

(12) DEMANDE INTERNATIONALE PUBLIÉE EN VERTU DU TRAITÉ DE COOPÉRATION
EN MATIÈRE DE BREVETS (PCT)

(19) Organisation Mondiale de la Propriété
Intellectuelle
Bureau international



(43) Date de la publication internationale
26 juillet 2007 (26.07.2007)

PCT

(10) Numéro de publication internationale
WO 2007/082948 A1

(51) Classification internationale des brevets :
G06F 17/27 (2006.01)

Julien [FR/FR]; 8, Allée Des Pierrats, F-95870 Bezons (FR).

(21) Numéro de la demande internationale :
PCT/EP2007/050569

(74) Mandataires : **DUDOUIT, Isabelle** etc.; 31-33 Avenue Aristide Briand, F-94117 Arcueil (FR).

(22) Date de dépôt international :
19 janvier 2007 (19.01.2007)

(81) États désignés (*sauf indication contraire, pour tout titre de protection nationale disponible*) : AE, AG, AL, AM, AT, AU, AZ, BA, BB, BG, BR, BW, BY, BZ, CA, CH, CN, CO, CR, CU, CZ, DE, DK, DM, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IS, JP, KE, KG, KM, KN, KP, KR, KZ, LA, LC, LK, LR, LS, LT, LU, LV, LY, MA, MD, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PG, PH, PL, PT, RO, RS, RU, SC, SD, SE, SG, SK, SL, SM, SV, SY, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, ZA, ZM, ZW.

(25) Langue de dépôt : français

(26) Langue de publication : français

(30) Données relatives à la priorité :
06 00537 20 janvier 2006 (20.01.2006) FR

(71) Déposant (*pour tous les États désignés sauf US*) :
THALES [FR/FR]; 45 Rue De Villiers, F-92200 Neuilly Sur Seine (FR).

(84) États désignés (*sauf indication contraire, pour tout titre de protection régionale disponible*) : ARIPO (BW, GH, GM, KE, LS, MW, MZ, NA, SD, SL, SZ, TZ, UG, ZM, ZW), eurasien (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), européen (AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI,

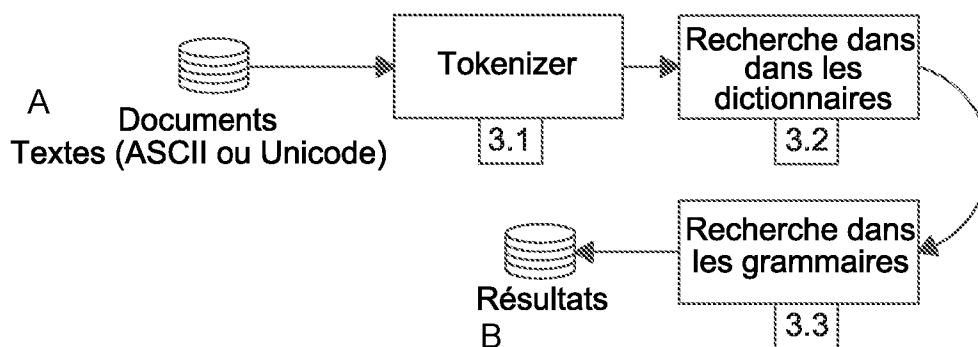
(72) Inventeur; et

(75) Inventeur/Déposant (*pour US seulement*) : **LEMOINE,**

[Suite sur la page suivante]

(54) Title: METHOD AND DEVICE FOR RETRIEVING DATA AND TRANSFORMING SAME INTO QUALITATIVE DATA OF A TEXT-BASED DOCUMENT

(54) Titre : PROCEDE ET DISPOSITIF POUR EXTRAIRE DES INFORMATIONS ET LES TRANSFORMER EN DONNEES QUALITATIVES D'UN DOCUMENT TEXTUEL



A TEXT-BASED DOCUMENTS (ASCII OR UNICODE)

3.1 TOKENIZER

3.2 DICTIONARY SEARCH

3.3 GRAMMAR SEARCH

B RESULTS

(57) Abstract: The invention concerns a method for retrieving data from a data file including a first step whereby the data are transmitted to a device (3.1) or tokenizer for converting same during a first step into elementary units or tokens, the elementary units being transmitted to a second step of dictionary search and a third step (3.3) of grammar search. The invention is characterized in that for the converting step, a sliding window of given dimension is used, the data are converted into tokens as they reach the tokenizer and the tokens are transmitted as they are formed at the dictionary search step (3.2), then at the grammar search step (3.3).

[Suite sur la page suivante]

WO 2007/082948 A1



FR, GB, GR, HU, IE, IS, IT, LT, LU, LV, MC, NL, PL, PT,
RO, SE, SI, SK, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA,
GN, GQ, GW, ML, MR, NE, SN, TD, TG).

Publiée :

— avec rapport de recherche internationale

En ce qui concerne les codes à deux lettres et autres abréviations, se référer aux "Notes explicatives relatives aux codes et abréviations" figurant au début de chaque numéro ordinaire de la Gazette du PCT.

(57) Abrégé : Procédé pour extraire des informations à partir d'un fichier de données comprenant une première étape où les données sont transmises à un dispositif (3.1) ou $\leq \text{tokenizer} \geq$ adapté à les convertir au cours d'une première étape en unités élémentaires ou $\leq \text{tokens} \geq$, les unités élémentaires étant transmises à une deuxième étape de recherche des dictionnaires (3.2) et une troisième étape (3.3) de recherche dans des grammaires, caractérisé en ce que pour l'étape de conversion, on utilise une fenêtre glissante de taille donnée, on convertit en $\leq \text{tokens} \geq$ les données au fur et à mesure de leur arrivée dans le tokenizer et on transmet les tokens au fur et à mesure de leur formation à l'étape de recherche dans des dictionnaires, (3.2), puis à l'étape de recherche des grammaires, (3.3).

PROCEDE ET DISPOSITIF POUR EXTRAIRE DES INFORMATIONS ET LES TRANSFORMER EN DONNEES QUALITATIVES D'UN DOCUMENT TEXTUEL

L'invention concerne notamment un procédé d'extraction d'informations et de transformation en données qualitatives d'un document textuel.

Elle est utilisée notamment dans le domaine de l'analyse et de la compréhension de documents textuels.

5 Dans la description, on désigne sous le mot « token » la représentation d'une unité par un motif de bits et par « tokenizer » le dispositif adapté à effectuer cette conversion. De même, le terme « match » a pour sens l'« identification » ou la « reconnaissance ».

10 En présence de documents non structurés, par exemple de textes, le problème posé est d'extraire l'information pertinente tout en gérant la complexité et les ambiguïtés de la langue naturelle.

Aujourd'hui, les flux d'informations sont de plus en plus présents et leur analyse est nécessaire si l'on veut améliorer la productivité et la rapidité de lecture des textes.

15 Plusieurs méthodes d'extraction sont connues de l'art antérieur. Par exemple, la méthode utilisée par AT&T dont un exemple est accessible par le lien internet <http://www.research.att.com/sw/tools/fsm/>, la méthode développée par Xerox illustrée sur le lien Internet <http://www.xrce.xerox.com/competencies/content-analysis/fst/home.en.html> et la méthode utilisée par Intex/Unitex/Nooj illustrée sur le
20 lien <http://www-igm.univ-mlv.fr/~unitex/>.

Toutes ces techniques ont toutefois comme inconvénients, de ne pas être suffisamment flexibles et performantes, car l'accent a été donné sur l'aspect linguistique et sur la puissance d'expression, plutôt que sur l'aspect industriel. Elles ne permettent pas de traiter des flux importants en temps raisonnable tout en
25 conservant la qualité d'analyse.

L'objet de l'invention repose notamment sur une nouvelle approche : on choisit une taille de fenêtre en début de procédé, on traite les « tokens » un par un,

les tokens arrivant en flux, on applique ensuite la recherche dictionnaire et les grammaires recevant les « tokens » les uns à la suite des autres, dans le cas où elles sont utilisées de manière séquentielle.

L'objet de la présente invention concerne un procédé pour extraire des informations à partir d'un fichier de données comprenant une première étape où les données sont transmises à un dispositif ou « tokenizer » adapté à les convertir au cours d'une première étape en unités élémentaires ou « tokens », les unités élémentaires étant transmises à une deuxième étape de recherche des dictionnaires et une troisième étape de recherche dans des grammaires, caractérisé en ce que pour l'étape de conversion, on utilise une fenêtre glissante de taille donnée, on convertit en « tokens » les données au fur et à mesure de leur arrivée dans le tokenizer et on transmet les tokens au fur et à mesure de leur formation à l'étape de recherche dans des dictionnaires, puis à l'étape de recherche des grammaires.

L'objet de la présente invention offre notamment les avantages suivants :

- l'architecture permet d'éviter la duplication des données et d'utiliser plusieurs grammaires en parallèle ou en série sans résultat intermédiaire,
- du fait de la rapidité de la méthode mise en œuvre, on peut appliquer une multitude de grammaires complexes et donc extraire une grande quantité d'informations des documents sans dégrader les modèles linguistiques,
- l'architecture gère nativement la priorité des grammaires ce qui permet de définir des "modèles à échelles".

D'autres caractéristiques et avantages de la présente invention apparaîtront mieux à la lecture de la description qui suit d'un exemple donné à titre illustratif et nullement limitatif annexé des figures qui représentent :

- La figure 1, un schéma fonctionnel du fonctionnement général de la chaîne de traitement du domaine de l'analyse de documents,

- La figure 2, un schéma fonctionnel des traitements qui peuvent être effectués dans une chaîne de traitement,
- La figure 3, un schéma fonctionnel du procédé selon l'invention permettant d'extraire des entités, des relations entre ces entités, et de convertir des documents en données numériques,
- La figure 4, un exemple d'automate de conversion d'un code (grammatical, flexionnel, sémantique ou syntaxique) en entier,
- La figure 5, un automate permettant de reconnaître une suite d'entiers représentant les codes (grammaticaux, flexionnels, sémantiques et syntaxiques) définis en figure 4,
- La figure 6, un procédé de construction d'un sous dictionnaire optimal pour un ensemble de grammaires à partir d'un dictionnaire d'origine,
- La figure 7, un procédé de suppression des transitions vide dans un transducteur,
- La figure 8, un exemple d'automate pour illustrer le procédé de la figure 7,
- La figure 9, la sortie du procédé de la figure 7 appliqué à l'automate de la figure 8,
- La figure 10, un ensemble de lemmes et de formes fléchies avant la séparation en deux automates,
- La figure 11, l'automate sur lettre des lemmes de la figure 10,
- La figure 12, l'automate sur lettre des formes fléchies de la figure 10,
- La figure 13, les étapes d'un procédé permettant de calculer les nœuds successeurs d'un nœud de l'automate à partir d'une entrée,
- La figure 14, une utilisation des grammaires de réécriture et d'extraction,
- La figure 15, un procédé de détection des «matches» dans un automate,

- La figure 16, un procédé de mise à jour des «matches» potentiels, ce procédé est utilisé par le procédé de la figure 15,
- La figure 17, la gestion de la priorité entre deux grammaires G1 et G2 (G2 étant prioritaire sur G1) via une méthode de scoring ou de sélection du «matche» de plus haute priorité lorsqu'il y a chevauchement,
- La figure 18, la gestion de la désambiguïsation quand il y a un chevauchement entre une grammaire d'extraction et une grammaire de désambiguïsation, et
- La Figure 19 un exemple d'application du procédé selon l'invention pour un serveur de messagerie.

