Requêtes à Préférences et Bipolarité

Ludovic Liétard *, Daniel Rocacher **

** IUT – Université Rennes 1*
Département Informatique
Rue Edouard Branly BP 302191 Lannion

*** ENSSAT*
Rue de Kérampont BP 447 Lannion

ludovic.lietard@univ-rennes1.fr ; daniel.rocacher@enssat.fr

Section de rattachement : 27
Secteur : Secondaire

RÉSUMÉ. L'intégration des préférences utilisateurs dans des requêtes adressées à des bases de données relationnelles permet d'obtenir des réponses discriminées. Cet article étudie les différents langages (SQLf, PreferenceSQL) proposés pour exprimer des préférences dans les requêtes. L'accent est mis sur la typologie des préférences et sur son impact sur les résultats présentés à l'utilisateur. En particulier, on s'intéresse à l'hypothèse de commensurabilité et sur la bipolarité.

MOTS-CLÉS : langage de requêtes, préférence, bipolarité

1. Introduction

Le volume de l'information gérée par les systèmes de gestion de base de données (SGBD) devient de plus en plus important et, en conséquence, son interrogation se doit d'être de plus en plus performante. Cette performance peut être évaluée en termes de temps de réponse ou en termes de pertinence de la réponse délivrée. Dans ce dernier cas, l'intégration des préférences de l'utilisateur dans la requête est un moyen de personnaliser la recherche et d'obtenir des réponses plus adéquates. De telles requêtes impliquent la présentation à l'utilisateur d'un ensemble discriminé de réponses (des plus au moins préférées). Le problème de l'expression des préférences utilisateurs dans les requêtes a reçu beaucoup d'attention ces dernières années [Bosc et al. 1995, Bosc et al. 2008, Chomicki 2003, Kießling et al. 2002a, Kießling et al. 2002a, Lietard et al. 2008].

Lorsqu'un utilisateur définit ses préférences, il fait face à deux difficultés : 1) quel sens à donner aux préférences atomiques ? (par exemple comment exprimer des

préférences sur les prix ?), 2) comment agréger des préférences atomiques ? (comment combiner des préférences sur les prix avec celles sur des couleurs ?).

Dans cet article nous avons choisi le cadre où les préférences atomiques sont définies par des fonctions de scores. Cette représentation est la plus utilisée car la plus simple et la plus naturelle. En conséquence, une fonction délivrant des scores est définie pour chaque préférence, une valeur étant préférée à une autre si elle a obtenu un meilleur score.

Cet article considère le modèle relationnel de bases de données et il propose une classification des différentes techniques proposées pour l'intégration de préférences (définies par des scores) dans les requêtes. Notre motivation est de clarifier l'impact de ces techniques sur les réponses obtenues.

La section 2 présente la définition et la manipulation des préférences dans le contexte de requêtes utilisateurs. L'accent est mis sur l'hypothèse de commensurabilité. La section 3 définit la notion de bipolarité dans les préférences. La section 4 s'intéresse à deux langages de requêtes permettant l'expression de préférences (PreferenceSQL et SQLf). Ces deux langages sont situés par rapport à la commensurabilité et la bipolarité.

2. Préférences et commensurabilité

Un des objectifs d'une interrogation prenant en compte les préférences de l'utilisateur est la recherche et l'ordonnancement des réponses en fonction des préférences. Dans le contexte des bases de données relationnelles, des préférences élémentaires sont définies sur les domaines des attributs par des fonctions de score. Une valeur d'attribut est préférée à une autre si elle a obtenue un meilleur score. Par exemple, le score peut être une distance par rapport à une valeur optimale, plus la distance est petite, plus la valeur est préférée. La théorie des ensembles flous est également un outil permettant de définir des préférences élémentaires [Zade 65].

