Résumés de données pour la personnalisation de requêtes
Personalized database querying using data summaries

Laurent UGHETTO W. Amenel VOGLOZIN Noureddine MOUADDIB

LINA – Université de Nantes
2 rue de la Houssinière, BP44322, NANTES Cedex 3, FRANCE
laurent.ughetto@univ-nantes.fr, amenel.voglozin@univ-nantes.fr, noureddine.mouaddib@univ-nantes.fr

Résumé :

La surcharge d'informations et les temps de réponse aux requêtes augmentent conjointement avec la taille et la distribution des bases de données. Il apparaît aussi que les systèmes d'interrogation tiennent insuffisamment compte des caractéristiques de l'utilisateur. Dans ce contexte, cet article étudie la possibilité de combiner l'utilisation de résumés de données et de la personnalisation dans un processus d'interrogation. La personnalisation consiste ici à laisser l'utilisateur définir son vocabulaire d'interrogation par le biais de variables linguistiques. L'utilisation de résumés est destinée à accroître l'efficacité du requêtage. Plusieurs façons de les combiner sont présentées, accompagnées d'algorithmes.

Mots-clés :

Interrogation des bases de données, Personnalisation, Résumé de données

Abstract:

The increasing size of distributed data sources often leads to large amounts of answers to user queries (information overload), and long response times. Moreover, querying systems do not sufficiently take into account and make use of the user-related information. This paper discusses the possibility and implications of combining the use of summaries with a personalized service, in the querying of a database. The personalization aspect consists in allowing the user to choose his own querying vocabulary, by means of linguistic variables, while the use of summaries increases the efficiency of the querying. Several ways of doing so are identified and search algorithms for the different cases are proposed.

Keywords:

Database querying , Personnalization, Data summary

1 Introduction

Ces dernières années, la taille, la diversité et la distribution des bases de données ont crû très fortement. Une conséquence est que la quantité de données disponible dépasse largement notre capacité d'appréhension. Il est aussi de plus en plus difficile de trouver toute, et uniquement, l'information pertinente dans les grandes BD.

Les résumés de données (e.g. voir [3, 4, 7, 9]), et parmi eux SAINTETIQ [10], le système considéré ici, font partie des moyens de réponse à ce problème. L'utilisation de résumés accélère l'accès aux données et conduit à des réponses plus compactes, plus facilement appréhendables et, en contrepartie, moins précises. Le plus souvent, les systèmes existants produisent des résumés à exploiter « à la main », ce qui n'est pas envisageable que pour quelques résumés. C'est pourquoi des algorithmes d'interrogation des résumés ont été proposés pour SAINTETIQ [13].

Les systèmes de personnalisation, qui tendent à traiter chaque utilisateur de façon spécifique (en fonction de son profil), sont très présents sur le web (e.g. [6]), et leur cadre d'application s'étend. L'utilisation des intérêts, préférences et caractéristiques des utilisateurs, par l'ajout d'informations et de contraintes, permet aux systèmes d'interrogation de réduire à la fois l'espace de recherche et le nombre de sources interrogées (réduisant les temps de recherche), pour des réponses moins nombreuses et mieux ciblées (réduisant la surcharge d'informations).

Cet article essaie d'exploiter les deux approches, et de montrer l'intérêt des résumés de données dans le contexte des systèmes d'interrogation personnalisés. Dans cette approche, les utilisateurs peuvent définir et utiliser leur propre vocabulaire (au lieu du domaine des attributs). Ce vocabulaire est défini à l'aide de variables linguistiques, stockées dans leur profil, et leur permet de définir la granularité souhaitée aussi bien dans les requêtes que les réponses. Les résumés

Cette recherche a été partiellement soutenue par le Ministère Délégué à la Recherche et aux Nouvelles Technologies, dans le programme ACI Masses de Données, projet #MD-33.

de données sont utilisés pour améliorer l'efficacité du processus d'interrogation.
Cette approche étend un travail antérieur sur l'interrogation flexible, et les outils d'utilisation des résumés ([12, 14]). Des algorithmes d'interrogation utilisant le vocabulaire prédéterminé par SAINTETIQ ont déjà été proposés [13]. Ils ne permettent pas l'utilisation d'un vocabulaire différent, contrairement aux systèmes d'interrogation flexible tels que SQLf [2] ou FQuery [8]. Les algorithmes de [13] sont étendus ici à l'utilisation d'un vocabulaire spécifique à l'utilisateur, et défini dans son profil.
La section 2 détaille le type de résumé considéré ici, puis présente brièvement le modèle SAINTETIQ. La section 3 donne des applications possibles des résumés dans les systèmes de personnalisation. Enfin, la section 4 propose des algorithmes d'interrogation.

