

# Représentation et indexation d'objets mobiles dans un entrepôt de données

Tao Wan, Karine Zeitouni

Laboratoire PRISM, Université de Versailles  
45, avenue des Etats-Unis, 78035 Versailles Cedex, France  
Tao.Wan@prism.uvsq.fr, Karine.Zeitouni@prism.uvsq.fr  
<http://www.prism.uvsq.fr/users/karima/>

**Résumé.** Le développement considérable des techniques de géo localisation et des périphériques mobiles mène à une profusion de bases d'objets mobiles et soulève la question de l'exploitation pour l'aide à la décision. Si les systèmes conventionnels d'analyse en ligne (OLAP) sont utilisés efficacement dans l'analyse des données basées sur la modélisation multidimensionnelle, ils ne sont pas adaptés aux bases d'objets mobiles. Ces objets possèdent outre les attributs descriptifs, une trajectoire variant continuellement dans des dimensions spatiales et temporelles. Cet article étudie le problème de l'entreposage d'objets mobiles dans un réseau. Ses contributions sont : (1) un modèle conceptuel qui étend le modèle multidimensionnel aux dimensions et aux faits continus ; (2) un modèle de représentation adapté ; (3) une structure d'indexation et des algorithmes optimaux pour les requêtes spatiotemporelles OLAP sur ces objets mobiles. Un prototype a été développé et les résultats de l'expérimentation, détaillés ici, montrent l'efficacité des méthodes proposées.

## 1. Introduction

La gestion de bases d'objets mobiles a reçu une attention particulière durant ces dernières années en raison des avancées et la banalisation des technologies mobiles et de géo-localisation, telles que les téléphones cellulaires, le GPS (*Global Positioning Systems*) et récemment le RFID (*Radio Frequency Identification*). La plupart des travaux s'inscrivent dans un contexte transactionnel et sont axés sur la modélisation d'objets mobiles dans Güting et al. (2000) et dans Vazirgiannis et Wolfson (2001), sur les méthodes d'accès dans Pfoser et al. (2000) et dans Saltenis et al. (2000), et enfin sur les requêtes prédictives et l'optimisation des mises à jour dans Chon et al. (2002), Tao et al. (2002) et dans Jensen (2004). Cependant, beaucoup reste à faire pour l'exploitation d'historiques d'objets mobiles dans un but décisionnel.

Prenons comme exemple une application type en santé environnementale<sup>1</sup>. Dans cette application, il était intéressant d'analyser l'exposition aux différents facteurs de risques (pollution, bruit, danger d'accident) générés par le transport routier. Ainsi, un expert en santé environnementale cherche à estimer l'exposition aux risques sous différents angles : risque

---

<sup>1</sup> Cette application est issue du projet Européen HEARTS (<http://www.euro.who.int/heart>)

## Représentation et indexation d'objets mobiles dans un entrepôt

individuel, par groupe (ex. d'âge), par lieu quelconque et/ou par période. Cela mène à des requêtes du style :

Q1 : Combien de personnes sont exposées au cours du temps au champ de pollution ?

Q2 : Combien de personnes par catégorie d'âge sont exposées au champ de pollution ?

Ces requêtes exigent principalement des agrégations groupées par localisation, intervalle de temps et/ou attributs d'objets mobiles (ex : âge). Si l'on suppose les cartes de pollution données par des mesures par localisation et par intervalle de temps, ces requêtes correspondent, a priori, à un calcul d'agrégat d'objets mobiles, ici un comptage du nombre de personnes distinctes. En notant INTERSECT l'intersection spatiotemporelle, on pourrait exprimer ces requêtes en notation SQL comme suit :

```
SELECT  p.localisation, p.temps, COUNT (DISTINCT id_personne) AS nombre_personnes
FROM    population m, pollution p
WHERE   m.trajectoire INTERSECT (p.localisation, p.temps)
GROUP BY      p.localisation, p.temps
```

et :

```
SELECT  p.localisation, p.temps, COUNT (DISTINCT id_personne) AS nombre_personnes
FROM    population m, pollution p
WHERE   m.trajectoire INTERSECT (p.localisation, p.temps)
GROUP BY      m.âge, p.localisation, p.temps
```

