(19) RÉPUBLIQUE FRANÇAISE

INSTITUT NATIONAL
DE LA PROPRIÉTÉ INDUSTRIELLE

PARIS

(11) **N° de publication :** **2 826 804**
(à n'utiliser que pour les commandes de reproduction)

(21) **N° d'enregistrement national :** **02 07994**

(51) Int Cl[7] : **H 03 M 7/30,** H 03 M 7/42

(12) DEMANDE DE BREVET D'INVENTION A1

(22) **Date de dépôt :** 27.06.02.

(30) **Priorité :** 30.06.01 DE 10131801.

(43) **Date de mise à la disposition du public de la demande :** 03.01.03 Bulletin 03/01.

(56) **Liste des documents cités dans le rapport de recherche préliminaire :** *Ce dernier n'a pas été établi à la date de publication de la demande.*

(60) **Références à d'autres documents nationaux apparentés :**

(71) **Demandeur(s) :** *ROBERT BOSCH GMBH Gesellschaft mit beschränkter Haftung* — DE.

(72) **Inventeur(s) :** BRAUN A ARND.

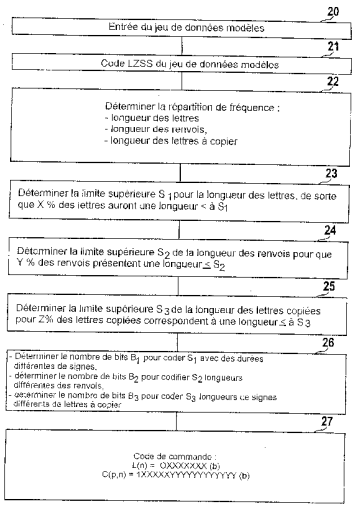
(73) **Titulaire(s) :**

(74) **Mandataire(s) :** CABINET HERRBURGER.

(54) PROCEDE DE COMPRESSION DE DONNEES ET SYSTEME DE NAVIGATION.

(57) Procédé de compression/ décompression de données du type LZSS, caractérisé par l'utilisation des codes de commande suivants:

- un premier code de commande pour une séquence littérale d'une première longueur maximale prédéterminée,
- un second code de commande pour un index avec un renvoi vers une séquence littérale à comprimer, le renvoi correspondant à une seconde longueur maximale, et la séquence littérale à copier ayant une troisième longueur maximale.



FR 2 826 804 - A1

La présente invention concerne un procédé de compression et/ou décompression de données du type de procédé LZSS ainsi qu'un système électronique notamment un système de navigation mettant en œuvre ce procédé.

Les procédés de type LZSS sont connus selon le document US-A-4 876 541 ou T.C. Bell « Better OPM/L Text Compression », « IEEE Trans. On Communications », Vol. COM-34, No. 12. Dec , 1986.

Un procédé LZSS est un développement du procédé Lempel Ziv (Lempel – Ziv – Stac).

Dans l'application du procédé LZSS, on cherche dans les signes transmis en dernier lieu à l'intérieur d'une fenêtre de données d'une certaine longueur, une chaîne de signes correspondant aux signes qui seront transmis ensuite. Si l'on trouve une telle chaîne de signes, on la remplace par un renvoi.

Pour un codage correspondant, on utilise deux codes de commande différents. Le code « L » indique que l'on transmettra ensuite un nombre de signes « vrais », c'est-à-dire littéraux. Au contraire, le code de commande « C » indique qu'à partir des signes déjà transmis, on copiera une chaîne de signes :

- F(s) : fenêtre de données dans laquelle on cherche les chaînes de signes identiques. Cela comprend un nombre de (s) signes précédant la position de lecture actuelle dans le flux de don nées d'entrée.
- L(n) : signes de commande indiquant qu'il y aura ensuite un nombre (n) de lettres, c'est-à-dire une séquence littérale de longueur (n) qui sera transmise.
- C(p, n) : signes de commande d'identification d'une séquence littérale précédente à copier, c'est-à-dire que l'on revient (p) signes en arrière et on copie à partir (n) signes.

La figure 1 donne un exemple du codage d'une chaîne de signes 1 selon le procédé LZSS connu selon l'état de la technique. Le résultat du codage est la chaîne de signes 2 de la figure 1 ; il s'agit pour les signes imprimés en grains, de lettres alphabétiques.

En outre, selon l'état de la technique, on connaît différentes variantes du procédé LZSS, par exemple le procédé LZSS à codage arithmétique adaptatif et le procédé LZSS à codage Huffman adaptatif. On trouvera une description d'ensemble dans le document « redundanz », 5ème exposé, Maximilian Hrabowski (http ://goethe.ira.uka.de/semi-

nare/redundanz/vortrag05//#LZSS). D'autres présentations du procédé LZSS se trouveront à l'adresse suivante http //ttrip1.th worms.de/sem/ws95 96/kompressionsalgorithmen/node19.html et http ://-ttrip1.fh-worms.de/sem/ws95 96/kompressionsalgorithmen/node20.html.

Selon le document US-A-5 502 439, on connaît un procédé de compression de données binaires selon le procédé LZSS. On utilise un tampon dans une mémoire avec accès sélectif pour l'enregistrement provisoire de bits de drapeau générés par exemple pour l'exécution du procédé LZSS. D'autres procédés du type LZSS sont donnés dans les documents suivants US-A-5 701 125, US-A-5 673 042, US-A-5 867 114.

La présente invention a pour but de développer un procédé du type LZSS et un programme d'ordinateur correspondant ainsi qu'un système électronique pour la mise en œuvre.

A cet effet, l'invention concerne un procédé du type défini ci-dessus, caractérisé par l'utilisation des codes de commande suivants :

- un premier code de commande pour des séquences littérales d'une première longueur maximale prédéterminée,
- un second code de commande pour un index à un renvoi vers une séquence littérale à comprimer, le renvoi correspondant à une seconde longueur maximale et la séquence littérale à copier ayant une troisième longueur maximale.

Le procédé selon l'invention du type LZSS permet une décompression particulièrement rapide des données avec un taux de compression simultané, excellent. Selon un mode de réalisation préférentiel de l'invention, le code de commande pour l'exécution du procédé LZSS sera fixé en fonction de la fréquence d'apparition des différentes longueurs de séquences de lettres, des longueurs de séquences de lettres à copier et des longueurs de renvoi.