La figure 1 représente une chaîne de traitement général d'analyse de documents. Dans la majorité des cas, cette chaîne comporte, par exemple :

- un élément chargé de convertir n'importe quel format en entrée vers un format texte, bloc 1.1,
- un module d'extraction de méta-données comme la date, l'auteur, la source, ...bloc 1.2,
- un module de traitement sur ces documents bloc 1.3,
- un module d'indexation, bloc 1.4, pour des recherches et des utilisations ultérieures.

Le procédé selon l'invention se situe plus particulièrement au niveau du bloc de traitement 1.3.

Sur la figure 2, on a illustré des exemples de traitements classiques comme le résumé de documents, 4 ou la recherche de documents doublons, 5.

Le procédé selon l'invention a notamment pour fonction d'effectuer les traitements suivants :

- l'extraction d'entités 6 : par exemple l'extraction de personnes, de faits, de la gravité d'un document, des sentiments, etc.
 - l'extraction de relations 7 entre les entités : par exemple, les relations entre les dates et les faits, entre les personnes et les faits, etc.
- 5 • la conversion 8 d'un document en un ensemble de données numériques pour un traitement ultérieur comme de la classification automatique, de la gestion de connaissance, etc.

Pour effectuer ces traitements, on utilise, par exemple, un ensemble de documents sous forme de fichiers ou de zones mémoire en ASCII ou Unicode. On
10 applique ensuite le procédé de transformation d'un texte décrit à la figure 3 qui se décompose notamment en 3 grandes étapes :

- 1) le découpage d'un document source en un ensemble d'unités élémentaires ou « tokens », par un dispositif ou « Tokenizer », 3.1, adapté à convertir un document en éléments,
- 15 2) la reconnaissance des unités simples et composées, 3.2, présentes dans les dictionnaires,
- 3) les applications de grammaires, 3.3.

Etape 3.1

Le procédé selon l'invention utilise une fenêtre glissante d'unités, c'est à
20 dire qu'il conserve uniquement les X derniers « tokens » du texte (X étant un nombre assez grand puisqu'il détermine le nombre maximal d'unités qui pourront être réécrites par une grammaire). La taille de la fenêtre glissante est choisie au début du procédé.

Lors de l'étape de conversion des données en « tokens », le tokenizer
25 3.1 convertit les données au fur et à mesure qu'elles sont reçues avant de les transmettre sous forme de flux à l'étape de recherche dans un dictionnaire, 3.2.

Les types de « tokens » sont par exemple :

- espace : retour chariot, tabulation, ...

- séparateur : slash ; parenthèses ; crochets ; ...
 - ponctuation : virgule, point-virgule, point d'interrogation, point d'exclamation,...
 - nombre uniquement : de 0 à 9,
- 5
- alphanumérique : ensemble de caractères alphabétiques (dépendant de la langue) et des nombres,
 - fin de document.

Le « tokenizer » 3.1 est pourvu, par exemple, d'un processeur adapté à convertir un caractère minuscule en majuscule et vice versa, puisque cela dépend
10 de la langue.

En sortie de « tokenizer », 3.1, les « tokens » sont transmis au fur et à mesure, à l'étape de recherche dans les dictionnaires, 3.2.

Etape 3.2, la recherche dans les dictionnaires

Les dictionnaires 3.2 sont constitués d'entrées composées notamment
15 des éléments suivants :

- une forme fléchie,
 - un lemme,
 - une étiquette ou « tag » grammaticale,
 - un ensemble de codes flexionnels,
- 20
- un ensemble de codes sémantiques,
 - un ensemble de codes syntaxiques.

Le dictionnaire 3.2 est, par exemple, un automate sur lettre dont chaque nœud possède des attributs linguistiques et peut être final ou non. Un nœud est final lorsque le mot est complètement présent dans le dictionnaire.

Les « tokens » sont transmis au module de recherche des dictionnaires 3.2 sous forme de flux, c'est-à-dire qu'ils arrivent les uns après les autres et sont traités de la même façon les uns après les autres par le module 3.2. Le module vérifie pour chaque « token » s'il correspond ou non à une entrée dictionnaire.

5 Dans le cas où un « token » correspond à une entrée dictionnaire, alors le procédé traite les deux cas suivants :

- soit le nœud correspondant de l'automate est un nœud final: dans ce cas l'entrée dictionnaire est ajoutée dans la fenêtre de « token », ainsi que la position du « token » et du nœud de l'automate dans une liste afin d'identifier
10 une potentielle entité composée,
- soit le nœud n'est pas un nœud final, dans ce cas, la position du « token » est juste un ajout pour identifier une potentielle entité composée.

Dans le deuxième cas, on ne sait pas encore si l'entrée est une entité composée du dictionnaire ou pas, car elle correspond uniquement au début (par
15 exemple on reçoit « pomme » qui correspond partiellement à l'entité composée « pomme de terre »). Si plus tard on reçoit la suite, « de terre », on a alors détecté l'entité composée, autrement on supprime l'entité potentielle puisqu'elle n'est pas présente.

Une option de la recherche dans les dictionnaires, permet de définir que
20 les caractères en minuscules dans le dictionnaire peuvent correspondre à un caractère en majuscule ou en minuscule dans le texte. Par contre, un caractère en majuscule dans le dictionnaire ne peut correspondre qu'à un caractère en majuscule dans le texte. Cette option permet notamment de prendre en compte des documents mal formatés comme, par exemple, un texte entièrement en majuscules
25 (on en rencontre souvent dans des bases de données anciennes).

Selon une variante de réalisation du procédé et dans le but d'optimiser les temps de recherche, le procédé construit un sous-ensemble du dictionnaire pendant la compilation de celui-ci. Un exemple de mise en œuvre d'étapes est donné à la figure 6.

Le procédé récupère toutes les transitions des grammaires qui font référence au dictionnaire (lemmes, tags grammaticaux ...). Toutes ces transitions sont compilées et on sélectionne toutes les entrées dictionnaire qui correspondent au moins à l'une de ces transitions. Les entrées dictionnaire reconnaissent au moins l'une des transactions.

Par exemple, si une grammaire contient uniquement les transitions <ADV(adverbe)+Temps> et <V> comme faisant référence au dictionnaire, on va extraire uniquement les entrées du dictionnaire qui sont des verbes ou des adverbess avec Time ou temps comme code sémantique.

Le processus de compilation des transitions en une transition unique comprend par exemple les étapes suivantes :

- la première étape consiste à extraire, de toutes les grammaires utilisées, l'ensemble des codes grammaticaux, sémantiques, syntaxiques et flexionnels contenus dans chacune des transitions des grammaires, et
- lors d'une deuxième étape, on construit un automate sur lettre qui associe à chaque code un entier unique.

Chaque ensemble de codes est donc constitué d'un ensemble d'entiers que l'on ordonne du plus petit au plus grand et que l'on insère dans un automate sur entier afin de déterminer si oui ou non cette combinaison de code est présente dans les graphes.

Si, par exemple, les grammaires contiennent les codes ADV+Temps et V, alors on a l'automate qui transforme les codes en entier de la figure 4.

Cet automate convertit :

- la chaîne de caractères «ADV» en valeur entière : 1
- la chaîne de caractères «V» en valeur entière : 2
- la chaîne de caractères «Temps» en valeur entière : 3

Une fois l'automate convertissant les codes en entier construit, le deuxième automate représentant les transitions est construit (figure 5). Sur cet

automate, la transition ADV+Temps est représentée par le nœud 2 et la transition V par le nœud 3.

De façon similaire, un automate sur texte est construit pour l'ensemble des lemmes utilisés dans les grammaires. Les lemmes étant du texte, il est facile
5 d'imaginer la conversion en automate sur texte.

De manière détaillée, le schéma de la figure 6 illustre la construction d'un sous dictionnaire optimal. Il comporte par exemple les étapes suivantes : pour chaque entrée E du dictionnaire D, 10, 12, on vérifie, 13, si E «matche» l'automate T représentant les transitions ou, 14, l'automate L contenant les lemmes. Si c'est le
10 cas, on ajoute, E, 15, dans le sous dictionnaire O. Ce processus se répète pour toutes les entrées du dictionnaire D.

Par cet élagage de dictionnaire, on construit le dictionnaire le plus petit possible pour une application donnée, ce qui permet de gagner en performance sur la plupart des grammaires.

15 Les éléments issus de l'étape de recherche de dictionnaire sont transmis un par un et sous forme de flux à l'étape d'application des grammaires dont un exemple est détaillé ci-après.

Etape 3.3, application des grammaires sur les éléments issus de l'étape de recherche des dictionnaires.

20 Avantageusement, le procédé met en œuvre des grammaires qui ont été compilées.

Compilation des grammaires

Avant même de pouvoir utiliser les grammaires dans le procédé selon l'invention, on effectue une compilation qui peut se décomposer en deux étapes :

- 25
- La suppression des transitions vides,
 - La décomposition des transitions en automate sur lettre.

La figure 7 décrit un exemple de suite d'étapes permettant de supprimer les transitions vides d'un automate, 20.

Pour tous les nœuds N de l'automate A, 21, pour toutes les transitions T du nœud N vers un nœud M. Si la transition T est une transition vide et que M est un nœud final, alors on supprime T, 26, et on duplique, toutes les transitions qui ont M comme nœuds de départ en mettant N comme nouveau nœud de départ (on ne change pas le nœud de destination). Si la transition T est une transition vide et que M est un nœud non final, alors on supprime T et on duplique, 27 toutes les transitions qui ont M comme nœud de destination en mettant N comme nouveau nœud de destination (on ne change pas le nœud source). On supprime tous les nœuds inaccessibles, 28, non accessibles par le nœud d'origine.