Aujourd'hui, les entrepôts de données et les systèmes OLAP (*On-Line Analytical Processing*) sont reconnus pour leur adaptation aux besoins d'analyse des décideurs et pour leurs performances d'exécution. L'analyse mentionnée ci-dessus est analogue à l'analyse multidimensionnelle dans les systèmes OLAP. Cependant, les modèles multidimensionnels actuels ne s'appliquent pas à l'analyse d'objets mobiles en raison des spécificités suivantes :

- Les requêtes spatiotemporelles OLAP comme ci-dessus exigent le calcul d'agrégats par intervalles de temps et zones de l'espace. Ce résultat d'agrégation dépend de la trajectoire des objets mobiles qui forment une variation continue selon les dimensions temps et espace. Or, les modèles multidimensionnels conventionnels sont tous basés sur des faits ponctuels et n'intègrent pas des critères spatiotemporels ;
- Les critères de regroupement (ici les champs de pollution) ne sont pas forcément connus à l'avance et ne peuvent se référer à un découpage spatial ou temporel prédéfinis. Par exemple, la mobilité à caractère périodique (stockée une fois pour toute) est croisée avec les champs de pollution relevés quotidiennement. Par conséquent, aucune discrétisation préalable de l'espace ou du temps ne serait satisfaisante et donc, leur caractère continu doit être pris en compte dans les dimensions et dans l'analyse ;
- Il est indispensable d'intégrer dans le modèle les dimensions continues avec les dimensions discrètes comme le type d'activité pouvant être combinées lors de l'analyse. L'identité de l'objet mobile et ses attributs sont en effet des dimensions d'analyse essentielles, comme pour analyser les risques individuels ou les risques agrégés par âge.

**Travaux liés** : Il existe des travaux tant sur les entrepôts de données spatiaux, dans lesquelles Stefanovic et al. (2000) était le pionnier, que sur les entrepôts spatiotemporels Marchand et al. (2004). Bien qu'ils proposent l'intégration des caractéristiques spatiales ou spatiotemporelles dans un entrepôt, le cas des objets mobiles n'y est pas considéré. Le premier travail lié à l'entreposage d'objets mobiles est Papadias et al. (2002). Dans ce

dernier, les auteurs traitent un modèle multidimensionnel se référant aux objets mobiles où une cellule mesure l'effectif d'objets mobiles dans une unité espace-temps. Ils proposent un index, combinaison du *R-tree* et du *B-tree*, pour stocker l'historique de l'effectif par zone. Cet article a été suivi par celui de Tao et al. (2004), qui soulève le problème du double comptage des objets mobiles traversant plusieurs unités espace-temps et comptés dans plusieurs cellules, puis le résout par une estimation probabiliste. Ces deux travaux partent de l'hypothèse que les objets mobiles sont agrégés à l'origine et ne permettent pas de représenter d'autres dimensions que l'espace et le temps. Limités par cette hypothèse, ces modèles ne peuvent ni représenter les trajectoires d'objets individuels ni les attributs descriptifs (ex : âge, sexe). De plus, le résultat de leurs requêtes OLAP peut seulement fournir des statistiques approximatives, bien que l'article dans Tao et al. (2004) essaie d'améliorer l'approximation. Récemment, une modélisation multidimensionnelle pour les objets mobiles a été proposée dans Wan et al. (2005). L'objectif de ce travail est d'intégrer les propriétés d'objets mobiles dans l'analyse de la mobilité. Cependant, cette modélisation se base sur des découpages de référence pour l'espace et le temps. Elle ne permet pas de répondre aux agrégats par requêtes spatiales et/ou temporelles à cheval sur ces découpages. Les travaux existants sur les entrepôts de données spatiales ou spatiotemporelles partent tous de l'idée d'un découpage connu de l'espace et du temps.