Par exemple :

- pour déterminer la première longueur maximale, on utilise la répartition de fréquence des longueurs des séquences littérales dans un jeu de données modèle codées par un code de commande sans limitation de longueur effectué par un procédé LZSS,
- pour déterminer la seconde longueur maximale, on utilise la répartition de fréquence des longueurs de l'index dans un jeu de données comprimées, obtenu à l'aide d'un procédé LZSS sans limitation de longueur,

- pour déterminer la troisième longueur maximale, on utilise la répartition de fréquence des longueurs des séquences littérales à copier, dans un jeu de données d'un jeu modèle, comprimées selon un procédé LZSS sans limitation de longueur.

Selon un mode de réalisation préférentiel, on forme des quantités de codes de commande que l'on peut coder par exemple selon le code Huffman pour une compression plus poussée.

Ainsi, pour effectuer le procédé LZSS, on utilise une première quantité d'un premier code de commande et une seconde quantité d'un second code de commande. Dans ce cas par exemple la première quantité contient chaque fois un premier code de commande pour des séquences littérales, à l'intérieur d'une plage de valeurs prédéterminées et les premier et second codes de commande sont codés selon un procédé Huffman, d'après la fréquence d'occurrence des séquences littérales ou la fréquence de l'occurrence du renvoi dans la plage de valeurs concernées. De plus, la seconde quantité présente chaque fois un code de commande pour une plage de valeurs de l'index et une plage de valeurs de la séquence littérale à copier, on code le second code de commande par un codage Huffmann, et on divise la plage de valeurs de l'index en fonction des octets ou d'un multiple d'une longueur d'octet.

Selon un autre mode de réalisation de l'invention, les renvois ne se font que dans une trame d'octets définie par la largeur du bus de données utilisé ou du processeur utilisé. Cela permet d'augmenter d'autant la vitesse de traitement à la décompression. De la même manière, on augmente également le taux de compression.

Ainsi, les premier et second codes de commande et les indications littéraires sont enregistrés dans deux segments séparés l'un de l'autre du résultat de compression, et un troisième code de commande sert à identifier la séparation. De plus le procédé de décompression prévoit les étapes suivantes :

- séparation du code de commande de la partie littérale à l'aide du troisième code de commande,
- enregistrement du code de commande dans un premier segment de mémoire,
- enregistrement de l'information littérale dans un second segment de commande de mémoire,

- accès aux séquences littérales à copier dans la seconde mémoire et enregistrement de la séquence littérale à copier dans une troisième mémoire.

Il est particulièrement avantageux d'utiliser le procédé selon l'invention dans un système électronique par exemple un système de navigation. Dans les systèmes de navigation connus, on utilise de manière générale des CD pour l'enregistrement des banques de données de navigation. Pour loger une quantité aussi importante que possible de données de navigation sur un CD, il est avantageux de comprimer les données de navigation selon un procédé de l'invention. La vitesse de compression des données est pratiquement d'ordre deux, car cette compression ne se fait qu'une fois et non pas pendant le fonctionnement continu.

Pour l'utilisation pratique du système de navigation, la vitesse de décompression est d'une importance capitale, car il faut décomprimer en permanence les données de navigation pendant le fonctionnement du système de navigation pour effectuer la planification routière et déterminer la position. Dans ces conditions, le procédé selon l'invention est particulièrement avantageux, car il permet une décompression particulièrement rapide des données.

La présente invention sera décrite ci-après à l'aide d'un exemple de réalisation représenté dans les dessins annexés, dans lesquels :

- la figure 1 montre le codage d'une séquence de signes selon l'état de la technique,
- la figure 2 montre un ordinogramme d'un mode de réalisation du procédé de l'invention,
- la figure 3 montre la répartition en pourcentage de séquences de lettres et de la longueur de renvoi dans un jeu de données modèles,
- la figure 4 montre un mode de réalisation pour déterminer les quantités du code de commande,
- la figure 5 montre le codage d'une chaîne de signes à l'aide du code de commande de la figure 4,
- la figure 6 montre le transcodage des chaînes de signes codées de la figure 5 à l'aide d'un autre code de commande,
- la figure 7 montre un schéma par blocs d'un système électronique selon l'invention.

Le procédé selon la figure 2 sert à déterminer le code de commande appliqué à un mode de réalisation du procédé de l'invention.

Pour cela, dans l'étape 20, on introduit tout d'abord un jeu de données modèles qui sera soumis dans l'étape 21 à un codage à l'aide d'un procédé LZSS connu selon l'état de la technique. Comme jeu de données modèles, on utilise un jeu de données caractéristiques ou encore un jeu de données réelles.

Dans l'étape 22, on soumet à une analyse statistique le résultat de compression de l'exécution de l'étape 21. Pour cela, on détermine par exemple la répartition de la fréquence des longueurs différentes de séquences littérales produites dans le résultat de compression ainsi que les répartitions des fréquences, des longueurs, des renvois et des longueurs de séquences littérales copiées par l'application de l'étape 21.

Pour optimiser la vitesse de décompression, on fixe ci-après les longueurs maximales. Pour cela, tout d'abord dans l'étape 23, on détermine une limite supérieure S_1 pour la longueur des séquences littérales de façon que X % des lettres contenues dans le résultat de compression de l'étape 21 possède une longueur $\leq$ à S_1. Pour la valeur X %, on peut par exemple prendre 95 %.

De façon correspondante, dans l'étape 24, on détermine une limite supérieure S_2 pour la longueur des renvois, de sorte que Y % des renvois ont une longueur $\leq$ à la limite supérieure S_2. Dans ce cas, on peut également prendre pour Y %, la valeur de 95 %.

Enfin dans l'étape 25, on détermine une limite supérieure S_3 pour la longueur des lettres copiées dans le résultat de compression de l'étape 21 pour que Z % des séquences de lettres copiées représentent une longueur inférieure ou égale à la limite supérieure S_3. Pour Z %, on peut de nouveau choisir 95 %.

Dans l'étape 26, on détermine les nombres de bits nécessaires chaque fois pour le codage et longueurs différentes, c'est-à-dire que l'on détermine le nombre des bits B_1 pour coder S_1 longueurs différentes de séquences littérales, le nombre de bits B_2 pour coder S_2 longueurs différentes de renvois, et le nombre de bits B_3 pour coder S_3 longueurs différentes de séquences littérales à copier.