2 Résumés de données

2.1 Résumer par l'imprécision

Les résumés de données considérés ici sont construits par l'introduction d'imprécision. Le principe est simple. Prenons une BD dont les attributs sont définis sur des domaines « précis », comme par exemple l'ensemble des entiers pour l'attribut `âge`. La technique de résumé consiste à modifier le domaine initial en un nouveau domaine plus grossier ou plus imprécis, par exemple {`bébé`, `enfant`, `adolescent`, `jeune adulte`, `adulte`, `senior`} : 6 valeurs au lieu d'une bonne centaine. La BD est réécrite avec ce nouveau « vocabulaire ». Les tuples devenus indiscernables sont groupés en un seul, plus imprécis, appelé « résumé ».
Ainsi, tous les tuples initiaux sont représentés dans les résumés (e.g., contrairement aux résumés par valeurs typiques), mais de façon plus grossière.
Le choix des nouveaux domaines d'attributs est une étape clé du processus de résumé. Dans le contexte de la personnalisation, ceux-ci doivent être signifiants pour l'utilisateur. Ils ne sont donc pas choisis par rapport aux données, à la différence des processus de classification.
Il est naturel de réécrire un domaine ordonné ou continu à l'aide d'intervalles réalisant une partition de ce domaine, mais ceux-ci entraînent un effet de seuil. Si cet effet est parfois intrinsèque au domaine et doit être conservé (e.g. on a le droit de vote à 18 ans), il peut être évité par l'utilisation de variables linguistiques [15].
La granularité (i.e., le nombre de termes) des variables linguistiques choisies détermine à la fois le ratio et la précision des résumés. Plus elle est grossière, plus le résumé est compact et imprécis. Plus elle est fine, moins le résumé est compact et plus il est précis. Ce niveau de granularité doit donc dépendre des besoins des utilisateurs. Il est intéressant de définir plusieurs niveaux de granularité lors du résumé d'une BD. Ces niveaux peuvent être obtenus pas des thésaurus ou par regroupement hiérarchique de valeurs d'attributs.

2.2 Le modèle SAINTETIQ

Les caractéristiques décrites ci-dessus sont implémentées dans le prototype de résumé de données SAINTETIQ ([10], [11]), qui utilise un algorithme de classification incrémental et produit des résumés structurés de façon hiérarchique. Le processus de construction, en deux étapes, est décrit sur un exemple.
Soit la relation R = (thickness, hardness, temperature) représentée par la table `MATERIALS`. Un tuple de R décrit un matériau d'une usine métallurgique imaginaire qui produit des feuilles. L'attribut `thickness`, en mm, a pour domaine les réels (de 0.15 à 50). `hardness` est la dureté souhaitée du produit fini, sur l'échelle de dureté B de Rockwell. `temperature` est le point de fusion en degrés Celsius. Les variables linguistiques associées aux différents attributs de R sont données sur la figure 1.

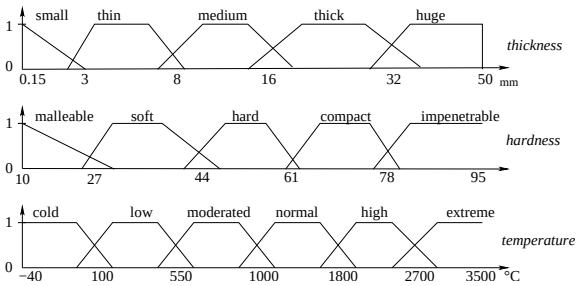

Figure 1 – *Var. linguistiques pour la table* `MATERIALS`

Supposons que la table MATERIALS contient 5 tuples : $t_a = \langle 10, 38, 900 \rangle$ (CuZn40), $t_b = \langle 8, 40, 850 \rangle$ (CuSn12), $t_c = \langle 12, 44, 896 \rangle$ (CuAs05), $t_d = \langle 19, 35, 1530 \rangle$ (Fe) et $t_e = \langle 5, 35, 1453 \rangle$ (Ni).