Lorsque plusieurs préférences sont impliquées dans une même requête, chaque tuple t_i est associé à un vecteur de scores $(s^1_i,...,s^n_i)$ où chaque s^j_i est l'évaluation d'une préférence sur l'attribut A_j du tuple t_i. La méthode utilisée pour comparer des tuples (des vecteurs de scores) dépend de la commensurabilité des scores élémentaires. Quand l'hypothèse de commensurabilité est retenue, cela signifie que les scores délivrés par des préférences différentes sont comparables et basés sur la même échelle. Dans l'autre cas, il n'est pas possible de comparer deux scores provenant de deux préférences différentes.

Les deux sous-sections suivantes détaillent les implications de cette hypothèse, en particulier sur les résultats présentés à l'utilisateur.

2.1. Préférences élémentaires commensurables

Si l'hypothèse de commensurabilité est vérifiée, les différents scores d'un même vecteur ont la même signification et ils peuvent être comparés les uns avec les autres. En conséquence, une fonction d'aggrégation *f* (telle qu'une moyenne, une moyenne pondérée, un minimum, …) peut être employée pour donner une évaluation globale de chaque vecteur. :

t_i est préféré à $t_j \Leftrightarrow f(s^1_i, \ldots, s^n_i) \geq f(s^1_j, \ldots, s^n_j)$.

La fonction *f* est en général une fonction numérique "ad-hoc" qui est définie par l'utilisateur. Evidemment, la fonction f doit prendre en compte les sens relatifs des scores afin que le mécanisme d'agrégation ait une signification. Un résultat important est qu'un ordre total est obtenu et, suite à une interrogation, un ensemble totalement ordonné de réponses est délivré à l'utilisateur.

Dans le cas particulier où les préférences sont définies par des ensembles flous [Zadeh 1965], les scores élémentaires sont des degrés dans [0, 1] et des opérateurs logiques étendus (norme et co-norme triangulaires, implications floues…) peuvent être employés ce qui garantit un cadre théorique fondé. On rappelle qu'un ensemble flou F défini sur un univers X est défini par une fonction d'appartenance μ_F de X vers l'intervalle [0,1] :

$$\mu_F : \quad X \rightarrow [0,1], \quad x \rightarrow \mu_F(x)$$

Le degré $\mu_F(x)$ est le degré indiquant dans quelle mesure la valeur *x* appartient à l'ensemble flou F. La fonction d'appartenance est une généralisation de la fonction caractéristique des ensembles et un ensemble ordinaire est un cas particulier d'ensemble flou (c'est un ensemble flou dont la fonction d'appartenance prend ses valeurs dans la paire {0, 1}). Les opérations définies sur les ensembles ordinaires ont été étendues aux esnembles floues et, en particulier, si *E* et *F* sont deux ensembles flous d'un même univers *X*, les opérations d'intersection, d'union et de produit cartésien ont pour définition :

$$\mu_{E \cap F}(t) = min(\mu_E(t), \mu_F(t)), \qquad \mu_{E \cup F}(t) = max(\mu_E(t), \mu_F(t)),$$
$$\mu_{E \times F}(t,t') = min(\mu_E(t), \mu_F(t')).$$

Le complément d'un ensemble flou est défini par la complémentation à 1 ce qui amène à satisfaire les lois de De Morgan. D'autres opérateurs peuvent être choisis pour définir l'intersection et l'union (tout couple de (norme triangulaire, co-norme triangulaire)) mais le couple (min, max) est le plus utilisé. Dans le cadre de l'interrogation de bases de

données, les ensembles flous sont utilisés pour définir les conditions atomiques. Le degré d'appartenance exprime alors en quelle mesure la valeur est préférée.