Sur la base des résultats de l'étape 26, dans l'étape 27, on détermine le code de commande. On distingue entre un code de commande L et un code de commande C par une première position de bits qui, dans l'exemple considéré est la valeur O pour le code de commande L et la valeur 1 pour le mode de commande C.

Dans le code de commande L, on a ensuite un nombre X de positions de bits B_1 pour coder la longueur (n) de la séquence littérale suivante. Après le premier 1 dans le code de commande C, on a un nombre B_2 de positions de bits X pour le codage des longueurs différentes des renvois et ensuite un nombre B_3 de positions de bits Y pour coder les longueurs de signes différentes des séquences littérales à copier.

Pour un jeu de données modèles, on a déterminé par exemple les valeurs suivantes : $S_1 = 128$, $S_2 = 4096$ et $S_3 = 32$. Il en résulte $B_1 = 7$, $B_2 = 12$ et $B_3 = 5$.

Les tableaux de la figure 3 montrent qu'un pourcentage élevé de données n'utilise qu'une partie faible du code de commande possible.

Dans le jeu de données modèles examiné, les séquences littérales de longueur 1 représentaient une partie de 50 % des signes de commande rencontrés L ; les séquences littérales d'une longueur de 2 à 8, représentent une proportion de 25 %, et les séquences littérales supérieures à 8 jusqu'à la limite supérieure S_1 correspondent à une proportion de 25 %.

De façon correspondante, les renvois avec des séquences littérales à copier correspondant à une longueur de 1 à 8 ont une proportion de 70 % du code de commande C. De plus, les renvois d'une longueur de l'index (p) comprise entre une 1 et 32 positions correspondent à une proportion de 50 % du code de commande C ; les renvois d'une longueur comprise entre 33 et 512 positions correspondent à une proportion de 25 % ; les renvois d'une longueur > à 512 jusqu'à la limite supérieure correspondent à une proportion de 25 %.

De façon correspondante, suivant la figure 4, on forme deux quantités différentes de code de commande L et C. Pour le code de commande L, on aura les codes L_1, L_2, L_3 chaque fois pour une plage de longueur des séquences littérales 1, 2 à 9 et 10 à 265. Pour les codes de commande L_1, L_2, L_3, on aura chaque fois un nombre nécessaire de bits B_1 correspondant à 0,3 ou 8. Dans l'exemple examiné ici, le code de commande L_1 est de 001 ; le code de commande L_2 correspondra à 010 et le code de commande L_3 à un code 011 ; la longueur respective pour le codage d'un code de commande correspond dans ce cas à chaque fois trois bits.

La figure 3 montre en outre le codage du code de commande C. Dans l'exemple envisagé, on forme six codes de commande C_1-

C_6 suivant la répartition des renvois de la figure 3. Le code de commande C_1 sera codé par 1001 ; le code de commande C_2 sera codé comme 1010 etc.

Le nombre de bits utilisés pour le codage de chacun des codes de commande C est toujours égal à quatre ; en variante, on peut toutefois coder les codes de commande L et C par exemple selon un procédé Huffman ; dans ces conditions, la probabilité d'arrivée d'un certain code sera prise en compte selon le tableau de la figure 3.

Après avoir déterminé selon le tableau 3 le nombre des codes et leur amplitude, on détermine la fréquence des différents codes sur l'ensemble des codes produits et à partir de cette fréquence, on attribue le code Huffman.

Si le code littéral représente 40 % de tous les codes, et si les codes de copie à chaîne de signes courte correspondent à 70 % de tous les codes de copie, le tableau 3 donne la distribution suivante :

Code	Fréquence
L1	20 %
L2	10 %
L3	10 %
C1	21 %
C2	10,5 %
C3	10,5 %
C4	9 %
C5	4,5 %
C6	4,5 %

Dans ce cas, on aura des longueurs de codes différentes ; le code de fréquence la plus élevée contient le codage le plus court. Dans l'exemple envisagé, il s'agit du code C_1.

Le code de commande C_1 est appliqué pour un renvoi avec un index dans la plage des valeurs de 2 à 33 signes pour une séquence littérale d'une longueur de 2 à 5 signes. Il faut remarquer qu'un renvoi ne se produit que si la longueur du renvoi correspond à au moins deux signes et si la longueur de la séquence littérale à copier à laquelle on se réfère est au moins égale à deux. De manière correspondante, le nombre de bits de codage de la plage de valeurs de l'index (« pointeur ») 5 et le nombre de bits du codage de la plage de valeurs de 2 à 5 correspondent à la longueur des séquences littérales à copier à savoir deux bits. Des associations correspondantes se trouvent dans le tableau de la figure 4 également pour les codes de commande C_2-C_6.

Si les signes dans la séquence à comprimer sont répartis suivant une trame d'octets, par exemple d'une largeur de deux à quatre octets, la compression des données peut être optimisée d'autant, en copiant seulement les longueurs d'index qui se produisent effectivement dans le code de commande C. Par exemple, le nombre de bits du codage de la longueur d'index dans le code de commande C_1 des données peut se réduire suivant une trame de deux octets de cinq à quatre bits, car les renvois d'ordre impair ne se produisent pas par définition. Dans le cas d'une trame d'une longueur de quatre octets, on peut avoir une réduction correspondante d'un autre octet. L'existence de données selon une trame d'octets est appelée de manière générale alignement. L'alignement des données se transmet en fonction des renvois.

La figure 5 montre le codage de la séquence 1 (voir figure 1) selon le procédé de l'invention à l'aide du code de commande de la figure 4. On obtient le résultat de compression 3.

Un inconvénient du résultat de compression 3 est que les séquences littérales contenues dans le résultat de compression 3 ne sont plus alignées sur les limites d'octets du fait du codage à orientation de bits des ordres et c'est pourquoi, il faut prévoir des moyens complémentaires.

Pour éviter ces inconvénients, on sépare les ordres de commande et les séquences littérales du codage pour les répartir tout d'abord en deux flux de données. Le flux de données des séquences littérales est à orientation d'octets. Le flux de données du code de commande est à orientation de bits.

Dès que l'on dispose complètement des deux flux de données, on peut de nouveau les réunir en un unique flux de données en accrochant par exemple les deux flux de données. La séparation des deux flux de données est caractérisée dans le flux de données obtenu par l'accrochage, par un autre code de commande. Celui-ci peut être placé sensiblement en début du flux de données résultant pour assurer la référence de la séparation des flux de données.