Motivés par les limitations des travaux existants, nous proposons dans cet article une modélisation et une représentation des objets mobiles dans un entrepôt de données pour une exploration de type OLAP. De plus, comme la grande majorité des objets se déplacent dans un environnement géographique contraint par un réseau (les routes, les chemins de fer ou les couloirs aériens), nous focalisons notre travail sur la mobilité restreinte au réseau. En effet, de tels objets mobiles contraints constituent une catégorie importante des travaux sur les objets mobiles dans le domaine des transports. Cette propriété de localisation relative au réseau nous permet de réduire la dimensionnalité et d'optimiser le stockage Pfoser et al. (2002).

**Contributions :** (1) Pour décrire la variation continue de trajectoires d'objets mobiles, nous étendons les notions de fait et de dimension. Ainsi, une dimension peut être continue (ici, le temps ou l'espace sont des dimensions continues). Un fait n'est plus rattaché à un événement à une certaine granularité des dimensions (ex. du temps et de l'espace), mais peut également varier de manière continue par rapport à ces dimensions. (2) Nous dérivons une structure de données basée sur des intervalles pour représenter des faits mobiles et ce, grâce à la technique de réduction de dimensionnalité. (3) Nous proposons et mettons en œuvre une structure d'index et l'utilisons pour optimiser des requêtes spatiotemporelles OLAP. A notre connaissance, il n'existe dans la littérature aucune véritable approche pour l'analyse en ligne d'objets mobiles.

Le reste de cet article est organisé comme suit. La section suivante introduit des concepts de base. La section 3 propose notre modèle d'objets mobiles contraints par le réseau dans un entrepôt et décrit ensuite sa représentation. La section 4 propose une technique d'indexation permettant l'implémentation optimisée de requêtes OLAP pour cet entrepôt. La Section 5 donne les résultats expérimentaux avant la conclusion par une discussion et des perspectives.

## 2. Notions préliminaires

Cette section donne les notions de base du réseau, des trajectoires contraintes et des objets mobiles.

## Représentation et indexation d'objets mobiles dans un entrepôt

Un réseau routier est couramment modélisé par un ensemble de tronçons et d'intersections (ou nœuds) délimitant les tronçons à leurs extrémités. On se réfère au modèle développé dans Vazirgiannis et Wolfson (2001) où un objet est supposé avoir une vitesse constante sur un segment de route et peut changer de vitesse d'un segment à l'autre. Une trajectoire est une courbe définie comme une fonction du temps dans un espace 2 dimensions, ce qui forme une courbe 3D. Lorsqu'il s'agit de trajectoires contraintes par le réseau, les coordonnées absolues de l'espace 2D peuvent être remplacées par la position relative dans les tronçons traversés du réseau. Si, de plus, on considère une variation moyenne de la vitesse par tronçon, il suffirait de renseigner et dater les points correspondant au passage d'un tronçon à l'autre. En outre, nous adaptons une technique de réduction de dimensionnalité de Pfoser et Jensen (2003). L'idée est d'appliquer une transformation du réseau, à l'origine 2D, en intervalles 1D connexes. Ensuite, il suffit de projeter la position relative de l'objet sur cet axe pour obtenir une position absolue en 1D et donc une transformation de la trajectoire en 2D au lieu des courbes 3D. Les avantages sont, d'une part la simplification de la représentation de l'espace et d'autre part, l'optimisation à la fois de l'espace de stockage et des temps de réponses aux requêtes. Cela nous amène aux définitions suivantes :

**Définition 1 :** Une **transformation 1D du réseau routier** est définie par une fonction  $Tr$  ( $rid_i$ ) =  $[i', i'+I]$  dans un espace 1D cible borné appelé TR, où  $rid_i$  est l'identifiant d'un tronçon et  $i, i' \in \{1 \dots, n\}$ .