2.2. Préférences élémentaires non commensurables

Si l'hypothèse de commensurabilité n'est pas valide, les scores associés aux diverses préférences ne sont pas comparable et une fonction d'agrégation n'est pas envisageable. Un exemple caractéristique est celui où chaque préférence est exprimée par une distance par rapport à une valeur optimale. Dans ce cas, l'hypothèse de commensurabilité n'est pas valide car deux distances définies sur deux attributs différents (par rapport à deux valeurs optimales différentes) ne sont pas comparables. En conséquence, les vecteurs de scores ne peuvent être comparés par une fonction d'agrégation et il faut penser à d'autres mécanismes. En particulier, l'ordre de Pareto peut être utilisé.

Ordre de Pareto: L'ordre de Pareto entre deux vecteurs $v = (v_1, \ldots, v_n)$ et $u = (u_1, \ldots, u_n)$ est le suivant : $v <_{\text{Pareto}} u \Leftrightarrow (\forall i, v_i \leq u_i, \exists j, v_j < u_j)$ où $v_i \leq u_i$ (resp. $v_i < u_i$) indique que le score u_i est meilleur ou égal à v_i (resp. u_i est strictement meilleur que v_i).

Quand l'ordre de Pareto est employé, l'ordre obtenu sur les réponses n'est pas total. En conséquence, le résultat final délivré à l'utilisateur est l'ensemble des réponses non dominées (les réponses telles que l'on ne peut en trouver de préférées). En d'autres termes, le résultat d'une requête sur une relation R est le résultat d'un opérateur *winnow* :

$$winnow\,(R, <_{Pareto}) = \{t_i \in R \mid \nexists\, t_j \in R : t_i <_{Pareto} t_j\},$$

où l'ordre de Pareto est définie sur la relation R.

3. La bipolarité dans les préférences

Le concept de requêtes bipolaires proposées dans [Dubois et al. 2002] est une façon particulière d'envisager les préférences dans les requêtes. Les requêtes bipolaires distinguent des préférences obligatoires (appelées contraintes) et des préférences optionnelles (appelées souhaits). La contrainte est obligatoire dans le sens où elle doit être satisfaites par les réponses, le souhait est optionnel dans le sens où il peut ne pas être satisfait par une réponse. L'idée est alors de rechercher les valeurs satisfaisant les contraintes et, *si possible*, celles satisfaisants les souhaits.

Les contraintes et les souhaits sont respectivement définies par deux conditions définissant l'ensemble des valeurs acceptables (dénoté A, qui définit la contrainte) et l'ensemble des valeurs désirées (dénoté D, qui définit les valeurs souhaitées). Dans le

cas ou ces deux conditions sont booléennes, les tuples satisfaisant en priorité les contraintes et les souhaits sont retournés à l'utilisateur. Si de telles réponses n'existent pas, seul l'ensemble des tuples satisfaisants la contrainte est délivré.

De telles requêtes peuvent être utiles dans de nombreux contextes et l'on peut considérer l'interrogation d'une base de données décrivant des voitures en vente. Les besoins de l'acheteur (une voiture rouge, un prix inférieur à 1000 euros) sont des contraintes car ce sont des conditions obligatoirement satisfaites (si elles ne sont pas satisfaites la vente n'aura pas lieu). Les besoins du vendeur (un bénéfice supérieur à 400 euros, …) sont des souhaits car leur satisfaction n'est pas obligatoire, il s'agit simplement de conditions souhaitables.

La propriété fondamentale des contraintes et des souhaits est que l'ensemble des valeurs désirées doit être un sous-ensemble de celui des valeurs acceptables ($D \subseteq A$). On exprime ainsi qu'il est incohérent de désirer des valeurs non-acceptables (et ainsi, la requête précédente exprimée sur le domaine des voitures devient rechercher les voitures rouges ayant un prix inférieur à 1000 euros et, *si possible*, les voitures rouges ayant un prix inférieur à 1000 euros et avec un bénéfice supérieur à 400 euros).

Des conditions bipolaires [Lietard et al. 2009a, Lietard et al. 2009b] avec des préférences peuvent être définies par des couples d'ensembles flous. Un tuple t satisfait la contrainte au degré $\mu_A(t)$ tandis qu'il satisfait le souhait au degré $\mu_D(t)$. La condition d'inclusion entre contrainte et souhait se réécrit :

Pour chaque tuple t, $\mu_D(t) \leq \mu_A(t)$.

