumentation des fichiers LFA par le logiciel: avant la version LFA 2.0, le code 1 pouvait correspondre à des réels 4 ou 8 octets, ceci suivant les options avec lesquelles LFA avait été compilé. A partir de la version 2.0, 1 correspond aux réels 8 octets, et 3 aux réels 4 octets; le fichier est donc mieux autodocumenté, puisque LFA sait reconnaître la précision sur fichier et l'interface de façon transparente avec celle du tableau de l'utilisateur.

Cependant il est clair que les versions même récentes de LFA ne peuvent pas "deviner" la précision des réels écrite dans les fichiers antérieurs à LFA 2.0, puisque cette information est absente. On est alors donc obligé de prendre une décision arbitraire. A partir de LFA 2.1, cette précision arbitraire des réels lus sur les fichiers écrits par les versions antérieures à LFA 2.0 est supposée égale à 8 octets (si on a compilé avec `r4pre2`) ou à 4 octets (sinon). Elle peut donc être réglée lors de la *compilation*.