Cette transformation revient à numéroter les tronçons de routes et les aligner les uns après les autres sur des intervalles de l'axe TR. Comme il n'y a pas de recouvrement entre les intervalles, la référence à  $rid_i$  peut être substituée par  $i'$ . De plus, toute position relative  $pos$  à un tronçon  $rid_i$  du réseau pourra être transformée en position absolue  $p$  dans la dimension TR par l'expression :  $p = i' + pos$ .

Dès lors, la transformation de trajectoire contrainte par le réseau pourra se baser sur les positions absolues dans TR. Il en découle la définition suivante de la transformation en 2D d'une trajectoire.

**Définition 2: Une transformation 2D de trajectoire TT** ( $tid, [(p_i^d, t_i^d), (p_i^f, t_i^f)]_{i=1..k}$ ) où  $k$  est le nombre de tronçons traversés,  $p_i^d$  (respectivement,  $p_i^f$ ) est une position absolue sur l'axe TR correspondant au point du début (respectivement, de la fin) de la trajectoire dans le tronçon  $rid_i$  à l'instant  $t_i^d$  (respectivement à l'instant  $t_i^f$ ) et tel que  $t_i^d = t_i^f$ .

Cette transformation revient à représenter une trajectoire par un ensemble de segments de droites dans l'espace 2D  $TR \times Temps$  (cf. figure 1).

Enfin, on introduit la définition des objets mobiles contraints. Contrairement à d'autres travaux, ces objets ne se limitent pas à leur trajectoire. En effet, dans de nombreuses applications, les attributs rattachés à l'objet tels que le type de véhicule, l'âge ou le sexe du conducteur, font partie des critères d'analyse. Par conséquent, nous proposons la définition suivante, similaire à celle de Vazirgiannis et Wolfson (2001).

**Définition 3:** Un objet mobile contraint est défini par **OMC** ( $mid, A_1, A_2, \dots, A_n, (tid_i)_{i=1}^m$ ) où  $mid$  est son identifiant,  $A_i$  ( $1 \leq i \leq n$ ) sont ses attributs et  $(tid_i)_{i=1}^m$  représentent l'ensemble des trajectoires parcourues par l'objet.

### 3. Conception et représentation de l'entrepôt

Traditionnellement, les entrepôts de données sont conçus dans un modèle multidimensionnel de Gray et al. (1996) et définissent des cubes de données que les outils OLAP permettent d'explorer aisément dans Agrawal et al. (1997). Le paradigme multidimensionnel est intéressant pour l'exploration des objets mobiles, mais il ne peut s'appliquer tel quel en raison de deux contraintes.

La première est la modélisation de la variation continue d'objets mobiles dans l'espace et dans le temps. En effet, contrairement à beaucoup d'approches dans lesquelles un fait spatiotemporel est limité à un événement, c'est-à-dire se produisant ponctuellement dans le temps, ou à un état, c'est-à-dire valide sur une durée, les objets mobiles sont des faits spatiotemporels continus, car leur localisation géographique change continuellement. Or, un entrepôt de données conventionnel exige des valeurs discrètes sur les faits et les dimensions.

La deuxième contrainte est d'agréger efficacement des données, quelque soient les critères spatiotemporels. Par exemple, dans la requête  $Q1$ , à des périodes différentes, les champs de pollution forment des découpages différents. Le croisement des trajectoires mobiles avec le champ de pollution correspond à une agrégation par zones et périodes fournies au moment de la requête. Par conséquent, aucune discrétisation des dimensions espace ou temps dans le cube ne serait satisfaisante. Cela rend difficile l'application des techniques de pré agrégations comme dans Stefanovic et al. (2000). La deuxième conséquence est que cette agrégation implique la combinaison avec un critère spatiotemporel pour appliquer le découpage.

En considérant les contraintes mentionnées ci-dessus, nous proposons d'étendre les notions de fait et de dimension à la variabilité continue. Ensuite, nous dérivons une structure de données basée sur des intervalles pour représenter des faits mobiles, grâce à la technique de réduction de dimensionnalité.