10 Les figures 8 et 9 schématisent un automate de remplacement sur lequel est appliqué le procédé décrit en relation avec la figure 7 et le résultat obtenu. Cette modification de l'automate permet d'en simplifier le parcours puisque les transitions vides sont toujours 'vrai' et doivent toujours être parcourues. La deuxième étape consiste à transformer l'ensemble des lemmes et l'ensemble des formes fléchies, contenus dans les transitions de l'automate en deux nouveaux automates sur lettres afin d'accélérer les recherches de nœuds suivants.

Par exemple, les transitions du nœud 0 vers 1 dans la figure 10 contiennent un ensemble de lemmes et de formes fléchies.

Une recherche classique devrait donc balayer l'ensemble de ces transitions pour 20 détecter celles qui peuvent correspondre à l'entrée reçue.

La transformation de cet ensemble de lemmes et de forme fléchiée donne deux automates :

- le premier automate contient uniquement les lemmes, c'est à dire «lemme», «autre» et «test» comme le montre la figure 11,
- 25 • le deuxième automate contient uniquement les formes fléchies, c'est à dire «forme», «fléchi» et «test» comme le montre l'automate de la figure 12.

Dans le procédé selon l'invention, une transition d'un nœud vers N autres nœuds est définie notamment par un ensemble de trois automates :

- l'automate des lemmes,
- l'automate des formes fléchies,
- l'automate des codes grammaticaux, syntaxiques, sémantiques et flexionnels.

5 Chacun de ces automates renvoie un entier. S'il y a une reconnaissance ou « matche », cet entier est en fait un indice de tableau dans lequel est stocké l'ensemble des nœuds suivants accessibles par cet état.

La figure 13 représente différentes étapes permettant de calculer les nœuds successeurs à partir d'une entrée de la fenêtre glissante de « tokens ».

10 Le procédé décrit à la figure 13 comporte, par exemple, les étapes décrites ci-après. Lorsqu'un token arrive il y a deux possibilités :

1) le token est une entrée du dictionnaire, il est alors reconnu par le dictionnaire,

2) le token n'est pas reconnu par le dictionnaire.

15 Le but est de calculer pour un nœud actuel N, l'ensemble des nouveaux nœuds atteignables par une entrée E de la fenêtre glissante.

Si l'entrée E est une entrée du dictionnaire, 30, on recherche, 31, les nœuds qui peuvent être atteints par E dans l'automate des codes (grammaticaux, syntaxiques, sémantiques et flexionnels) du nœud N et, 32, dans l'automate des
20 lemmes du nœud N. On ajoute tous ces nœuds qui peuvent être atteints, à la liste L.

Si l'entrée E n'est pas une entrée du dictionnaire, on recherche, 33, les nœuds pouvant être atteints par E dans l'automate des formes fléchies du nœud N et on les ajoute dans la liste L.

Application des grammaires sur la fenêtre glissante de tokens

25 Les grammaires locales se décomposent, par exemple, de deux manières :

- les grammaires d'extraction uniquement (représentées par des automates à états finis) qui sont exécutées en parallèle,
- les grammaires de réécriture (représentées par des transducteurs) qui sont appliquées de manière séquentielle.

5 Le schéma 14 illustre l'utilisation des grammaires de réécriture (ou transformation) et d'extraction sur des flux de tokens et les entrées dictionnaires.

Grammaire d'extraction

Les grammaires d'extraction 42i utilisent la suite de tokens et d'entrées du dictionnaire 40 définie auparavant pour détecter un «matche» dans un automate.

10 Pour cela, on utilise une liste de candidats potentiels d'extraction notée P qui contient les éléments suivants :

- l'indice du prochain nœud à tester,
- la position du prochain token attendu,
- la position d'origine de ce candidat.

15 Ces informations permettent de détecter si un nouveau token «complète» un «matche» potentiel ou non en regardant si sa position est celle qui est attendue et s'il valide une ou plusieurs transitions.

Un exemple de sous-procédé permettant de mettre à jour les «matches» potentiels et de détecter les «matches» complet est décrit dans la figure 15, qui
20 utilise lui-même un sous-procédé de mise à jour de la liste de clients potentiels dont les étapes sont détaillées à la figure 16.

La figure 15 représente un exemple d'étapes permettant de mettre à jour les «matches» potentiels et de détecter les «matches» complet.

Soit P la liste des candidats potentiels d'extraction et Q une liste vide, A
25 un transducteur ou grammaire d'extraction et T une entité.

Pour tous les candidats potentiels d'extraction N de la liste P, on cherche les nœuds accessibles depuis le nœud P en utilisant l'entrée T par le

procédé de recherche des nœuds successeurs décrits à la figure 13. On ajoute ensuite tous les nœuds accessibles à la liste Q en utilisant le procédé de mise à jour de la liste décrit ci-dessous, 51, 52, 53.

Une fois que la liste P est entièrement parcourue, on recherche les
5 nœuds accessibles depuis le nœud original de la grammaire en utilisant l'entrée T par le procédé de recherche des nœuds successeurs, figure 13. On ajoute ensuite, 54, 55 tous les nœuds accessibles à la liste Q en utilisant le procédé de mise à jour de la liste décrit en relation à la figure 16. On ajoute les éléments de la liste Q dans la liste P.

10 Le procédé de mise à jour décrit à la figure 16 comprend notamment les étapes suivantes :

- soit P la liste des candidats potentiels d'extraction, N la liste des nœuds pouvant être atteints,

- pour tous les nœuds I identifiés comme étant accessibles par le
15 procédé précédant, 61, 62, si I est un nœud final (ou terminal) de la grammaire, 63, alors il s'agit d'une occurrence de la grammaire d'extraction (« matche »). Si I possède des transitions vers d'autres nœuds, 64, on ajoute I attendant la prochaine entrée à la liste P, 65.

L'application des dictionnaires permet en outre de détecter les entités
20 composées constituées de plusieurs tokens. C'est pour cette raison que le module de recherche dans les dictionnaires informe les grammaires qu'une position ne peut plus être atteinte et qu'il est dorénavant impossible de recevoir des données à cette position. Le module de recherche envoie, par exemple, un message au module suivant qui le relaie à son tour au sous module (lorsqu'on utilise des grammaires
25 séquentielles).

On a donc réussi à récupérer l'ensemble des «matches» possibles avec une approche permettant de rapidement rajouter/enlever des candidats potentiels.

La sélection du « matche » le plus long ou en utilisant un autre critère tel que la priorité d'une grammaire sur une autre nécessite uniquement un passage
30 linéaire sur les « matches » identifiés.

Grammaire de réécriture

Le fonctionnement des grammaires de réécriture est le même que celui des grammaires d'extractions, excepté que chaque « matche » nécessite une modification partielle ou totale des tokens mis en jeu.

5 La méthode de fonctionnement, selon l'invention, pour ce type de grammaire consiste notamment à stocker le résultat directement dans la fenêtre de tokens. Chaque grammaire de réécriture a sa propre fenêtre qui sera transmise aux grammaires suivantes dans la chaîne de traitement, tel que schématisé à la figure 14.

10 Il y a deux types d'exécution possible pour ces grammaires :

- la réécriture en conservant le plus grand «matche», c'est typiquement le cas d'une grammaire de reconnaissance des phrases qui ajoute un token à la fin de chaque phrase,
- l'identification de tous les «matches» pour le remplissage d'une base de données par exemple (conversion de texte en données numériques).

Identification de tous les «matches» pour la transformation en données structurées

Dans ce cas, chaque élément de la liste des candidats potentiels P est doté d'une liste de références vers les transformations à appliquer sur les tokens.

20 Nous pouvons ensuite appliquer une transformation par un automate sur lettres sur chaque variable pour revenir à des données qualitatives et ainsi transformer le texte en données structurées.

Réécriture en conservant le plus grand «matche»

25 Cette mise en œuvre est utilisée lors de l'application d'une grammaire de reconnaissance de fin de phrase.

Le plus grand « matche » peut correspondre :

- soit à une fin de phrase (on ajoute ainsi le token fin de phrase),

- soit à une désambiguïsation (par exemple «M. Exemple» ne correspond pas à une fin de phrase).

Le résultat de cette réécriture est utilisé par d'autres grammaires. Il est donc nécessaire d'être capable de faire des modifications sur un flux de tokens.

- 5 Pour ce faire, nous décidons de stocker les résultats des «matches» dans la fenêtre de tokens, cela permet de :

- rendre cette réécriture transparente pour les grammaires suivantes,
- sélectionner le plus grand «matche» facilement: il suffit de regarder les remplacements existants et de conserver le plus grand.

10 Application des grammaires en parallèle

L'utilisation de grammaires en parallèle est permise nativement par l'architecture. En effet, il suffit de fournir le flux de tokens sortant d'une grammaire à plusieurs autres grammaires en même temps pour obtenir un parallélisme au niveau de l'extraction.

- 15 En prenant le cas de l'extraction d'entités nommées, nous appliquons une grammaire d'identification des phrases puis nous fournissons ce résultat aux différentes grammaires d'extraction (par exemple le lieu, la date, l'organisation ...). Le même parallélisme que celui décrit dans la figure 14 est ainsi obtenu.

Priorités des grammaires

- 20 Selon une variante de mise en œuvre de l'invention, le procédé met en œuvre des règles de priorité ou une notation statistique ou « scoring » sur les résultats des grammaires d'extraction.

- 25 Ainsi, si nous avons N grammaires, sachant que la grammaire G_i (i appartient à $1..N$) est prioritaire sur les grammaires $G_1.. G(i - 1)$, la méthode consiste à utiliser de manière parallèle ou séquentielle les N grammaires pour extraire l'ensemble des «matches» possibles et conserver uniquement le «matche» de priorité la plus élevée quand il y a une intersection entre deux «matches».

Suivant les applications, on va pouvoir sélectionner :

- le « matche » de priorité la plus élevée pour chaque phrase,
 - un ou plusieurs « matches » par phrase sachant qu'il n'y a pas d'intersection entre eux,
 - un score par phrase, le score étant défini par l'ensemble des «matches».
- 5

La figure 17 illustre un exemple de gestion de la priorité entre deux grammaires G1, 70, et G2, 71, (G2 étant prioritaire sur G1) via une méthode de scoring ou de sélection du «matche» de plus haute priorité lorsqu'il y a chevauchement.