La figure 6 montre un exemple correspondant dans lequel le résultat de compression 3 de la figure 5 a été transformé par codage. Tout d'abord, on répartit le résultat de compression 3 en un flux de données 4 de code de commande et un flux de données 5 de séquences littérales.

Par l'accrochage des flux de données 4 et 5, on obtient un flux de données résultat 6. Ce flux est précédé d'un index Z(n) qui désigne le premier signe du flux de données 5.

La figure 7 montre un schéma par blocs d'un système de navigation 7 comportant un lecteur de CD-ROM 8. Le système de navigation 7 comporte en outre un microprocesseur 9 et des zones de mémoire 10, 11, 12. Un CD-ROM placé dans le lecteur 8 contient les données de navigation comprimées selon le procédé de l'invention.

Les séquences de telles données de navigation sont demandées par le système de navigation au lecteur de CD-ROM 8 pour être transmises au système de navigation 7. A la réception d'un flux de données correspondant au flux de données 6 de la figure 6, le microprocesseur 9 répartit le flux de données reçu en un premier flux de données d'un code de commande et un second flux de données de séquences littérales ; cela se fait par l'utilisation de l'index Z(n) placé devant.

La séquence de données de codes de commande est enregistrée dans la zone de mémoire 10 ; les séquences littérales sont enregistrées dans la plage de mémoire 11. Pour le décodage, il faut que le microprocesseur 9 traite uniquement les codes de commande de la zone de mémoire 10 et accède alors aux séquences littérales de la zone de mémoire 11. Après l'exécution des résultats de décompression par le code de commande, on enregistre successivement dans la plage de mémoire 12 sans nécessiter des opérations de décalage. Cela permet un décodage très rapide dans le système de navigation 7, si bien que pendant le déplacement, on peut réagir très rapidement à des changements de trajet ou autres.

On accélère encore plus la décompression, si lors de la compression, on n'autorise que des renvois d'une longueur de signes supérieure à la longueur de la séquence littérale à copier. Par exemple, un renvoi C4 (17, 20) sera alors divisé en C4 (17, 17) et C4 (17, 3). Cela se traduit par une économie de puissance de processeur.

Si les données à comprimer contiennent des structures particulières, on peut obtenir par d'autres procédés complémentaires et le cas échéant d'autres codes de commande, une amélioration complémentaire du taux de compression ou du temps de décompression :

- certaines structures de données ont des plages dans lesquelles se produit une suite longue de mêmes signes ; ces séquences peuvent être codées tout d'abord, de manière préliminaire par un codage basé sur la longueur de chaîne (RUN – LENGTH – ENCODING),
- s'il s'avère que les séquences de code de commande se produisent de manière répétée plusieurs fois les unes à la suite des autres, on peut

les coder par un ordre de répétition. L'avantage est que la séquence de code de commande correspondante ne sera codée qu'une seule fois.

5

REVENDICATIONS

1°) Procédé de compression/décompression de données du type LZSS, caractérisé par
l'utilisation des codes de commande suivants :

- un premier code de commande pour des séquences littérales d'une première longueur maximale prédéterminée,
- un second code de commande pour un index à un renvoi vers une séquence littérale à comprimer, le renvoi correspondant à une seconde longueur maximale et la séquence littérale à copier ayant une troisième longueur maximale.

2°) Procédé selon la revendication 1,
selon lequel,
pour déterminer la première longueur maximale, on utilise la répartition de fréquence des longueurs des séquences littérales dans un jeu de données modèle codées par un code de commande sans limitation de longueur effectué par un procédé LZSS.

3°) Procédé selon les revendications 1 ou 2,
selon lequel,
pour déterminer la seconde longueur maximale, on utilise la répartition de fréquence des longueurs de l'index dans un jeu de données comprimées, obtenu à l'aide d'un procédé LZSS sans limitation de longueur.

4°) Procédé selon l'une quelconque des revendications 1 à 3,
selon lequel,
pour déterminer la troisième longueur maximale, on utilise la répartition de fréquence des longueurs des séquences littérales à copier, dans un jeu de données d'un jeu modèle, comprimées selon un procédé LZSS sans limitation de longueur.

5°) Procédé selon l'une quelconque des revendications 1 à 4,
caractérisé en ce que
pour effectuer le procédé LZSS, on utilise une première quantité d'un premier code de commande et une seconde quantité d'un second code de commande.

6°) Procédé selon la revendication 5,

caractérisé en ce que
la première quantité contient chaque fois un premier code de commande pour des séquences littérales, à l'intérieur d'une plage de valeurs prédéterminées.

7°) Procédé selon la revendication 6,
caractérisé en ce que
les premier et second codes de commande sont codés selon un procédé Huffman, d'après la fréquence d'occurrence des séquences littérales ou la fréquence de l'occurrence du renvoi dans la plage de valeurs concernées.

8°) Procédé selon l'une quelconque des revendications 5, 6, 7,
selon lequel
la seconde quantité présente chaque fois un code de commande pour une plage de valeurs de l'index et une plage de valeurs de la séquence littérale à copier.

9°) Procédé selon la revendication 8,
selon lequel
on code le second code de commande par un codage Huffmann.

10°) Procédé selon les revendications 8 ou 9,
dans lequel
on divise la plage de valeurs de l'index en fonction des octets ou d'un multiple d'une longueur d'octet.

11°) Procédé selon l'une quelconque des revendications précédentes 1 à 10,
dans lequel
les premier et second codes de commande et les indications littérales sont enregistrés dans deux segments séparés l'un de l'autre du résultat de compression, et un troisième code de commande sert à identifier la séparation.

12°) Procédé de décompression d'une chaîne de signes comprimés selon le procédé de l'une quelconque des revendications 1 à 11, et comprenant les étapes suivantes :

- séparation du code de commande de la partie littérale à l'aide d'un troisième code de commande,
- enregistrement du code de commande dans un premier segment de mémoire,
- enregistrement de l'information littérale dans un second segment de commande de mémoire,
- accès aux séquences littérales à copier dans la seconde mémoire et enregistrement de la séquence littérale à copier dans une troisième mémoire.

13°) Programme d'ordinateur sur un support susceptible d'être lu par un ordinateur ou d'être chargé par un réseau d'ordinateur avec des données et des parties de programme pour la mise en œuvre d'un procédé selon l'une quelconque des revendications 1 à 12,
caractérisé en ce que
le programme d'ordinateur est réalisé sous la forme d'un système électronique, de préférence un système de navigation.