**Définition 4:** Une *dimension continue* est une dimension variant dans un domaine continu sans limite de granularité.

**Définition 5:** Un *fait continu* est le résultat d'une fonction continue qui dépend entre autres d'une ou de plusieurs dimensions continues.

Certains travaux considèrent également des dimensions continues comme dans Shanmugasundaram et al. (1999). Pour autant, un fait reste un point isolé dans l'espace des dimensions. Par ailleurs, Ahmed et al. (2004) définissent des vues de cubes continus spatiotemporels en utilisant une interpolation des mesures sur l'espace et le temps. Notre définition est plus générale que ces derniers travaux. Elle va au-delà et permet la modélisation réelle d'objets mobiles dans un entrepôt par un cas particulier défini ci-dessous :

**Définition 6 :** Un *fait mobile* est un fait continu dépendant de l'objet et défini par une fonction  $F_{OMC}(t) = s$  continue du temps dans l'espace.

En fait, la fonction  $F_{OMC}$  dépend des deux dimensions continues « espace » et « temps ». Elle correspond aux trajectoires contraintes de l'objet OMC. Comme souligné dans la section 2, ces trajectoires peuvent être représentées en 2D avec les dimensions TR et temps. De plus, si l'on considère une vitesse constante ou une moyenne par tronçon, cette fonction devient linéaire par morceaux. Cette représentation du cube est illustrée dans FIG 1. Soient les quatre objets mobiles associés à Marie, Fred, Tao et Jack. Dans ce modèle, les trois dimensions représentent l'objet par son identité, le réseau routier en guise de dimension spatiale continue et le temps continu. Ce modèle adopte la réduction de dimensionnalité et ramène l'espace 2D

## Représentation et indexation d'objets mobiles dans un entrepôt

du réseau à une dimension continue. Le fait mobile trace la présence de chaque objet mobile dans le cube, donc par une infinité de faits s'il était défini en extension. Défini en intension, il correspond à une représentation paramétrique par une fonction linéaire continue par morceaux pour chaque objet, par exemple ici, en trois morceaux pour Jack et en quatre morceaux pour Marie.

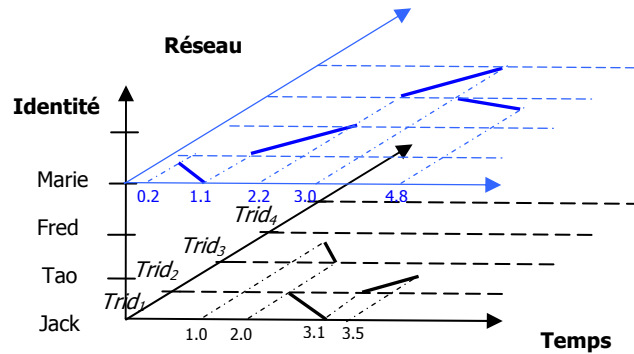


FIG. 1 - Modèle multidimensionnel pour objets mobiles contraints.

Comme il n'est pas possible de représenter une infinité de faits et qu'une représentation paramétrique par fonction n'est pas facilement manipulable dans un contexte de bases de données, nous proposons un modèle d'implémentation tirant profit des caractéristiques du modèle logique. L'idée est que chaque fonction linéaire peut être représentée par deux intervalles et une orientation. On propose donc de traduire le modèle logique de faits mobiles  $F_{OMC}$  en un ensemble fini de valeurs :

$$T_{OMC} = ([a_i, b_i], O_i, [t_i, t'_i])_{i=1}^n$$

où  $[a_i, b_i]$  est un intervalle inclus dans la dimension TR et représentant une partie d'un tronçon de route,  $O_i$  ( $O_i \in \{-1, 1\}$ ) est l'orientation d'OMC par rapport au sens (dans la représentation) du tronçon, et  $[t_i, t'_i]$  est l'intervalle de temps durant lequel OMC a parcouru le tronçon. L'avantage de cette représentation est de résumer les faits tout en facilitant les requêtes par plages de valeurs dans l'espace, le temps séparément ou leur combinaison.