Matériau	Tuples candidats
CuZn40	$t_{a_1} =$ ⟨ 0.7/medium, 1.0/soft, 0.85/moderated ⟩
CuSn12	$t_{b_1} =$ ⟨ 0.3/medium, 0.9/soft, 1.0/moderated ⟩ $t_{b_2} =$ ⟨ 0.4/thin, 0.9/soft, 1.0/moderated ⟩
CuAs05	$t_{c_1} =$ ⟨ 0.6/medium, 0.35/soft, 0.9/moderated ⟩ $t_{c_2} =$ ⟨ 0.4/thick, 0.45/hard, 0.9/moderated ⟩
Fe	$t_{d_1} =$ ⟨ 0.8/thick, 1.0/soft, 0.85/normal ⟩
Ni	$t_{e_1} =$ ⟨ 1.0/thin, 1.0/soft, 0.96/normal ⟩

Figure 2 – *Traduction des tuples de* MATERIALS

La première étape consiste à réécrire les tuples en utilisant les variables linguistiques définies dans les connaissances de domaine (cf. table 2). Les tuples réécrits sont appelés *tuples candidats*. Lorsqu'une valeur peut être réécrite par plusieurs termes (e.g., *8 mm* est décrit à la fois par medium et thin), on obtient autant de tuples candidats (e.g., t_{b_1} et t_{b_2} dans la table 2).

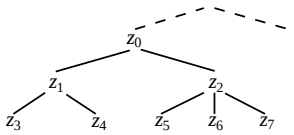

Figure 3 – *Hiérarchie de résumés de* MATERIALS

Au cours de la 2e étape, chaque tuple candidat est incorporé à un arbre de résumés où il va s'ajouter à une feuille (cf. figure 3). C'est une sorte de classification du tuple. Le processus de construction (peu pertinent ici) n'est pas rappelé ; il est détaillé dans [10]. Notons qu'il s'agit d'un processus incrémental qui adapte la forme de l'arbre de résumés au cours de sa construction (pour limiter l'effet d'ordre).

Dans la structure hiérarchique, un nœud résume une partie des données. Plus on descend dans la hiérarchie, plus la description est précise et moins elle représente de données. Ainsi, les feuilles sont les résumés les plus précis : ils sont décrits par un seul label par attribut (e.g. z_3 dans la table 4). Les nœuds intermédiaires sont un simple regroupement des nœuds fils, et sont donc décrits avec plusieurs labels sur au moins un attribut (e.g. z_1 et z_2). Ces labels sont l'union des

Résumé	Intension
z_3	⟨ 1.0/medium, 1.0/soft, 1.0/moderated ⟩
z_4	⟨ 0.8/thick, 1.0/soft, 0.85/normal ⟩
z_5	⟨ 0.4/thin, 0.9/soft, 1.0/moderated ⟩
z_6	⟨ 1.0/medium, 0.45/hard, 0.9/moderated ⟩
z_7	⟨ 1.0/thin, 1.0/soft, 0.96/normal ⟩
z_1	⟨ 1.0/medium + 0.8/thick, 1.0/soft, 1.0/moderated + 0.85/normal ⟩
z_2	⟨ 1.0/thin + 1.0/medium, 1.0/soft + 0.45/hard, 1.0/moderated + 0.96/normal ⟩
z_0	⟨ 1.0/thin + 1.0/medium + 0.8/thick, 1.0/soft + 0.45/hard, 1.0/moderated + 0.96/normal ⟩

Figure 4 – *Description de quelques résumés*

labels des fils. La racine est le résumé le plus général. Il représente toutes les données.

Les feuilles de l'arbre généré par SAINTETIQ contiennent aussi la référence des tuples de la BD qu'elles résument, ce qui confère à l'arbre des propriétés d'index multidimensionnel.

2.3 Pourquoi interroger des résumés ?

Si les résumés de bases de données sont un moyen de réduire le volume des données interrogées, et donc les temps de réponse, ce gain se fait aux dépens de la précision des réponses. Ce n'est pas gênant lorsqu'une réponse grossière suffit. Par exemple, pour anticiper la contamination par la grippe aviaire et les mesures à mettre en place, on peut vouloir estimer « *combien* de *gros* élevages se trouvent *près* des routes migratoires dans ma zone d'intervention ». La réponse souhaitée est peu, moyennement ou beaucoup. Les lieux exacts, le nombre de volailles ... sont des informations inutiles et parasites.

La précision est inutile lorsque la requête a pour seul but de savoir s'il existe des réponses. Avec des sources de données distribuées, on peut adresser ce genre de *pré-requêtes* pour n'interroger, in fine, que les sources susceptibles d'apporter une réponse [1].

L'imprécision est même parfois souhaitable. Par exemple, lors du traitement statistique d'informations médicales, des données trop précises pourraient violer le secret médical.

Lorsque des réponses précises sont attendues,

l'interrogation des résumés peut rester utile. En effet, les tuples de la base initiale peuvent être obtenus à partir des résumés réponses. Le mécanisme de requête reste efficace, sans perte de précision, car l'arbre des résumés fonctionne alors comme un index multidimensionnel.
Enfin, les résumés sont intéressants dans le contexte de l'interrogation flexible, et en particulier lorsque les résumés et les requêtes partagent les mêmes termes linguistiques.