14°) Système électronique de préférence un système de navigation, comportant des moyens pour la mise en œuvre des étapes de procédés selon l'une quelconque des revendications 1 à 12.

15°) Système électronique de préférence système de navigation selon la revendication 14, comportant une première zone de mémoire (10) pour enregistrer un code de commande, une seconde zone de mémoire (11) pour enregistrer les séquences littérales et une troisième mémoire (12) pour enregistrer des séquences littérales copiées.

Fig. 1

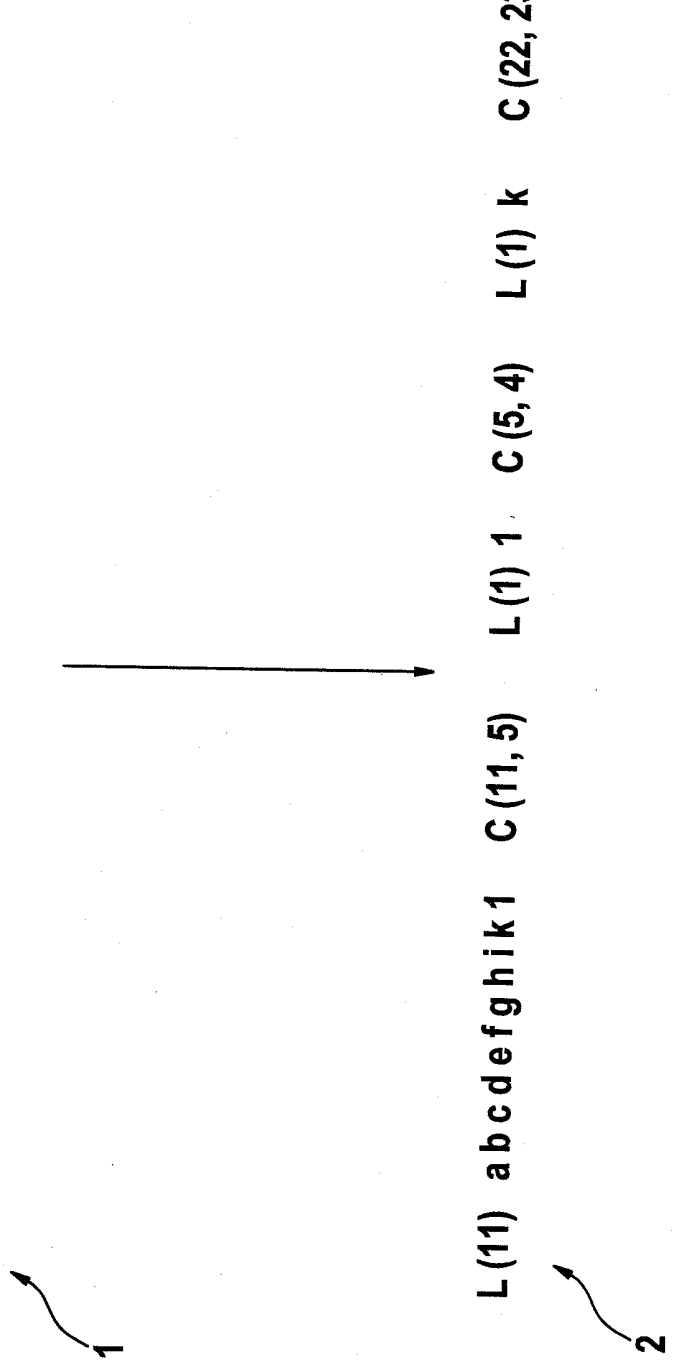

abcdefghik1abcde1bcdekabcdefghik1abcde1bcdeka
1
L(11) abcdefghik1 C(11,5) L(1) 1 C(5,4) L(1) k C(22,23)
2