10 Désambiguïsation

Le procédé peut aussi comporter une étape ayant notamment pour fonction de lever l'ambiguïté « désambiguïsation ». Pour cela, on sépare chaque grammaire d'extraction en deux parties :

- la grammaire d'extraction, 72, en tant que telle,
 - une ou plusieurs grammaires permettant de lever une « ambiguïté », 73, et permettant de définir les «contre exemples».
- 15

Il suffit ensuite d'extraire simplement tous les «matches» de ces grammaires en parallèle et de supprimer les «matches» lorsqu'il y a une intersection entre une grammaire d'extraction et une grammaire pour lever l'ambiguïté, comme le montre le schéma de la figure 18.

20

La figure 19 représente un exemple d'utilisation du procédé selon l'invention dans un serveur de messagerie e-mail, dont le contenu des messages arrivant ou entrant est analysé, on extrait des informations du message reçu par le procédé, 83, en exécutant les étapes du procédé détaillées ci-avant, afin de déterminer le service d'une entreprise le plus apte à le traiter (par exemple, marketing, comptabilité, technique) et le transmet, 84, au service adapté pour le traiter.

25

REVENDICATIONS

1 - Procédé pour extraire des informations à partir d'un fichier de données comprenant une première étape où les données sont transmises à un dispositif (3.1) ou « tokenizer » adapté à les convertir au cours d'une première étape en unités
5 élémentaires ou « tokens », les unités élémentaires étant transmises à une deuxième étape de recherche des dictionnaires (3.2) et une troisième étape (3.3) de recherche dans des grammaires, caractérisé en ce que pour l'étape de conversion, on utilise une fenêtre glissante de taille donnée, on convertit en « tokens » les données au fur et à mesure de leur arrivée dans le tokenizer et on
10 transmet les tokens au fur et à mesure de leur formation à l'étape de recherche dans des dictionnaires, (3.2), puis à l'étape de recherche des grammaires, (3.3).

2 – Procédé selon la revendication 1, caractérisé en ce qu'il comprend une étape de génération d'un sous-ensemble du dictionnaire comprenant les étapes suivantes :

- récupérer toutes les transitions des grammaires qui font référence au
15 dictionnaire (lemmes, tags grammaticaux ...),
- compiler toutes les transitions, et
- sélectionner les entrées dictionnaire qui correspondent au moins à l'une de ces transitions.

3 – Procédé selon la revendication 2, caractérisé en ce que l'étape de compilation
20 des transitions en une transition unique comprend les étapes suivantes :

- la première étape consiste à extraire, de toutes les grammaires utilisées, l'ensemble des codes grammaticaux, sémantiques, syntaxiques et flexionnels contenus dans chacune des transitions des grammaires, puis,
- la deuxième étape à construire un automate sur lettre qui associe à chaque
25 code un entier unique.

4 – Procédé selon la revendication 1, caractérisé en ce qu'il comporte une étape de construction d'un sous-dictionnaire optimal comprenant au moins les étapes suivantes : pour chaque entrée E d'un dictionnaire D, on vérifie si l'entrée E reconnaît au moins une des transitions ou au moins un lemme des grammaires qui
5 font référence au dictionnaire.

5 - Procédé selon la revendication 1, caractérisé en ce que l'on utilise une grammaire locale sur la fenêtre glissante des tokens, la grammaire comprenant une grammaire d'extraction et une grammaire de réécriture.

6 - Procédé selon la revendication 1, caractérisé en ce qu'il utilise des grammaires
10 compilées, une grammaire étant définie par un automate à états finis, l'étape de compilation comprenant :

- la suppression des transitions vides,
- la décomposition des transitions en automate sur lettre.

7 – Procédé selon la revendication 6, caractérisé en ce l'étape de suppression des
15 transitions vides d'un automate A composé de plusieurs nœuds comprend les étapes suivantes : pour tous les nœuds N (21) de l'automate A, pour toutes les transitions T du nœud N vers un nœud M,

- si la transition T est une transition vide, et si M est un nœud final, alors on supprime la transition T et on duplique (26) toutes les transitions qui ont M
20 comme nœud de départ en mettant N comme nouveau nœud de départ,

- si la transition T est une transition vide et que M est un nœud final, alors on supprime T on duplique (27) toutes les transitions qui ont M comme nœud de destination en mettant N comme nouveau nœud de destination.

8 – Procédé selon la revendication 7, caractérisé en ce qu'une transition d'un nœud
25 vers N autres nœuds est définie par un ensemble de trois automates : l'automate des lemmes, l'automate des formes fléchies, l'automate des codes grammaticaux, syntaxiques, sémantiques et flexionnels.

9 – Procédé selon la revendication 7, caractérisé en ce que le calcul pour un nœud actuel de l'ensemble des nouveaux nœuds pouvant être atteint par une entrée E de la fenêtre glissante de « tokens » comporte les étapes suivantes :

- 5 • si l'entrée E est une entrée du dictionnaire (30), on recherche (31), les nœuds qui peuvent être atteints par E dans l'automate des codes (32) du nœud N et dans l'automate des lemmes du nœud N et on ajoute les nœuds pouvant être atteints à une liste L,
- 10 • si l'entrée E n'est pas une entrée du dictionnaire, on recherche (33) les nœuds pouvant être atteints par E dans l'automate des formes fléchies du nœud N et on les ajoute à la liste L.

10 – Procédé selon la revendication 1, caractérisé en ce qu'une grammaire d'extraction (42) utilise la suite de tokens et d'entrées du dictionnaire (40) pour détecter les « identifications » dans un automate, et en ce qu'on utilise une liste de candidats potentiels d'extraction P contenant les éléments suivants : l'indice du
15 prochain nœud à tester, la position du prochain « token » attendu, la position d'origine de ce candidat.