La FIG. 2 illustre un exemple de représentation par intervalles, traduction de la FIG. 1. Par exemple, on résume le fait mobile  $F_{marie}$  de la FIG. 1 par le passage dans le tronçon de route transformé Trid1 entre les positions 0.6 et 0 dans l'intervalle de temps  $[0, 1.1]$ , puis par le passage le long de Trid2 et de Trid4 dans les intervalles  $[1.1, 2.2]$  et  $[2.2, 3.0]$ , et en dernier par le passage dans Trid3 où sa trajectoire s'est arrêtée à la position 2.6 dans l'intervalle de temps  $[3.0, 4.8]$ . Ceci revient à représenter la mobilité de Marie par :

$$T_{marie} = (([0, 0.6], -1, [0.2, 1.1]), ([1, 2], 1, [1.1, 2.2]), ([3, 4], 1, [2.2, 3.0]), ([2.6, 3] - 1, [3.0, 4.8]))$$

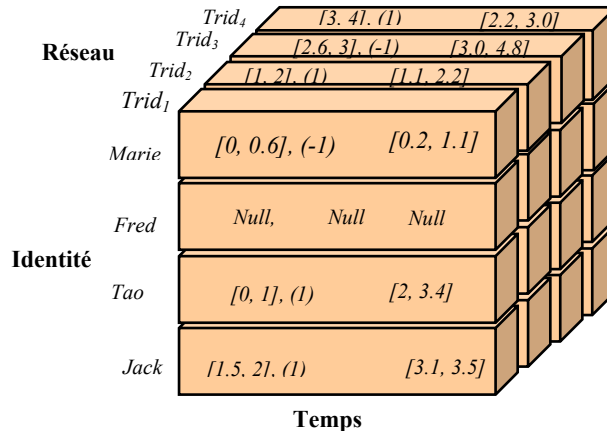


FIG. 2. - Représentation par intervalles de cubes d'objets mobiles contraints.

#### 4. Implémentation optimisée des requêtes OLAP

L'implémentation des cubes de données utilise diverses techniques d'optimisation de requêtes OLAP, dont les principales sont l'indexation et la pré-agrégation. Néanmoins, ces méthodes sont insuffisantes lorsque l'agrégation porte sur d'autres plages de valeurs que celles prédéfinies dans le cube.

Examinons les deux requêtes types données dans l'introduction. La première retourne le nombre de personnes exposées au champ de pollution. Quand à la seconde requête, plus complexe, elle différencie l'exposition par catégorie d'âge.

Pour y répondre, il faut optimiser les opérateurs spatiotemporels tout en tenant compte de la différenciation par identité de l'objet et en optimisant le calcul du comptage distinct (*count distinct*). En fait, le problème posé est de traiter efficacement l'agrégation par intervalle ou par combinaison d'intervalles et de dimensions discrètes. La combinaison d'intervalles traduit des requêtes spatiotemporelles, tandis que les dimensions discrètes expriment l'exploration par les attributs de l'objet.

Pour ces requête OLAP, dites « *range query* » ou « *requêtes par intervalles* », des travaux ont proposé une optimisation par le stockage d'agrégats dans des index *R-tree* comme dans Ho et al. (1997) et dans Junger et Lenz (1998). Notre problématique est similaire, mais, ces travaux, se limitent tous aux fonctions agrégats algébriques ou distributives. Or, la fonction « *count distinct* » est holistique et ne satisfait pas ces propriétés. C'est pourquoi nous proposons une nouvelle structure d'index. Une autre spécificité de notre modèle de représentation est que les requêtes par intervalles portent sur des objets eux-mêmes définis par des intervalles. Ces objets, dits avec extension, ont été pris en compte dans Zhang et al. (2001), Zhang et al. (2002). Néanmoins, à la différence de ces derniers, le croisement des extensions temps et espace est interprété comme un segment de droite auquel on peut appliquer une requête spatiotemporelle précise. Enfin, la transformation en 1D des données engendre la transformation de la requête avant sa résolution. Nous devons donc adapter les algorithmes en conséquence.