Fig. 2

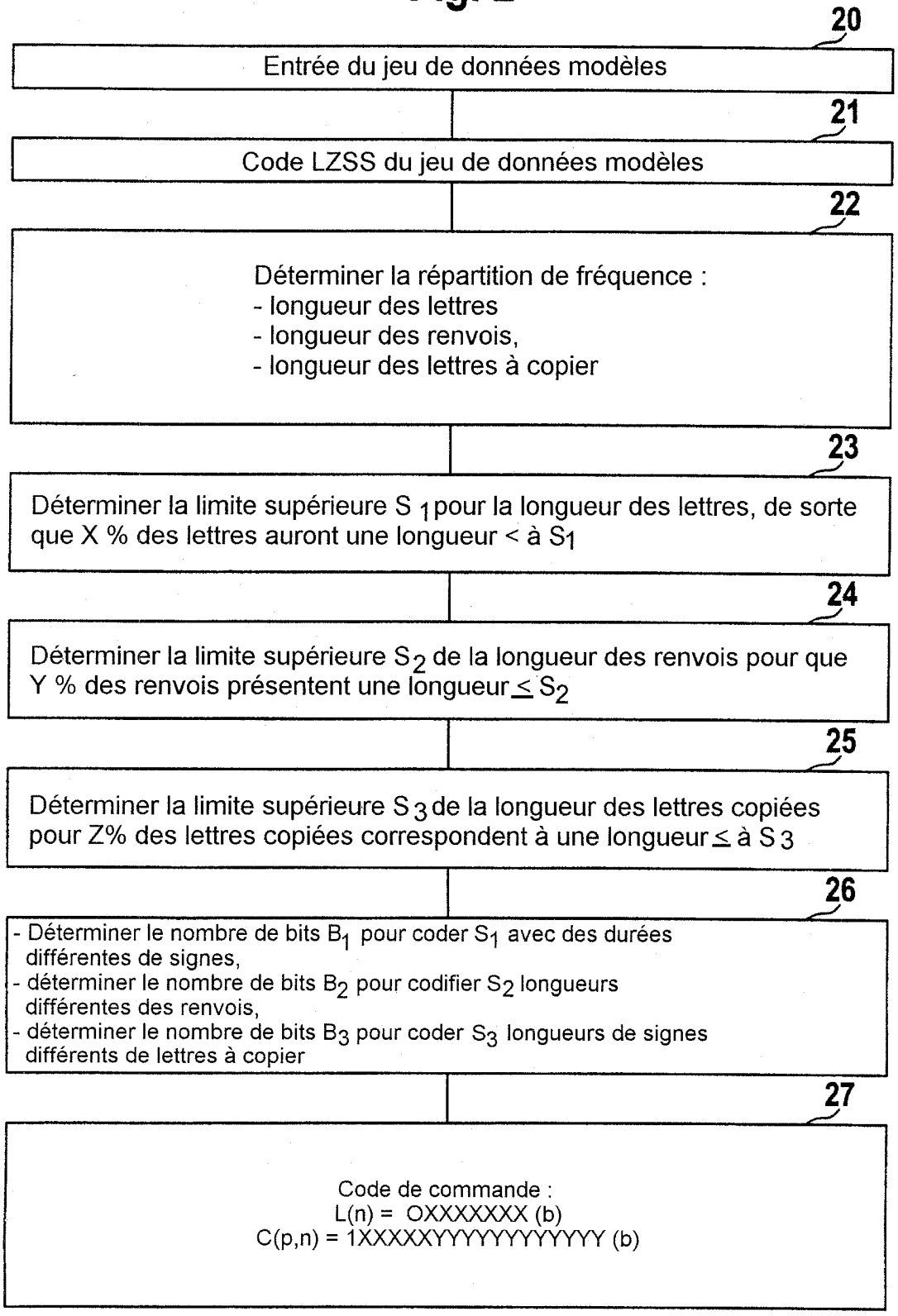

20
Entrée du jeu de données modèles
21
Code LZSS du jeu de données modèles
22
Déterminer la répartition de fréquence :
- longueur des lettres
- longueur des renvois,
- longueur des lettres à copier
23
Déterminer la limite supérieure S 1 pour la longueur des lettres, de sorte que X % des lettres auront une longueur < à S1
24
Déterminer la limite supérieure S2 de la longueur des renvois pour que Y % des renvois présentent une longueur ≤ S2
25
Déterminer la limite supérieure S3 de la longueur des lettres copiées pour Z% des lettres copiées correspondent à une longueur ≤ à S3
26
- Déterminer le nombre de bits B1 pour coder S1 avec des durées différentes de signes,
- déterminer le nombre de bits B2 pour codifier S2 longueurs différentes des renvois,
- déterminer le nombre de bits B3 pour coder S3 longueurs de signes différents de lettres à copier
27
Code de commande :
L(n) = OXXXXXXX (b)
C(p,n) = 1XXXXXYYYYYYYYYYY (b)

Fig. 3

Lettres :

n + 1	Composants du code
1	50 %
2 - 8	25 %
> 8	25 %

Renvois :

n + 1	Composants du code
1 - 8	70 %
> 8	30 %

p + 1	Composants du code
1 - 32	50 %
33 - 512	25 %
> 512	25 %

Fig. 4

	Commande		Argument	
Lettres	Code	Bits	Plage de valeurs	Bits
L1	001	3	1	0
L2	010	3	2 - 9	3
L3	011	3	10 - 265	8

	Commande		Argument 1		Argument 2	
			Pointeur p + 1		Copier n + 1	
Renvois	Code	Bits	Plage de valeurs		Plage de valeurs	Bits
C1	1001	4	2 - 33	5	2 - 5	2
C2	1010	4	34 - 545	9	2 - 5	2
C3	1011	4	546 - 4641	12	2 - 5	2
C4	1100	4	2 - 33	5	6 - 37	5
C5	1101	4	34 - 545	9	6 - 37	5
C6	1110	4	546 - 4641	12	6 - 37	5

Fig. 5

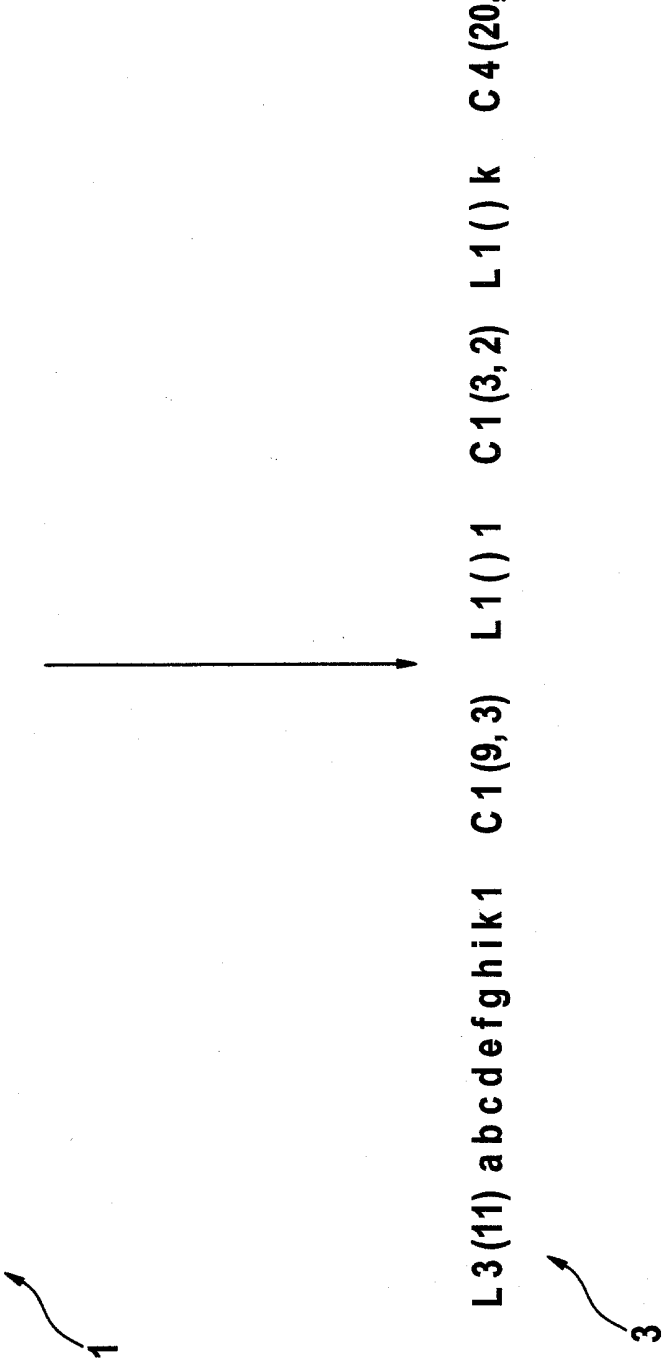

a b c d e f g h i k 1 a b c d e 1 b c d e k a b c d e f g h i k 1 a b c d e 1 b c d e k a
1
L 3 (11) a b c d e f g h i k 1 C 1 (9, 3) L 1 ( ) 1 C 1 (3, 2) L 1 ( ) k C 4 (20, 17)
3