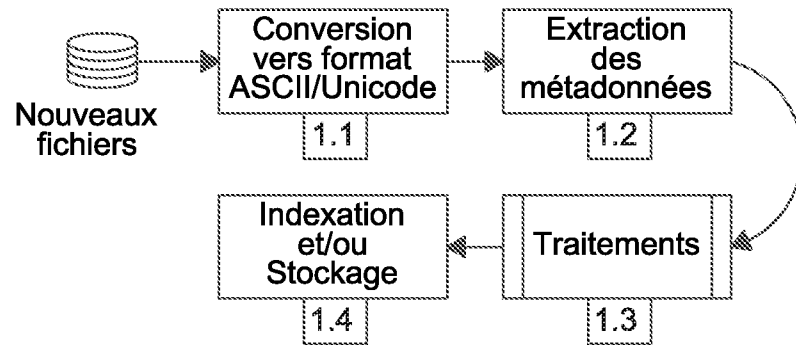


FIG.1

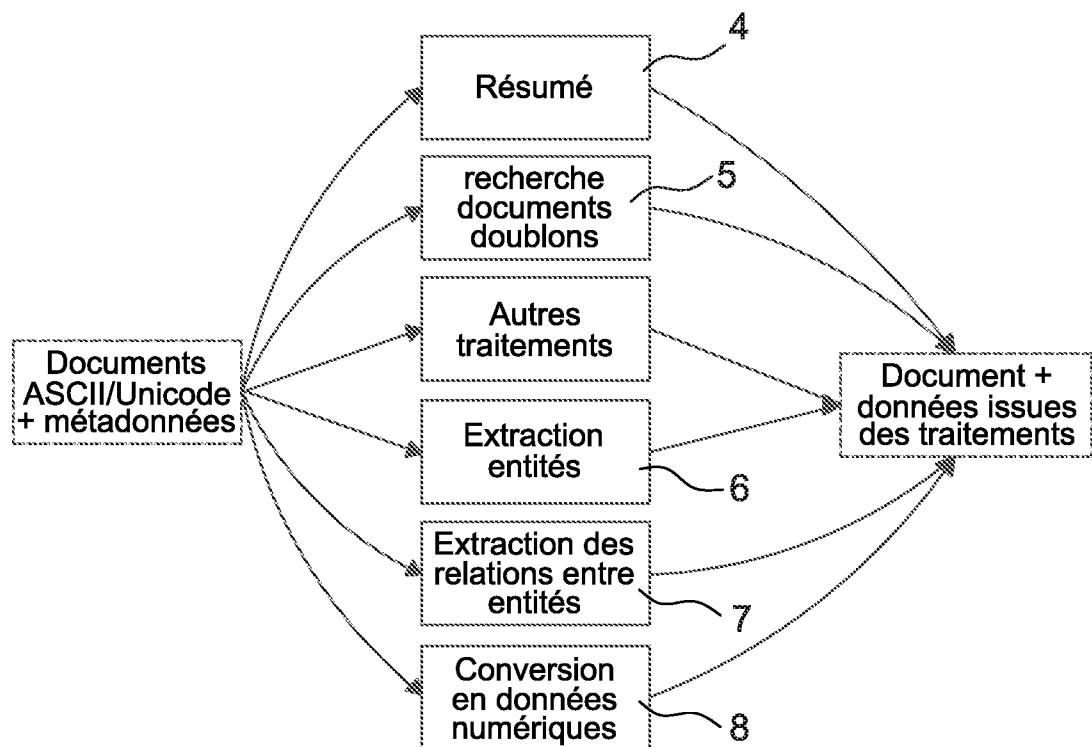


FIG.2

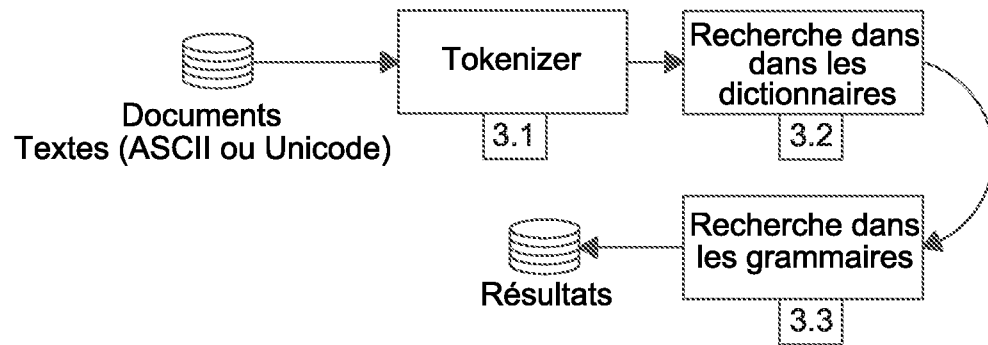


FIG.3

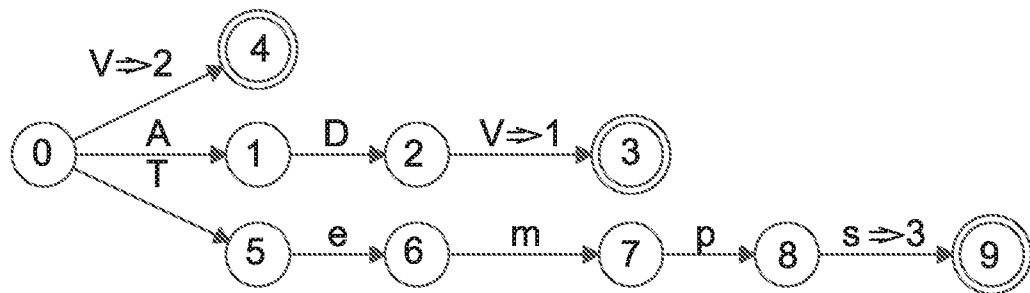


FIG.4

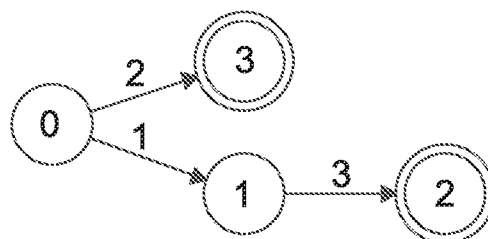


FIG.5

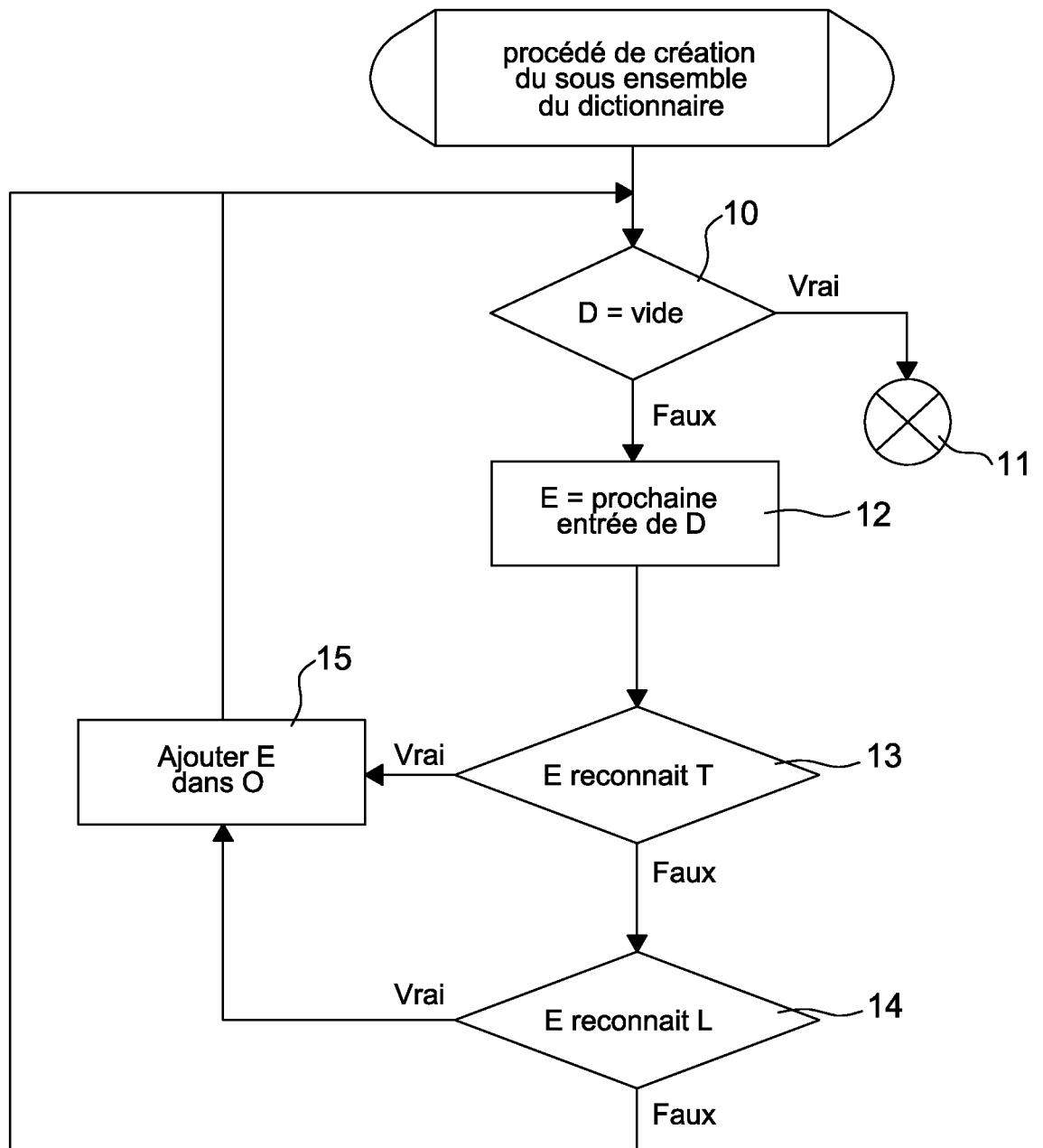


FIG.6

4/10

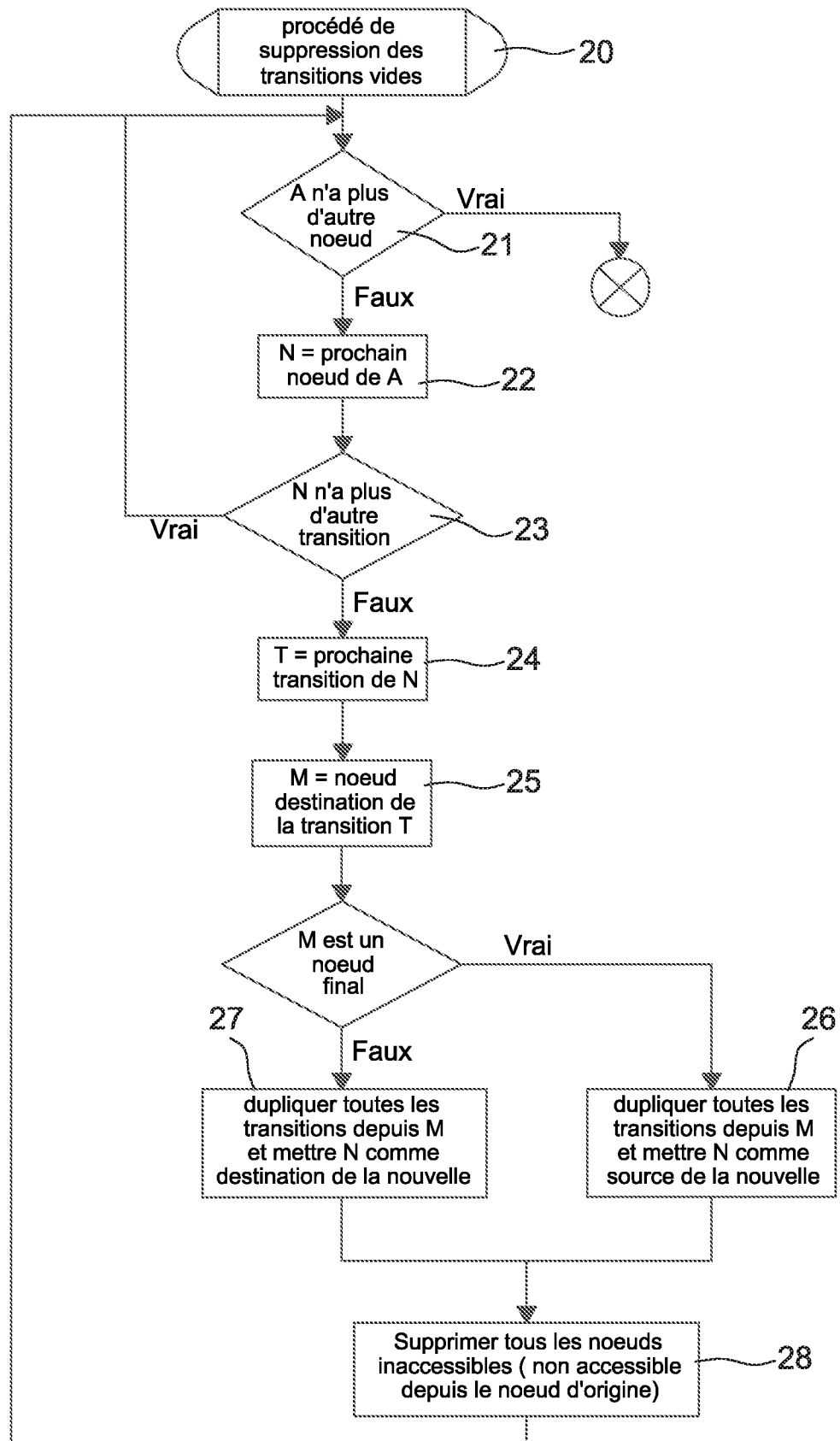


FIG.7

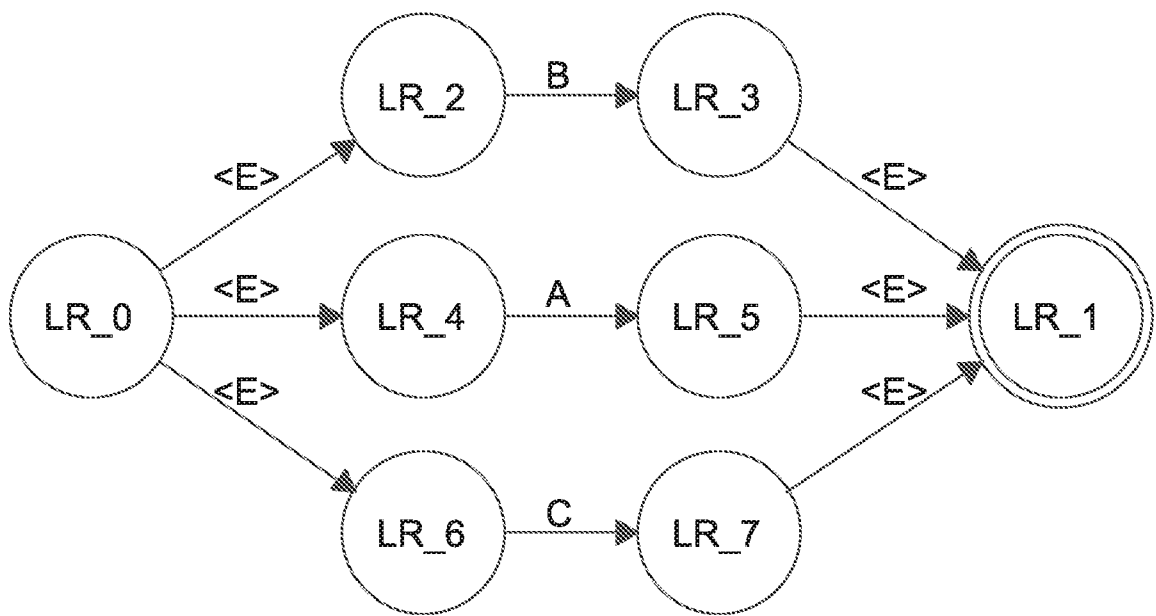


FIG.8

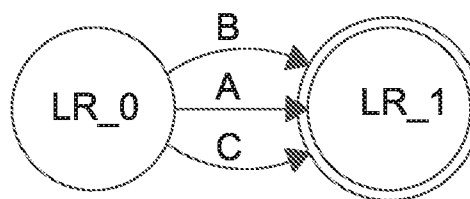


FIG.9

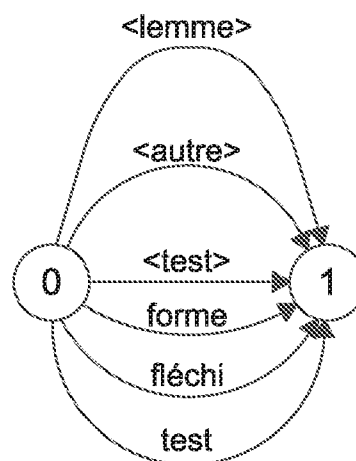


FIG.10

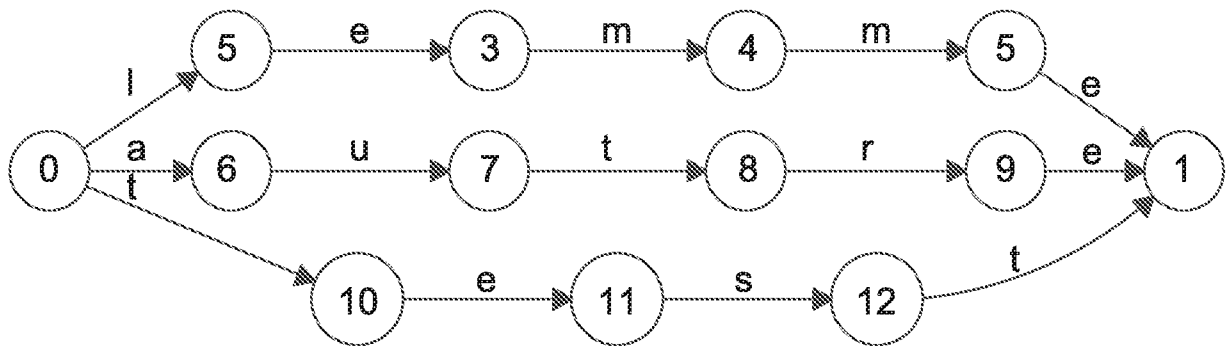


FIG.11

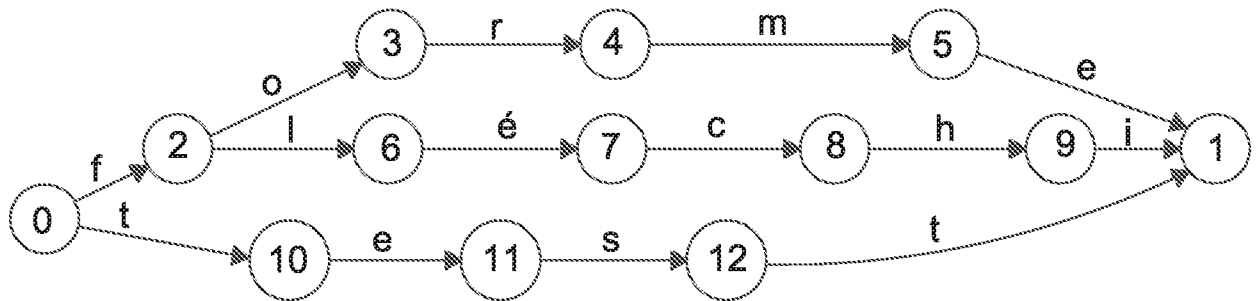


FIG.12

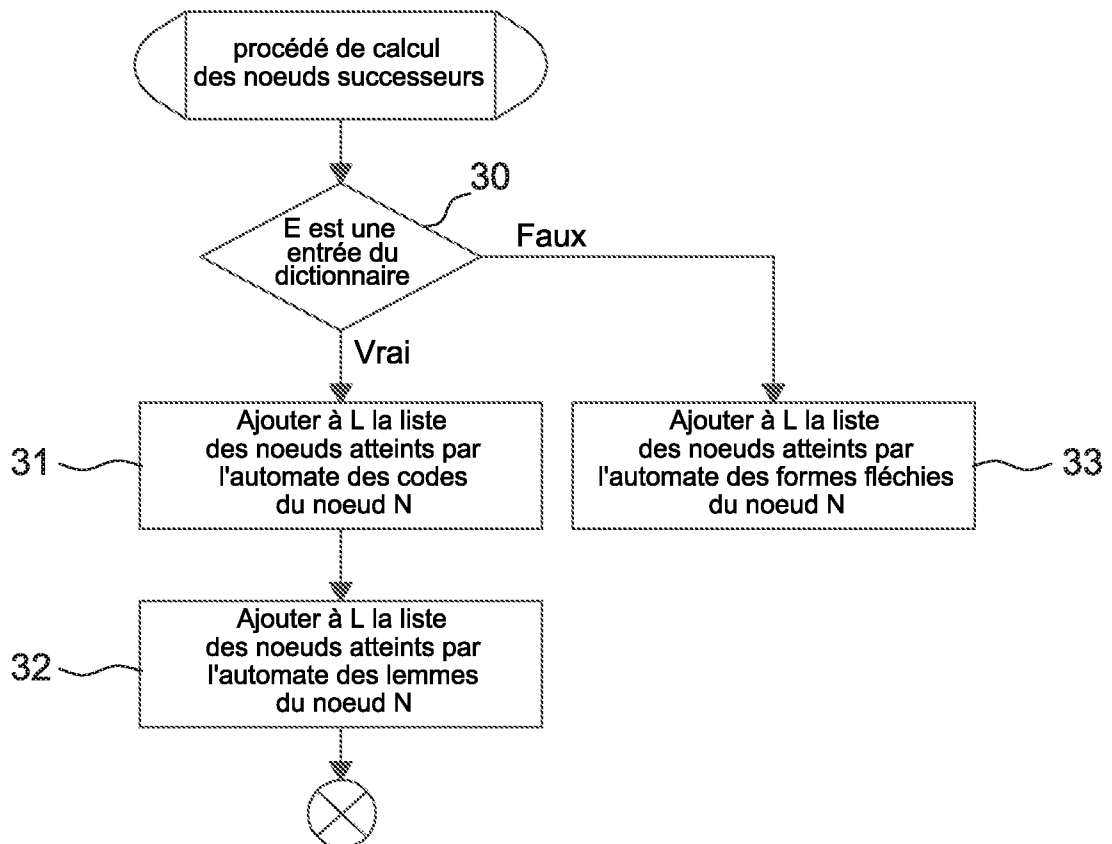


FIG.13

7/10

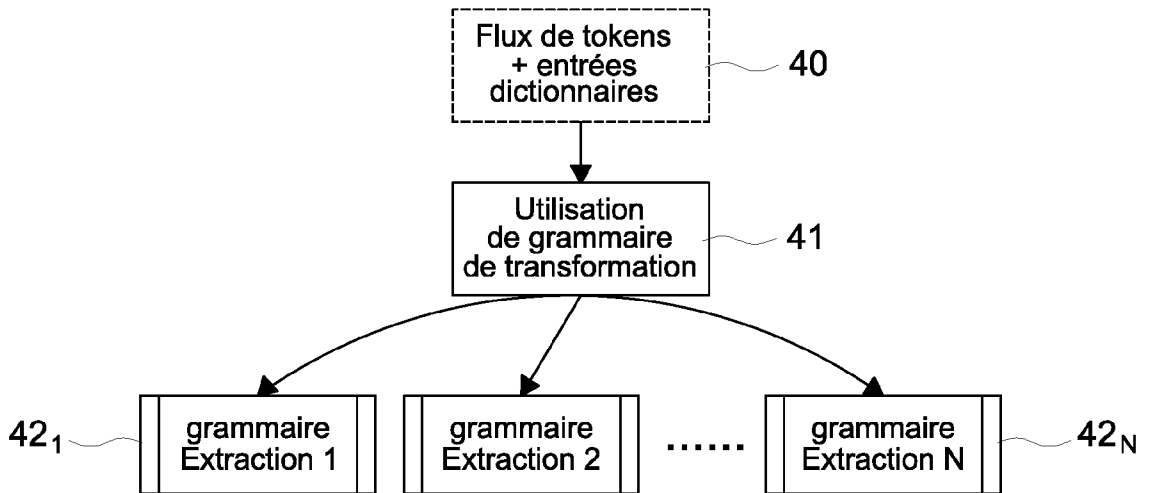


FIG.14

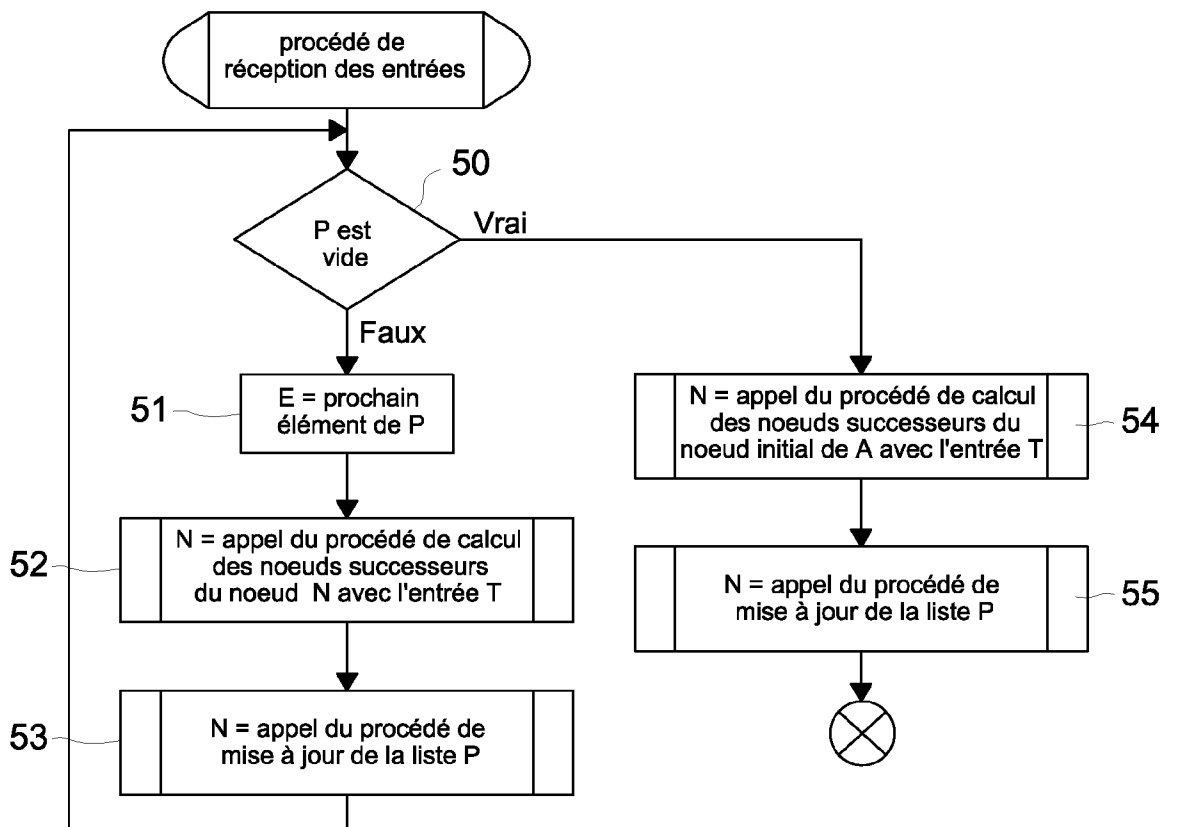


FIG.15

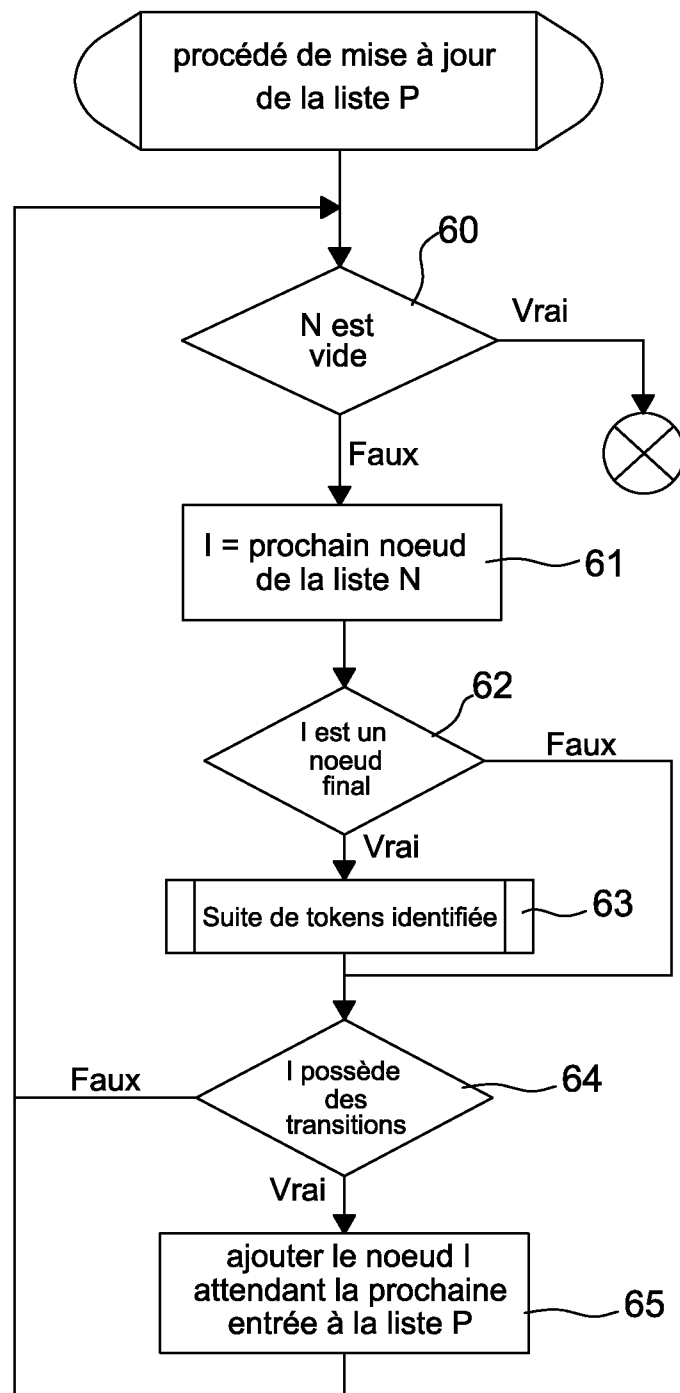


FIG.16

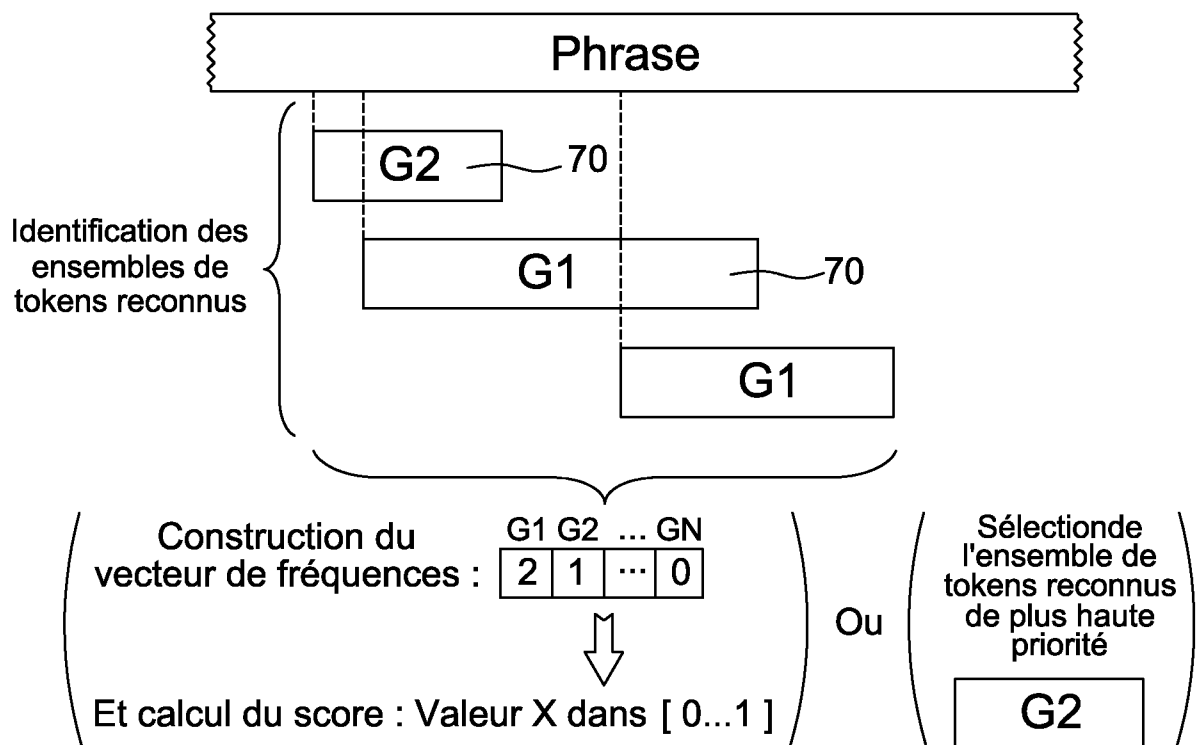


FIG.17

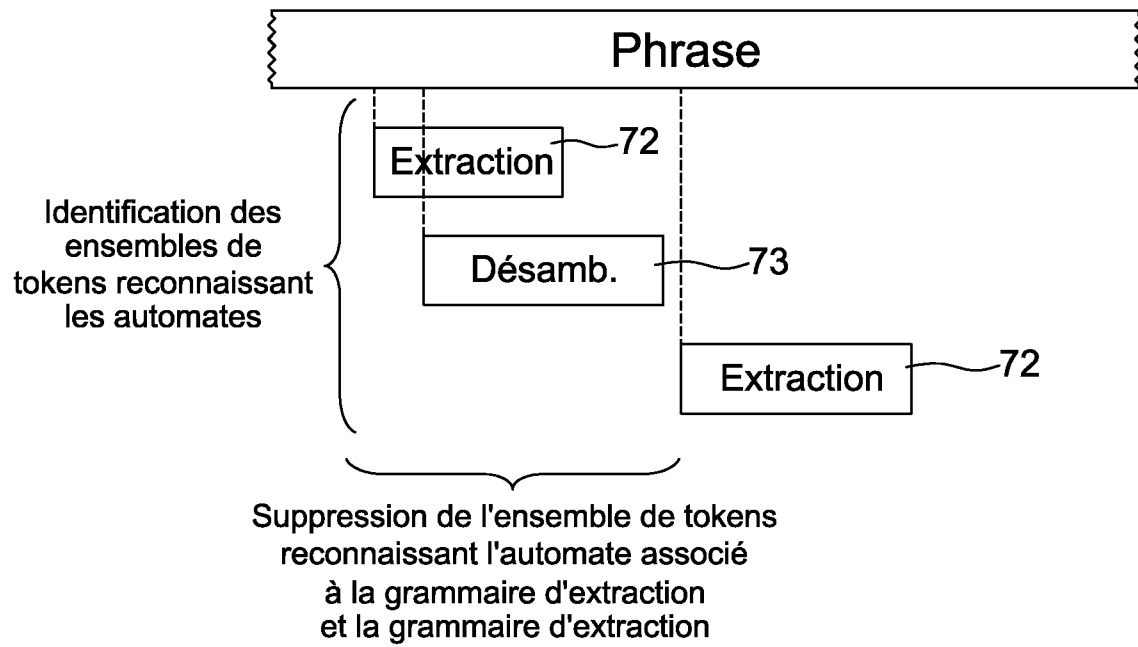


FIG.18

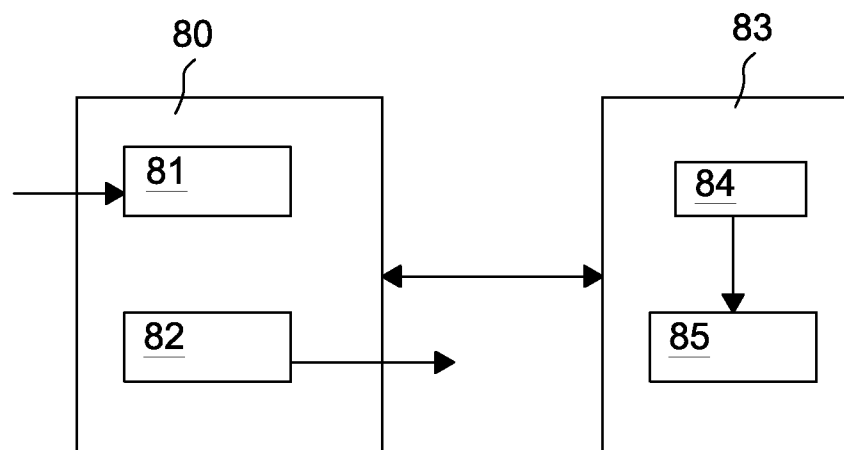


FIG.19

INTERNATIONAL SEARCH REPORT

International application No

PCT/EP2007/050569

A. CLASSIFICATION OF SUBJECT MATTER

INV. G06F17/27

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)

G06F

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practical, search terms used)

EPO-Internal, WPI Data, INSPEC, COMPENDEX, IBM-TDB

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	ARNAUD ADANT: "Study and implementation of a weighted finine-state library - application to speech synthesis" [Online] 2000, , TCTS LABS, FACULTÉ POLYTECHNIQUE DE MONS , XP002407725 Retrieved from the Internet: URL:http://membres.lycos.fr/adant/tfe/tfe.pdf> [retrieved on 2006-11-16] page 2 page 4; figure 1.1 sequence 2 sequence 4.3 sequence 5.2 page 59 ----- -/--	1-10

☒ Further documents are listed in the continuation of Box C.

☒ See patent family annex.

* Special categories of cited documents :

"A" document defining the general state of the art which is not considered to be of particular relevance

"E" earlier document but published on or after the international filing date

"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)

"O" document referring to an oral disclosure, use, exhibition or other means

"P" document published prior to the international filing date but later than the priority date claimed

