Documentation du module DICT

Philippe WAILLE (UFR IMA, université Joseph Fourier)

Mai 2015

Table des matières

1 Introduction : les principes

1.1 Identification des séquences

Ce dictionnaire exploite une propriété de l'algorithme LZW : les seules séquences non vides qui peuvent être ajoutées au dictionnaire sont formées en ajoutant un octet à la fin d'une des séquences déjà présentes.

Une séquence s peut être identifiée soit par un numéro de type dict_index_t, soit par un couple :
- numéro de type dict_index_t de la séquence préfixe (s privée de son dernier octet)
- valeur de type dict_char_t du dernier octet de s.

1.2 Structure de données

Le dictionnaire est organisé comme un arbre n-aire, un nœud pouvant théoriquement avoir autant de fils que de valeurs possibles d'octet (n=256 pour un fichier binaire) possibles d'un octet.

Pour économiser la mémoire et accélérer la recherche des fils, un nœud n'est pas constitué d'un tableau de n pointeurs à faible taux de remplissage, mais pointe sur la liste triée des fils effectivement présents, chaînés entre eux.

1.3 Numérotation

Le type dict_index_t limite en pratique les numéros de séquences à 15 bits et le dictionnaire à 2^{15} séquences. Un dictionnaire trop gros devient contre-productif : le surcoût de la taille des numéros pour toutes les séquences excède le gain obtenu en allogeant la longueur moyenne des séquences encodées.

Les constantes négatives DICT_ROOT_INDEX et DICT_NONODE permettent de désigner respectivement :
- la séquence vide préfixe de toutes les autres séquences (y compris mono-octet)
- un numéro de séquence illégal

Le premier numéro légal de séquence non vide est donc 0.

1.4 Autoindexation

L'algorithme LZW repose sur un dictionnaire initial composé de toutes les séquences mono-octet. En pratique, chacune d'entre elles aura le même numéro que le contenu de son dernier et unique octet.

L'autoindexation est une optimisation qui permet dès sa création de préremplir le dictionnaire de séquences mono-octet de numéro identique à l'octet, et d'accélérer leur recherche dans l'arbre.

2 API

2.1 Codes d'erreur communs

Les fonctions partagent les codes de retour d'erreur suivant lorsque leurs paramètres d'appel sont invalides :
- DICT_NODICT : dictionnaire invalide
- DICT_INVALID : index ou préfixe invalide

2.2 Création et destruction

La fonction dict_new (max_éléments,nb_autoindex) alloue et initialise un nouveau dictionnaire pouvant mémoriser max_éléments séquences. Les nb_autoindex premiers codes sont affectés aux séquences d'un seul octet de même valeur. L'accélération par autoindexation est désactivée pour nb_autoindex=0.

dict_delete détruit le dictionnaire et libère la mémoire qu'il occupait.

2.3 Recherche et insertion

Recherche(d,p,v,&i) teste si une séquence de préfixe p et dernier octet v est présente dans d :
- dict_insert ajoute la séquence si elle n'était pas déjà présente
- dict_search ne modifie pas le dictionnaire

Interprétation des codes de retour :
- DICT_NOTFOUND : la séquence cherchée est absente du dictionnaire
- DICT_FOUND : la séquence était présente, son numéro est copié dans i
- DICT_ADDED : la séquence, initialement absente, a été ajoutée sous le numéro i
- DICT_FULL : dictionnaire saturé (plus d'insertion possible)

2.4 Informations sur une séquence, sur le dictionnaire

La fonction dict_index_content(d,i,&v,&p) permet de convertir un numéro de séquence i en couple (valeur d'octet v, numéro de préfixe p).
V = DICT_CHAR_EOS dans le cas de la séquence vide racine.

La fonction dict_info(d,&last,&max) donne l'index (last) de la dernière séquence ajoutée et le dernier numéro de séquence utilisable (max).

Les deux fonctions retournent DICT_INFOOK si les paramètres sont corrects.

2.5 Parcours et récupération des octets d'une séquence

La fonction dict_apply_reverse(d,i,f) applique une fonction prédicat p sur le dernier octet de i puis, si f retourne vrai, récursivement sur le dernier octet du préfixe de i et ainsi de suite.

Elle permet de parcourir les octets de i dans l'ordre inverse, du dernier vers le premier et n'a pas d'effet sur la séquence vide racine.

Le parcours est interrompu sur le premier octet visité pour lequel f retourne faux, auquel cas dict_apply_reverse(d,i,f) retourne DICT_FUNCERROR (retourne DICT_ERROR_OK en cas de succès).

La fonction dict_firstchar_length(d,i,&v,&l) retourne l'application de dict_apply_reverse à une fonction permettant de caluler la longueur l et la valeur v du premier octet de la séquence i.

La fonction dict_get_mallocedstring(d,i,&t,&contenu) détermine la taille t de i et remplit un tableau contenu alloué dynamiquement avec la séquence des octets de i. Utilise les deux fonctions précédentes et retourne les mêmes codes d'erreur, ou DICT_MALLOCERROR.

3 Traces de débogage

Il est possible de tracer l'activité du module sous forme de messages envoyés sur la sortie standard d'erreur en augmentant le niveau de trace (0= désactivé).

Le programmeur peut écrire un appel explicite de la fonction dict_set_debug_level pour gérer finement et dynamiquement la génération des traces.

Le niveau de trace peut aussi être défini dans la variable d'environnement DICT_DEBUG_LEVEL, qui est automatiquement consultée par dict_new.