Les contraintes et les souhaits ne sont pas commensurables car ils ne véhiculent pas la même sémantique (par exemple, une valeur ne satisfaisant pas la contrainte est rejetée, ce qui n'est pas le cas de la non satisfaction au souhait). En conséquence, les degrés de satisfaction ne peuvent être agrégés et ils doivent être traitées indépendamment. La contrainte étant impérative, il est possible d'ordonner les tuples par un ordre lexicographique sur les contraintes et les souhaits:

un tuple t_i est préféré à un tuple t_j
si($\mu_A(t_i) > \mu_A(t_j)$) ou (($\mu_A(t_i) = \mu_A(t_j)$) et ($\mu_D(t_i) > \mu_D(t_j)$)).

En d'autres termes, la satisfaction par rapport au souhait est utilisée pour distinguer les tuples qui ont obtenu un même degré de satisfaction par rapport à la contrainte.

4. Langage de requêtes avec préférences

Dans cette section, nous présentons brièvement les principales propositions pour exprimer des préférences utilisateurs dans un langage de requêtes de type SQL. Ces propositions sont situées par rapport à l'hypothèse de commensurabilité et la notion de bipolarité.

4.1. Le langage PreferenceSQL

Les requêtes en PreferenceSQL [Kießling et al. 2002a, Kießling et al. 2002a] sont principalement constituées de deux parties : une clause WHERE pour sélectionner des tuples par des conditions Booléennes (appelées conditions de type *must*), et une clause PREFERRING pour exprimer des conditions impliquant des préférences (appelées conditions de type *light*). La discrimination entre tuples est établie sur les conditions de type *light* et le bloc de requête de base en PreferenceSQL est :

```
SELECT * FROM <liste de relation>
WHERE <conditions must>
PREFERRING <conditions light>
```

Les preferences de la clause PREFERRING sont définies par des distances (depuis des valeurs optimales) ou bien des scores (des niveau de satisfaction) et un ordre de Pareto est utilisé pour distinguer les tuples. Une requête PreferenceSQL délivre donc les tuples satisfaisant la clause WHERE qui ne sont pas dominés par rapport aux préférences. Si de tels tuples n'existent pas, tous les tuples satisfaisant la clause WHERE sont délivrés à l'utilisateur.

PreferenceSQL suit le modèle bipolaire de requêtes car les prédicats *must* (qui sont booléens) représentent des contraintes tandis que les prédicats de type *light* représentent des souhaits (un tuple qui ne satisfait pas les conditions *must* sont écartés et, *si possible*, les tuples non dominés sont présentés à l'utilisateur). De plus, étant donné que les préférences sont des distances ou des niveaux, l'hypothèse de commensurabilité n'est pas valide.

Pour conclure, on peut dire que PréférenceSQL implémente un cas particulier de bipolarité où les contraintes sont booléennes et où les souhaits sont des préférences non commensurables.

4.2. Le langage SQLf

Le langage SQLf [Bosc et al. 1995, Bosc et al. 2008] est une extension de SQL qui

permet l'expression de préférences définies par des ensembles flous. Dans ce contexte, les ensembles flous sont utilisés pour définir des conditions atomiques (prédicats vagues ou flous) qui expriment des préférences.

Dans ce contexte, la satisfaction de la valeur d'un attribut à une condition vague est un degré, plus le degré est élevé, plus la valeur est préférée. De plus, les ensembles flous généralisant les ensembles ordinaires, une conditions floue est une généralisation d'une condition Booléenne. Des normes et des co-normes triangulaires peuvent être utilisées pour définir respectivement la conjonction et la disjonction de conditions vagues et l'on utilise souvent la norme *min* et la co-nome *max* (la négation d'une condition floue étant la complémentation à 1).