"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art.

"&" document member of the same patent family

Date of the actual completion of the international search

2 March 2007

Date of mailing of the international search report

13/03/2007

Name and mailing address of the ISA/

European Patent Office, P.B. 5818 Patentlaan 2
NL - 2280 HV Rijswijk
Tel. (+31-70) 340-2040, Tx. 31 651 epo nl,
Fax: (+31-70) 340-3016

Authorized officer

Lopez, Patrice

INTERNATIONAL SEARCH REPORT

International application No

PCT/EP2007/050569

C(Continuation). DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	S. KANTHAK, H. NEY: "FSA: An Efficient and Flexible C++ Toolkit for Finite State Automata Using On-Demand Computation" [Online] July 2004 (2004-07), ACL , PROCEEDINGS OF THE 42ND ANNUAL MEETING OF THE ASSOCIATION FOR COMPUTATIONAL LINGUISTICS (ACL 2004), BARCELONA, SPAIN , XP002407726 Retrieved from the Internet: URL:http://www-i6.informatik.rwth-aachen.de/PostScript/InterneArbeiten/kanthak_acl2004.pdf> [retrieved on 2006-11-16] abstract sequence 3	1-10
X	E. ROCHE: "Parsing with Finite-State Transducers" [Online] November 1996 (1996-11), MITSUBISHI ELECTRIC RESEARCH LABORATORIES , MERL TECHNICAL REPORT TR-96-30, CAMBRIDGE, MA, USA , XP002407727 Retrieved from the Internet: URL:http://citeseer.ist.psu.edu/rd/60346150%2C76907%2C1%2C0.25%2CDownload/http://citeseer.ist.psu.edu/cache/papers/cs/3645/http:zSzzSzwww.ling.gu.sezSz%7EsylvanazSzPostscriptszSzPPzSzTR96-30.pdf/roche97parsing.pdf> [retrieved on 2006-11-16] abstract sequence 4	1-10
A	WO 01/29697 A1 (SONY ELECTRONICS INC [US]) 26 April 2001 (2001-04-26) figures 3,4 page 13, line 13 - page 18, line 31	1-10

Information on patent family members

PCT/EP2007/050569

nm PCT/ISA/210 (patent family annex) (April 2005)

RAPPORT DE RECHERCHE INTERNATIONALE

Demande internationale n°

PCT/EP2007/050569

A. CLASSEMENT DE L'OBJET DE LA DEMANDE

INV. G06F17/27

Selon la classification internationale des brevets (CIB) ou à la fois selon la classification nationale et la CIB

B. DOMAINES SUR LESQUELS LA RECHERCHE A PORTE

Documentation minimale consultée (système de classification suivi des symboles de classement)

G06F