Fig. 6

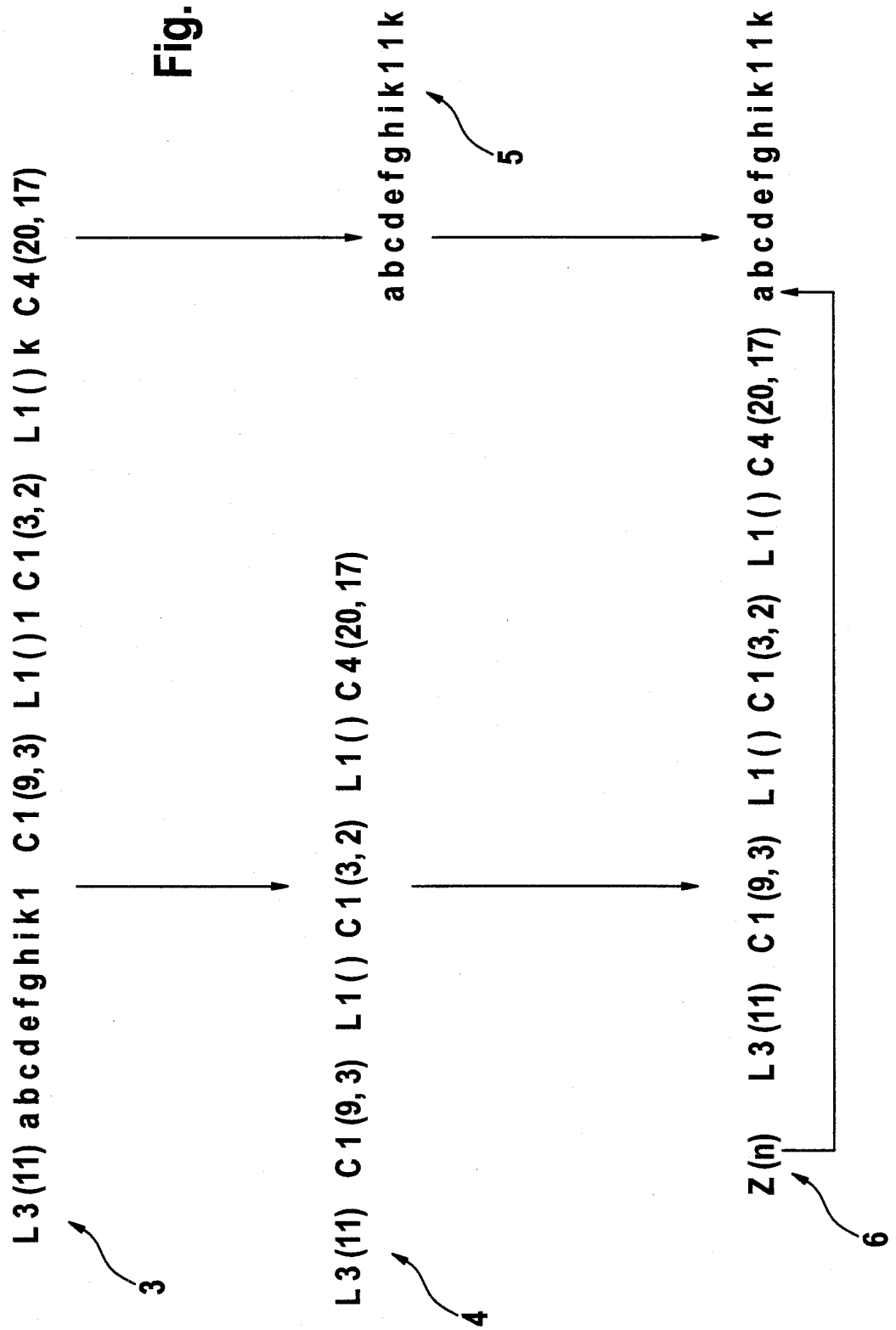
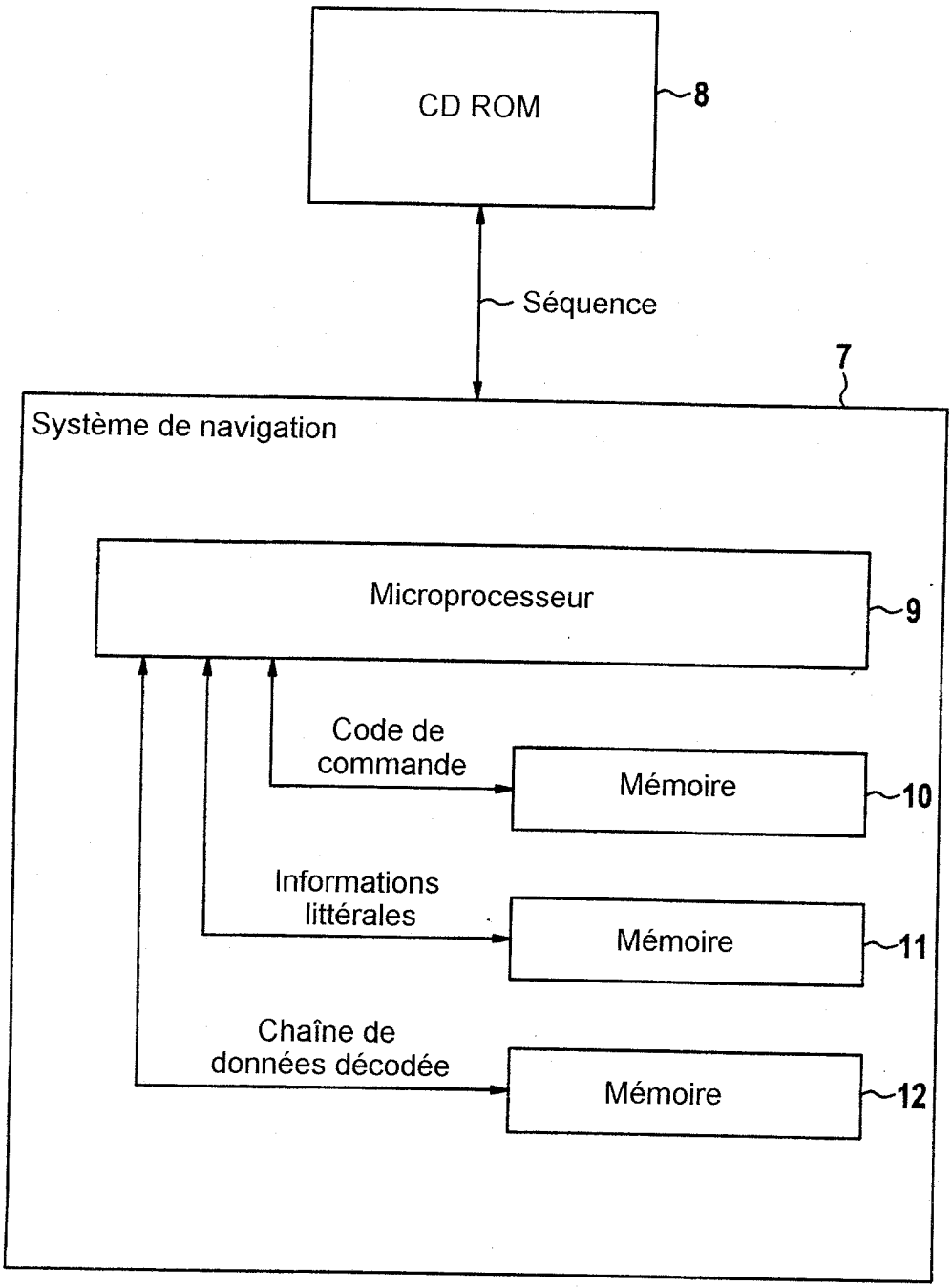