3 Interrogation de résumés dans le contexte de la personnalisation

La section précédente a montré l'intérêt d'interroger des résumés, en particulier lorsque le vocabulaire d'expression des requêtes et des réponses est plus grossier, moins précis que le domaine des attributs de la base de données.
Du point de vue de la personnalisation, chaque utilisateur peut définir son propre vocabulaire, à l'aide de variables linguistiques (stockées dans son *profil*) qui mettent en correspondance les termes (le nouveau vocabulaire) et les valeurs du domaine (voir figure 1). Une première idée simple consiste alors à créer, avec ce vocabulaire, un résumé de la base de données, stocké également dans le profil de l'utilisateur. Cette idée et ses limites sont discutées ci-après.

3.1 Interrogation de résumés personnalisés

Supposons qu'un utilisateur a défini son vocabulaire d'interrogation. Lorsqu'il pose une requête, celle-ci doit être réécrite avec le vocabulaire de la base, la base doit être interrogée, puis la réponse réécrite à nouveau dans le vocabulaire utilisateur. La méthode est inutilement coûteuse cat la première réécriture (peu coûteuse en elle-même) implique un niveau de finesse trop important par rapport à celui de la requête et des réponses.
Pour gagner en temps de réponse, et éviter une précision superflue dans le mécanisme de requête, on peut utiliser des résumés du type de SAINTETIQ. Supposons qu'un résumé de la BD soit construit en utilisant les variables linguistiques de l'utilisateur. On peut alors utiliser les algorithmes d'interrogation de [13, 14]. Ils sont particulièrement efficaces car ils n'impliquent que des opérateurs booléens (bien que manipulant des ensembles flous), et tirent avantage de la structure hiérarchique des résumés.
Parmi les avantages de cette approche, il y a donc la possibilité pour chaque utilisateur de définir son vocabulaire, d'obtenir des réponses très rapides, au niveau de précision souhaité.
Le nombre de résumés à construire et à mémoriser (un par utilisateur), est un premier inconvénient. Si le nombre d'utilisateurs est faible, les résumés peuvent être gérés par le SGBD. Sinon, ils devront plutôt être délocalisés dans le profil utilisateur, ce qui n'est faisable que pour des résumés de petite taille. Pour de très grandes BD et de nombreux utilisateurs, des solutions de stockage spécifiques restent à définir.
Le temps de construction des résumés est un autre inconvénient. Même avec un processus incrémental, de complexité polynomiale, la construction est coûteuse. Si, du point de vue utilisateur, on peut anticiper la construction du résumé, du point de vue système, construire et mettre à jour un résumé pour chaque utilisateur potentiel n'est pas envisageable. Ainsi, la solution qui consiste à construire un résumé pour chaque utilisateur, bien qu'idéale pour l'interrogation, n'est pas toujours envisageable à grande échelle.
Dans les sections suivantes, deux solutions alternatives sont discutées. La première consiste à utiliser des profils de groupe pour réduire le nombre de résumés à gérer. La seconde consiste à générer un seul arbre de résumés, avec un vocabulaire prédéfini, et à l'interroger avec un vocabulaire différent, propre à chaque utilisateur.

3.2 Résumer selon des profils de groupe

Pour réduire la complexité spatiale et temporelle, il faut réduire le nombre de résumés construits, stockés et mis à jour. Comme leur nombre ne peut être réduit, les utilisateurs doivent partagent des résumés communs. En conséquence, on ne peut plus permettre à chaque utilisateur de définir ses propres variables linguistiques.
L'idée est d'utiliser des *profils de groupe*, c'est-à-dire des profils partagés par des groupes d'utilisateurs. Un nombre limité de profils est défini,

pour des classes d'utilisateurs, avec des variables linguistiques choisies avec soin pour chaque classe. Le SGBD, ou un médiateur, doit alors créer et maintenir les résumés correspondants. Chaque utilisateur *souscrit* au groupe dont il partage le profil.

Le nombre de groupes peut être défini en fonction des besoins et de la capacité du système.

Parmi les avantages de cette solution, on retrouve bien sûr le nombre limité (et contrôlé) de résumés. De plus, le vocabulaire d'interrogation reste celui de construction des résumés.

L'inconvénient principal est que l'utilisateur n'utilise pas son propre vocabulaire, mais celui d'un groupe. Cette solution est acceptable lorsque la notion de groupe est pertinente, i.e., lorsque les classes d'utilisateurs partagent de fait le même vocabulaire. Dans d'autres cas, où les groupes sont un peu artificiels, chaque utilisateur doit *apprendre* le vocabulaire du groupe avant de l'utiliser. Au pire, ce n'est pas plus facile ni plus pertinent que d'apprendre un vocabulaire ad hoc. Dans ce cas, les profils de groupe ne sont pas une solution, car un des buts de la personnalisation (telle qu'envisagée ici) est que l'utilisateur puisse « parler son propre langage » et ne pas avoir à apprendre celui des autres.