Documentation consultée autre que la documentation minimale dans la mesure où ces documents relèvent des domaines sur lesquels a porté la recherche

Base de données électronique consultée au cours de la recherche internationale (nom de la base de données, et si cela est réalisable, termes de recherche utilisés)

EPO-Internal, WPI Data, INSPEC, COMPENDEX, IBM-TDB

C. DOCUMENTS CONSIDERES COMME PERTINENTS

Catégorie*	Identification des documents cités, avec, le cas échéant, l'indication des passages pertinents	no. des revendications visées
X	ARNAUD ADANT: "Study and implementation of a weighted finite-state library - application to speech synthesis" [Online] 2000, , TCTS LABS, FACULTÉ POLYTECHNIQUE DE MONS , XP002407725 Extrait de l'Internet: URL: http://membres.lycos.fr/adant/tfe/tfe.pdf [extrait le 2006-11-16] page 2 page 4; figure 1.1 séquence 2 séquence 4.3 séquence 5.2 page 59 ----- -/--	1-10

☒ Voir la suite du cadre C pour la fin de la liste des documents

☒ Les documents de familles de brevets sont indiqués en annexe

* Catégories spéciales de documents cités:

- "A" document définissant l'état général de la technique, non considéré comme particulièrement pertinent
- "E" document antérieur, mais publié à la date de dépôt international ou après cette date
- "L" document pouvant jeter un doute sur une revendication de priorité ou cité pour déterminer la date de publication d'une autre citation ou pour une raison spéciale (telle qu'indiquée)
- "O" document se référant à une divulgation orale, à un usage, à une exposition ou tous autres moyens
- "P" document publié avant la date de dépôt international, mais postérieurement à la date de priorité revendiquée

"T" document ultérieur publié après la date de dépôt international ou la date de priorité et n'appartenant pas à l'état de la technique pertinent, mais cité pour comprendre le principe ou la théorie constituant la base de l'invention

"X" document particulièrement pertinent; l'invention revendiquée ne peut être considérée comme nouvelle ou comme impliquant une activité inventive par rapport au document considéré isolément

"Y" document particulièrement pertinent; l'invention revendiquée ne peut être considérée comme impliquant une activité inventive lorsque le document est associé à un ou plusieurs autres documents de même nature, cette combinaison étant évidente pour une personne du métier