### 4.1 Index proposé TTR-tree

L'avantage de la réduction de dimensionnalité adoptée est de résoudre les requêtes spatiotemporelles par un index 2D comme un *R-tree* 2D. Mais à la différence du *R-tree*, les requêtes, telles celles données en exemple, comprennent des agrégats et combinent parfois les attributs de l'objet. Stocker simplement le résultat de la fonction agrégat dans l'index comme dans Ho et al. (1997) par exemple, ne suffit pas car cela ne garde pas la trace de l'objet pour adjoindre ses propriétés et engendre le double comptage comme soulevé dans Papadias et al. (2002).

Nous proposons une structure appelée *TTR-tree* (ou *Transformed Trajectory R-tree*) inspirée du *R-tree*, mais dont les nœuds comprennent un bitmap représentant les objets mobiles indexés par ce nœud. La FIG. 3 illustre un exemple de *TTR-tree* qui indexe les objets mobiles de la FIG. 3. Chaque objet mobile est représenté par une position dans le bitmap. Ainsi, Marie correspond à la première position, Fred à la deuxième et ainsi de suite. Par exemple, R1 comprend une partie de la trajectoire de Marie, Fred et Jack. Le nœud correspondant à R1 pointe donc le bitmap 1101. On peut remarquer qu'il suffit de représenter les cellules au niveau des feuilles et une référence à l'objet pour que le *TTR-tree* remplace complètement la structure du cube.

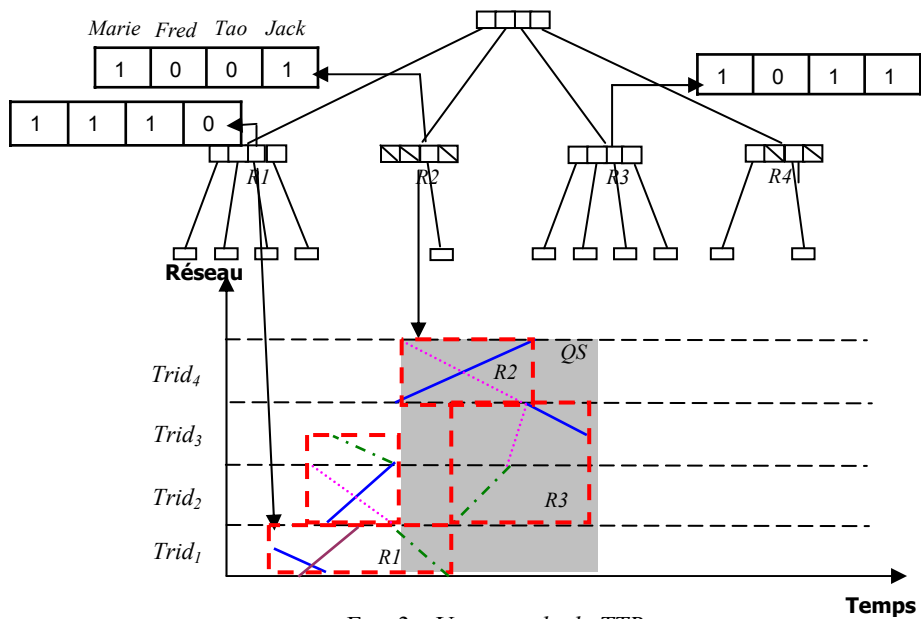


FIG. 3 - Un exemple de TTR-tree.