4 Interrogation d'un résumé avec un vocabulaire personnalisé

Si on souhaite que chaque utilisateur puisse utiliser son vocabulaire, sans pour autant lui construire un résumé spécifique, il faut pouvoir interroger des résumés de données avec un vocabulaire différent de celui qui a servi à le construire. C'est l'objet des algorithmes proposés dans cette section. On considère donc une seule hiérarchie de résumés, construite avec un vocabulaire prédéfini mais interrogée avec celui de chaque utilisateur.

4.1 Requêtes précises

Considérons tout d'abord le cas de requêtes *précises*, i.e. exprimées avec des valeurs précises, issues des domaines initiaux des attributs. Par exemple : « Quelle est la dureté des matériaux dont le point de fusion est 1530˚C ».

Le point principal est qu'un résumé ne peut donner une information plus précise que celle qu'il contient. Une valeur précise doit donc être réécrite en un terme linguistique. Ici, 1530 est réécrit en `normal` (cf. figure 1).

Supposons que le système (interrogé grâce aux algorithmes de [12]) donne la réponse disjonctive (`soft, hard`) à la requête réécrite. Dans un premier temps, la réponse ne peut être que « les matériaux dont le point de fusion est `normal` sont `soft` ou `hard` ». Suivant la sémantique des résumés de SAINTETIQ, cela signifie qu'il y a dans la base de données des matériaux dont le couple (`hardness`, `temperature`) a pour valeur (`soft`, `normal`) et d'autres qui ont pour valeur (`hard`, `normal`). Le nombre de matériaux de chaque type est aussi fourni par les résumés. Toutefois, rien n'assure qu'il y ait des matériaux dont le point de fusion est 1530. Ce n'est pas surprenant puisque le résumé se fait par introduction d'imprécision.

Si une réponse précise est nécessaire, on peut retrouver les tuples représentés par les résumés (`soft`, `normal`) et (`hard`, `normal`), puisque les références à ces tuples sont contenues dans les résumés. En filtrant cet ensemble réduit de tuples avec le critère initial (`temperature` = 1530), on obtient la réponse qu'on aurait obtenue par une requête classique.

Il peut sembler étrange d'utiliser des résumés lorsqu'on attend des réponses précises à des requêtes précises. Toutefois, si les résumés existent, il y a au moins deux avantages à les interroger. En cas de réponse vide, cette réponse est obtenue très rapidement, et des tuples *proches* de ceux requis peuvent même être proposés (voir [13]). Et dans les autres cas, la hiérarchie de résumés utilisée dans les algorithmes fonctionne comme un index multidimensionnel. Son efficacité en tant qu'index est encore en cours d'évaluation, mais semble intéressante pour un système qui n'a pas été conçu pour cela.

Cette procédure conduit à l'algorithme 1, dans lequel les informations comme « valeur précise requise » ou « réparer les requêtes sans réponse » peuvent faire partie du profil de l'utilisateur.

Dans la réécriture de la requête initiale, une va-

Algorithm 1 Interrogation avec une requête précise

Input: QueryInit // *requête initiale de l'utilisateur*
Input: SummTree // *arbre de résumés de* SAINTETIQ
Input: Profile // *profil de l'utilisateur*
  // *Réécrire la requête avec les var. ling. du profil*
  Q ← Rewrite(QueryInit, Profile.LingVar)
  // *Interroger l'arbre de résumés avec les algos de [12]*
  AnsSet ← SummQueryingAlg(SummTree, Q)
  // *Traiter les requêtes sans réponse*
  if AnsSet == {} AND Profile.RepairNullQuery == yes **then**
    // *Utiliser l'algo de réparation des requêtes de [13]*
    AnsSet ← SummQueryingRepair(SummTree, Q)
  end if
  // *Traiter l'imprécision*
  if Profile.Precision == yes **then**
    // *Retrouver les tuples à partir des résumés réponses*
    PrecAnsSet ← {}
    for all S in AnsSet **do**
      PrecAnsSet ← PrecAnsSet ∪ TuplesFromSumm(S)
    end for
    // *Filtrer les tuples par rapport à la requête initiale*
    PrecAnsSet ← FilterOut(PrecAnsSet, QueryInit)
  end if
Output: AnsSet // *résumés réponses (peut être vide)*
Output: PrecAnsSet // *tuples réponse (si requis)*

leur précise peut parfois être réécrite de plusieurs façons. Dans ce cas, il est suffisant de la réécrire avec un des termes seulement (n'importe lequel) car, par construction des résumés, un tuple pouvant être réécrit de plusieurs façons est référencé dans tous les résumés correspondants.