"&" document qui fait partie de la même famille de brevets

Date à laquelle la recherche internationale a été effectivement achevée

2 mars 2007

Date d'expédition du présent rapport de recherche internationale

13/03/2007

Nom et adresse postale de l'administration chargée de la recherche internationale

Office Européen des Brevets, P.B. 5818 Patentlaan 2
NL - 2280 HV Rijswijk
Tel. (+31-70) 340-2040, Tx. 31 651 epo nl,
Fax: (+31-70) 340-3016

Fonctionnaire autorisé

Lopez, Patrice

RAPPORT DE RECHERCHE INTERNATIONALE

Demande internationale n°

PCT/EP2007/050569

C(suite), DOCUMENTS CONSIDERES COMME PERTINENTS

Catégorie*	Identification des documents cités, avec, le cas échéant, l'indication des passages pertinents	no. des revendications visées
X	S. KANTHAK, H. NEY: "FSA: An Efficient and Flexible C++ Toolkit for Finite State Automata Using On-Demand Computation" [Online] juillet 2004 (2004-07), ACL , PROCEEDINGS OF THE 42ND ANNUAL MEETING OF THE ASSOCIATION FOR COMPUTATIONAL LINGUISTICS (ACL 2004), BARCELONA, SPAIN , XP002407726 Extrait de l'Internet: URL:http://www-i6.informatik.rwth-aachen.de/PostScript/InterneArbeiten/kanthak_acl2004.pdf> [extrait le 2006-11-16] abrégé séquence 3	1-10
X	E. ROCHE: "Parsing with Finite-State Transducers" [Online] novembre 1996 (1996-11), MITSUBISHI ELECTRIC RESEARCH LABORATORIES , MERL TECHNICAL REPORT TR-96-30, CAMBRIDGE, MA, USA , XP002407727 Extrait de l'Internet: URL:http://citeseer.ist.psu.edu/rd/60346150%2C76907%2C1%2C0.25%2CDownload/http://citeseer.ist.psu.edu/cache/papers/cs/3645/http:zSzzSzwww.ling.gu.sezSz%7EsylvanazSzPostscriptszSzPPzSzTR96-30.pdf/roche97parsing.pdf> [extrait le 2006-11-16] abrégé séquence 4	1-10
A	WO 01/29697 A1 (SONY ELECTRONICS INC [US]) 26 avril 2001 (2001-04-26) figures 3,4 page 13, ligne 13 - page 18, ligne 31	1-10

RAPPORT DE RECHERCHE INTERNATIONALE

Renseignements relatifs aux membres de familles de brevets

Demande internationale n°

PCT/EP2007/050569

Document brevet cité au rapport de recherche		Date de publication	Membre(s) de la famille de brevet(s)	Date de publication
WO 0129697	A1	26-04-2001	AU 2466901 A	30-04-2001
			US 6721697 B1	13-04-2004
			US 2004167771 A1	26-08-2004