L3(11) abcdefghik1 C1(9,3) L1()1 C1(3,2) L1()k C4(20,17)
3
abcdefghik11k
5
L3(11) C1(9,3) L1() C1(3,2) L1() C4(20,17)
4
Z(n) L3(11) C1(9,3) L1() C1(3,2) L1() C4(20,17) abcdefghik11k
6

Fig. 7


CD ROM
8
Séquence
7
Système de navigation
Microprocesseur
9
Code de commande
Mémoire
10
Informations littérales
Mémoire
11
Chaîne de données décodée
Mémoire
12

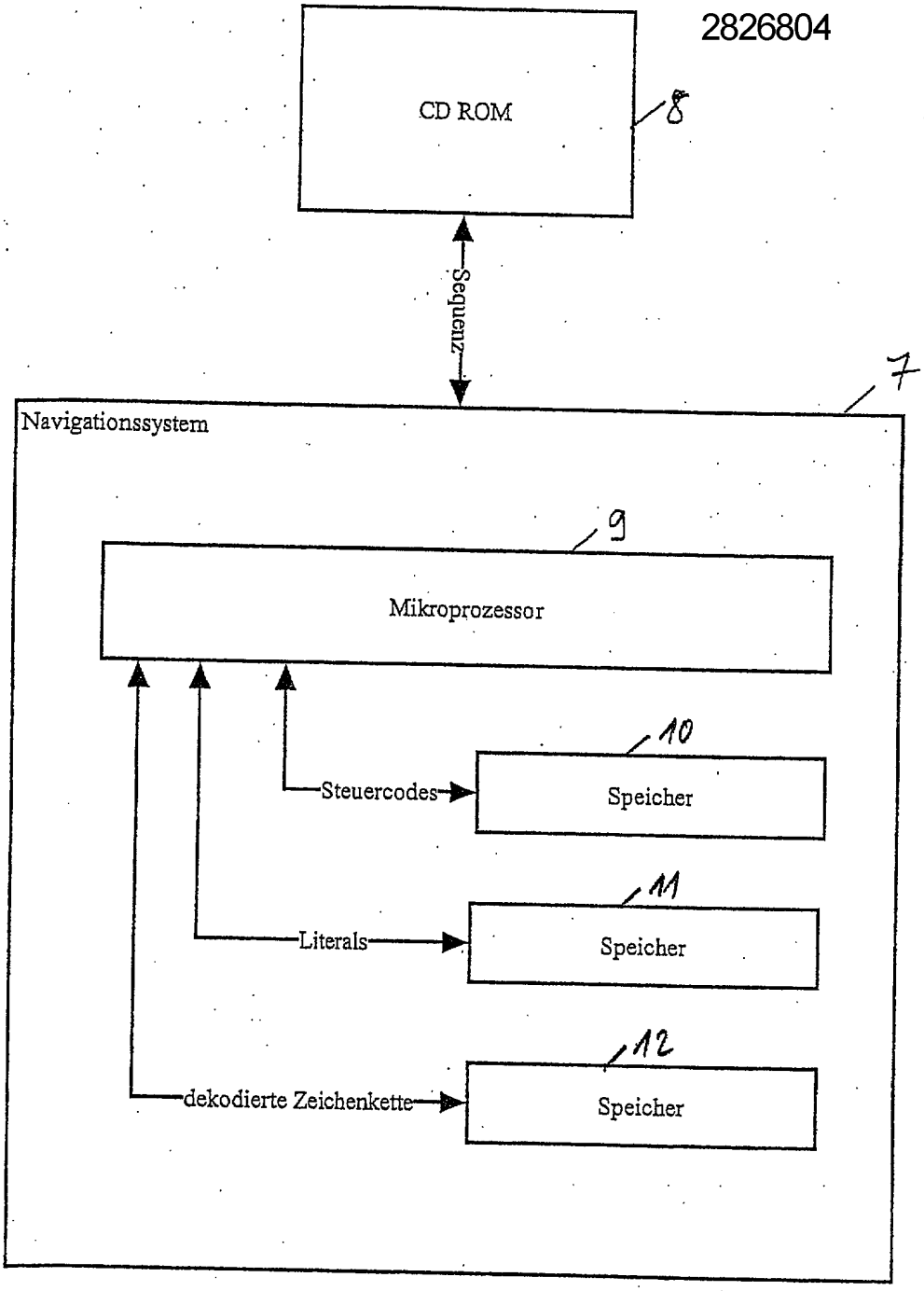

CD ROM
8
Sequenz
7
Navigationssystem
9
Mikroprozessor
10
Steuercodes
Speicher
11
Literals
Speicher
12
dekodierte Zeichenkette
Speicher

Fig. 7