4.2 Requêtes imprécises - calcul exact

Prenons maintenant une requête *imprécise*, i.e. exprimée avec des termes linguistiques. Suivant les préférences de l'utilisateur, les réponses peuvent être précises (des tuples de la base) ou imprécises, écrites avec les termes linguistiques de l'utilisateur. Cette section montre comment l'algorithme 1 peut être adapté aux requêtes imprécises, avec à un calcul exact des réponses. Ensuite, des algorithmes plus rapides, mais effectuant seulement un calcul approché des réponses sont proposés.

Comme précédemment, l'algorithme consiste à obtenir le plus petit ensemble des résumés qui représente tous les tuples réponses. Ainsi, la majeure partie de l'algorithme 1 est inchangée. Seuls la réécriture de la requête et le filtrage des réponses sont modifiés. Chaque terme de la requête est réécrit par un sur-ensemble (au sens de l'inclusion floue $A \subseteq B$ ssi $\forall x,\ \mu_A(x) \leq \mu_B(x)$) de termes du vocabulaire des résumés. Un exemple de réécriture est donné sur la figure 5. Tous les tuples réponses seront capturés par les résumés réponses obtenus. Il n'y aura pas de faux-négatifs (tuple réponse non capturé par un résumé réponse). Les tuples sont retrouvés à partir des résumés réponses, et filtrés par rapport à la requête initiale. Ensuite, en fonction du choix de l'utilisateur, ils sont donnés tels quels, ou réécrits à l'aide du vocabulaire utilisateur. Cette procédure est décrite par l'algorithme 2.

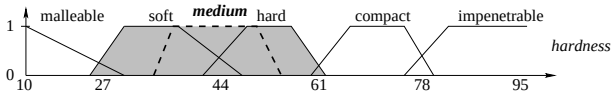

Figure 5 – Réécriture de **medium** en "soft ou hard".

En fait, la dernière réécriture est un processus de résumé (SAINTETIQ peut être utilisé, par exemple sous la forme du web-service décrit dans [11]). Elle permet à l'utilisateur d'adapter la granularité du vocabulaire de la réponse à ses besoins. Plus simplement, on peut réécrire chaque tuple avec le terme linguistique le plus représentatif, comme sur l'exemple (en une seule dimension) de la figure 6. Cela revient à considérer seulement les feuilles de l'arbre généré par SAINTETIQ.

Les réponses obtenues sont celles qu'auraient donné les algorithmes de [12] avec des résumés construits à l'aide du vocabulaire utilisateur.

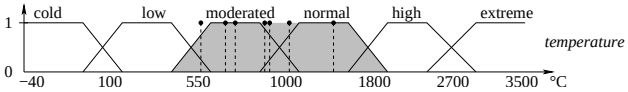

Figure 6 – Les 7 tuples réponses sont résumés par "`moderated` ou `normal`".

L'inconvénient principal de cette méthode est le temps de calcul. Pour obtenir des réponses en *langage utilisateur* à partir des réponses en *langage des résumés*, on doit passer par les tuples de la base de données initiale. La différence entre cette méthode et un système de requêtes personnalisées réside dans l'utilisation des résumés comme index multidimensionnel, puisque les deux méthodes font sensiblement les mêmes réécritures avant et après interrogation de la base.

Les réécritures et filtrages ne sont donc pas induits par l'utilisation des résumés, mais inhérents à la personnalisation du vocabulaire utilisé pour les requêtes et les réponses. Comme avantage, le mécanisme de résumé peut être utilisé pour le reformattage des réponses avec les termes de l'utilisateur.

Algorithm 2 interrogation exacte avec vocabulaire utilisateur

Input: QueryInit // *requête initiale de l'utilisateur*
Input: SummTree // *arbre de résumés de* SAINTETIQ
Input: Profile // *profil de l'utilisateur*
// *Réécrire la requête avec les var. ling. du profil*
Q ← Rewrite(QueryInit, Profile.LingVar)
// *Interroger l'arbre de résumés avec les algos de [12]*
AnsSet ← SummQueryingAlg(SummTree, Q)
// *Retrouver les tuples à partir des résumés réponses*
PrecAnsSet ← {}
for all S in AnsSet **do**
PrecAnsSet ← PrecAnsSet ∪ TuplesFromSumm(S)
end for
// *Filtrer les tuples par rapport à la requête initiale*
PrecAnsSet ← FilterOut(PrecAnsSet, QueryInit)
// *Réécrire (résumer) les tuples réponse avec les termes de l'utilisateur*
FinalAnsSet ← Summarize(PrecAnsSet)