Une requête typique en SQLf suit la syntaxe suivante :

```
SELECT [n|t] < attributes>
FROM <relations>
WHERE <fuzzy-condition>
```

où `<fuzzy-condition>` est une expression logique impliquant des conditions vagues (avec des conditions vagues atomiques ou des requêtes SQLf imbriquées). Le partionnement est également envisageable avec des conditions vagues portant sur les sous-ensembles issus du partitionnement.

Une telle requête délivre un ensemble de résultats pondérés par leur degré de satisfaction à la requête (niveau de préférence utilisateur). Il est possible d'effectuer un calibrage des réponses par le paramètre `n` (les `n` meilleurs réponses) ou t (les réponses ayant un degré de satisfaction au moins égal à `t`).

Le langage SQLf se situe dans un cadre algébrique (une algèbre relationnelle étendue) où l'hypothèse de commensurabilité est supposée. En conséquence, un ordre total est délivré à l'utilisateur et les équivalences de requêtes valides en SQL sont toujours vérifiées en SQLf (ce dernier point étant important pour l'optimisation de l'évaluation de la requête). Le langage SQLf n'a cependant aucune relation avec la bipolarité, les conditions vagues étant des contraintes pures (sans souhaits).

5. Conclusion

Nous avons présenté deux langages d'interrogation de bases de données relationnelles qui prennent en compte des préférences utilisateurs (PreferenceSQL et SQLf). Nous avons souligné que PréférenceSQL implémente un cas particulier de la bipolarité où les contraintes sont booléennes et où les souhaits sont des préférences non commensurables. Le langage SQLf se situe dans un cadre algébrique (une algèbre

relationnelle étendue) où l'hypothèse de commensurabilité est supposée mais il n'a aucune relation avec la bipolarité.

En conséquence, il nous semble pertinent d'enrichir le langage SQLf par la prise en compte de la bipolarité, ce qui nous permettra de rester dans un cadre algébrique étendu. Des travaux ont déjà commencé en ce sens [Liétard et al. 2009a, Liétard et al. 2009b] où une forme plus générale de conditions floues a été proposée : les conditions bipolaires floues.

Bibliographie

Bosc P., Pivert O. “SQLf: a relational database language for fuzzy querying”. In IEEE Transactions on Fuzzy Systems, vol(3) p.1–17, 1995.

Bosc P., Liétard L. “Aggregates computed over fuzzy sets and their integration into SQLf“, in International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems", vol. 16, no 6, p. 761-792, 2008.

Chomicki J. “Preference Formulas in Relational Queries”. In ACM Transactions on Database Systems, (TODS’03), 28(4):1–39, 2003.

Dubois D. and Prade H. “Bipolarity in flexible querying”. In Proc. of the 5th International Conference on Flexible Query Answering Systems, (FQAS’02), Copenhagen, Denmark, 2002.

Kießling W. “Foundations of Preferences in Database Systems”. In Proc. of the 28th International Conference on Very Large Data bases, (VLDB), Hong Kong, China, pp. 331–322, 2002a.

Kießling W. and Kostler G. “Preference SQL - Design, Implementation, Experiences”. In Proc. of the 28th International Conference on Very Large Databases, (VLDB), Hong Kong, China, pp. 990–1001, 2002b.

Liétard L. Rocacher R. “On the extension of SQL to Fuzzy Bipolar Conditions“. In proc. of the North American Fuzzy Information Processing Society, Cincinnati, Ohio, USA, 2009a

Liétard L. Rocacher R. “On the Definition of Extended Norms and Co-norms to Aggregate Fuzzy Bipolar Conditions“. In Proc. of the International Fuzzy Systems Association – European Society fo Fuzzy Logic and Technology, Lisbon, Portugal, 2009b

Zadeh, "Fuzzy sets". Information and Control, 8 (3) 338–353, 1965.