Output: FinalAnsSet // *ensemble de réponses exprimées dans le vocabulaire utilisateur*

4.3 Requêtes imprécises - calcul approché

Le passage par les tuples ne peut être évité pour un calcul *exact* de la réponse. Lorsqu'une estimation suffit, cette étape coûteuse peut être évitée, et un calcul approché peut être réalisé de deux façons, donnant aux résultats des sens différents. Tout d'abord, on peut partir de l'algorithme 2, mais une fois les résumés réponses obtenus, les réécrire directement avec les termes de l'utilisateur, ce qui donne l'algorithme 3.

Les résumés réponses représentant (au moins) tous les tuples réponses, chacun de leurs termes est réécrit par un sur-ensemble de termes de l'utilisateur. Cette réécriture se fait sans perdre de tuples réponses, mais conduit à plus d'imprécision qu'un calcul exact.

Un ensemble réponses (`FinalAnsSet`) vide garantit une réponse vide. Par contre, une réponse non vide veut seulement dire qu'il est *possible* que des tuples réponses existent (auquel cas ils sont dans les résumés réponses), mais ce n'est pas certain. Ceci est dû à la réécriture de la requête, qui a pu inclure des tuples faux positifs.

Les avantages de l'algorithme 3 sont son efficacité (la base initiale n'est pas interrogée), et l'utilisation du vocabulaire de l'utilisateur. Son inconvénient majeur, outre la perte de précision, est qu'il ne peut garantir que les réponses vides.

Algorithm 3 interrogation approchée par sur-ensemble

Input: QueryInit // *requête initiale de l'utilisateur*
Input: SummTree // *arbre de résumés de* SAINTETIQ
Input: Profile // *profil de l'utilisateur*
// *Réécrire la requête avec les var. ling. du profil*
Q ← Rewrite(QueryInit, Profile.LingVar)
// *Interroger l'arbre de résumés avec les algos de [12]*
AnsSet ← SummQueryingAlg(SummTree, Q)
// *Réécrire les résumés réponses avec les termes de l'utilisateur*
FinalAnsSet ← Rewrite(AnsSet, Profile.LingVar)

Output: FinalAnsSet // *ensemble de réponses exprimées dans le vocabulaire utilisateur*

Dans certains cas, il peut être plus intéressant de garantir l'existence de réponses, même si elles ne sont pas toutes capturées. Pour cela, il faut effectuer les réécritures différemment. Jusqu'ici, toutes les réécritures ont été effectuées *naturellement* de façon à ne perdre aucun tuple résultat potentiel (ajout éventuel de faux positifs, mais pas de faux négatifs supprimés). Pour garantir l'existence de tuples résultats, il faut procéder dans l'autre sens. La requête initiale doit donc être réécrite avec des sous-ensembles de termes linguistiques, comme sur la figure 7.

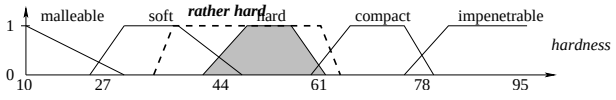

Figure 7 – le terme **rather hard** est réécrit par "hard".

La seconde réécriture est faite de la même façon. La réponse finale est donc un sous-ensemble de résumés exprimés dans le vocabulaire utilisateur. On peut garantir qu'ils contiennent (chacun) au moins un tuple réponse. En cas de réponse vide, on ne peut rien dire : des réponses peuvent exister quand même.

L'inconvénient de cet algorithme est de conduire à des réponses vides (inutilisables) lorsque la requête est exprimée avec des termes plus précis que ceux des résumés.

Notons aussi que les deux algorithmes approchés, peuvent être utilisés conjointement, l'un garantissant des réponses, et l'autre les réponses vides. Cette approche duale peut être vue comme un raisonnement avec à la fois des possibilités et des nécessités (comme dans [5]).

5 Conclusion

Cet article a discuté l'utilisation des résumés et de la personnalisation pour l'interrogation des BD. Ce travail peut être vu soit comme l'introduction des techniques de résumé dans l'interrogation flexible, soit comme l'introduction de la personnalisation dans l'interrogation des résumés. Sous le premier aspect, l'avantage des résumés est un gain sur les temps de réponse, soit par la propriété d'index des résumés, soit parce que la base initiale n'est pas interrogée. Sous le second aspect, on a montré que l'utilisation de termes linguistiques permet à l'utilisateur de définir un vocabulaire qui a pour lui du sens, et d'utiliser le niveau de granularité le plus adapté à ses besoins, tant dans la requête que dans l'expression des résultats.

Plusieurs méthodes ont été discutées. Du point de vue de l'interrogation, la méthode idéale consiste à interroger les résumés avec le vocabulaire qui a servi à leur création. Des algorithmes d'interrogation très efficaces existaient déjà pour ce cas. Mais on ne peut pas toujours créer et conserver un résumé par utilisateur. Deux alternatives ont été proposées. La première consiste à utiliser des profils de groupe, ce qui limite le nombre de résumés, mais limite aussi la personnalisation du vocabulaire. La deuxième consiste à construire un résumé avec un vocabulaire prédéfini, et à l'interroger avec un autre vocabulaire, celui de chaque utilisateur. Plusieurs algorithmes d'interrogation ont été proposés, permettant un calcul exact ou approché des résultats.

Ce travail n'est cependant qu'un premier pas dans l'interrogation des résumés avec des termes linguistiques quelconques. La prise en compte de la personnalisation, le choix des niveaux de granularité, … sont encore limités.

Enfin on a vu, à de nombreuses reprises, que la hiérarchie de résumés peut être utilisée comme un index multidimensionnel. La comparaison de SAINTETIQ avec de tels index est en cours.

Références

[1] B. Benatallah, M. Hassan, N. Mouaddib, H. Paik, and F. Toumani. Building and querying an e-catalog network using p2p and data summarisation techniques. *Int. J. of Intel. Info. Syst.*, 26(1) :7–24, 2006.

[2] P. Bosc and O. Pivert. Fuzzy queries and relational databases. In *Proc. of the ACM Symposium on Applied Computing*, pages 170–174, 1994.

[3] J. C. Cubero, J. M. Medina, O. Pons, and M. Amparo Vila Miranda. Data summarization in relational databases through fuzzy dependencies. *Information Sciences*, 121(3–4) :233–270, 1999.

[4] D. Dubois and H. Prade. Fuzzy sets in data summaries — outline of a new approach. In *Proc. 8th Int. Conf. on Information Processing and Managament of Uncertainty in Knowledge-based Systems (IPMU'2000)*, volume 2, pages 1035–1040, 2000.

[5] D. Dubois, H. Prade, and L. Ughetto. A new perspective on reasoning with fuzzy rules. *Int. J. of Intelligent Systems*, 18(5) :541–563, 2003.

[6] J. Fink and A. Kobsa. A review and analysis of commercial user modeling servers for personalization on the world wide web. *User Modeling and User-Adapted Interaction*, 10(2–3) :209–249, 2000.

[7] J. Kacprzyk. Fuzzy logic for linguistic summarization of databases. In *Proc. 8th Int. Conf. on Fuzzy Systems (FUZZ-IEEE'99)*, pages 813–818, 1999.

[8] J. Kacprzyk and S. Zadrożny. Computing with words in intelligent database querying : standalone and Internet-based applications. *Information Sciences*, 134 :71–109, 2001.

[9] D. H. Lee and M. H. Kim. Database summarization using fuzzy ISA hierarchies. *IEEE Trans. Syst., Man, Cybern. B*, 27 :68–78, 1997.

[10] G. Raschia and N. Mouaddib. SAINTETIQ : a fuzzy set-based approach to database summarization. *Fuzzy Sets and Systems*, 129 :137–162, 2002.

[11] R. Saint-Paul, G. Raschia, and N. Mouaddib. General purpose database summarization. In *Proc. 31st VLDB Conference*, pages 733–744, 2005.

[12] W. A. Voglozin, G. Raschia, L. Ughetto, and N. Mouaddib. Querying the SAINTETIQ summaries – a first attempt. In *Proc. 6th Int. Conf. on Flexible Query Ans. Syst. (FQAS'04)*, pages 404–417, 2004.

[13] W. A. Voglozin, G. Raschia, L. Ughetto, and N. Mouaddib. Querying the SaintEtiQ summaries – dealing with null answers. In *Proc. 14th Int. Conf. on Fuzzy Syst. (FUZZ-IEEE'05)*, pages 585–590, 2005.

[14] W. A. Voglozin, G. Raschia, L. Ughetto, and N. Mouaddib. Querying a summary of database. *Int. J. of Intel. Info. Syst.*, 26(1) :59–73, 2006.

[15] L.A. Zadeh. The concept of a linguistic variable and its application to approximate reasoning. *Information Sciences*, 1975. Part 1 : 8 :199-249 ; Part 2 : 8 :301-357 ; Part 3 : 9 :43-80.