### 4.2 Agrégat spatiotemporel utilisant le TTR-Tree

Grâce à l'index bitmap, la fonction « *count distinct* » est résolue efficacement à différents niveaux d'échelles spatiotemporelles. Il est en outre adapté au groupage combinant d'autres

attributs indexés par un bitmap. Nous donnons ci-dessous un algorithme générique (FIG. 4) de requête spatiotemporelle agrégat. Celui-ci prend en paramètre l'index *TTR-tree* et une fenêtre spatiotemporelle *QS*. Il retourne le résultat sous forme d'un bitmap référençant les objets vérifiant le critère spatiotemporel. Comme dans les index bitmap classiques en bases de données, cette structure est très efficace pour les agrégats et plus particulièrement le « *count distinct* » qui se fait sans accès aux données. Au-delà des agrégats, cette structure représente le résultat de la sélection spatiotemporelle. L'algorithme de recherche est similaire à celui d'un *R-tree* classique, dans le sens où les nœuds dont les rectangles englobants n'ont pas d'intersection avec *QS* ne seront pas visités. Il est plus performant que le *R-tree*, car les nœuds dont le rectangle est contenu dans *QS* délivrent directement l'information utile à l'agrégation (via le bitmap), ce qui fait l'économie du parcours de leur sous-arbre. Il suffit d'un opérateur « OU » sur les bitmaps pour obtenir l'union des résultats partiels sur les nœuds visités. En définitive, le *TTR-tree* offre une bonne sélectivité, que la fenêtre *QS* soit large exploitant les résultats pré-calculés des niveaux supérieurs de l'arbre, ou quelle soit petite offrant un accès au pire des cas en  $O(\log(n))$  comme un index *R-tree*. La seule nuance est qu'une requête *QS* est d'abord transformée en 2D et peut être éclatée en plusieurs sous requêtes, car les tronçons adjacents ne sont pas forcément consécutifs sur l'axe 1D TR. Il faut donc en tenir compte dans l'algorithme. L'union des résultats des sous-requêtes est effectuée simplement par l'opérateur binaire « OU » sur les bitmaps correspondants.

```

Fonction ST_AG (Xi, QS)
// Xi est un nœud du TTR-tree, initialement la racine, QS est une fenêtre spatiotemporelle
Résultat := <00...0>           -- Bitmap initialisé à 0
Transformer QS en ensemble de fenêtres {q1 ..., qk}
Pour tout x ∈ Xi
    Pour tout qi ∈ Q
        Résultat := Résultat OU ST_filtre (x, qi)
Fin.

Fonction ST_filtre (x, q)
Si q contient x
    // Retourner directement le bitmap pointé par x
    Résultat := Résultat OU x.bitmap
Sinon Si q intersecte x
    Si x est une feuille
        Pour tout objet o dans x
            Si o intersecte q
                // Retourner le bitmap correspondant à l'objet (bit à 1 sur la position de l'objet)
                Résultat := Résultat OU <0...1...0>
            Sinon -- x n'est pas une feuille
                // Visiter récursivement les nœuds pointés par x
                Pour tout fils y de x
                    Résultat := Résultat OU ST_filtre (y, qi)
Sinon
    Fin.

```

FIG. 4 - . Algorithme de requête agrégat spatiotemporelle.

La fenêtre *QS* est définie à l'origine par une fenêtre spatiale 2D et un intervalle temporel. Afin de permettre sa traduction en 2D, on dissocie le temps et l'espace. Le temps reste

identique après la transformation, quant à la fenêtre spatiale, elle est traduite en un ensemble d'intervalles de l'axe TR correspondant aux tronçons intersectés. Pour ce faire, on propose d'utiliser un index spatial de type *SKD-tree* de Ooi et al. (1987) permettant ainsi d'optimiser cette phase. L'avantage par rapport aux courbes de Hilbert préconisées par Pfoser et Jensen, (2003) est qu'il s'adapte mieux aux formes et aux tailles des tronçons. De plus, il constitue un excellent moyen de grouper et de numéroter les segments par proximité de manière adaptative à leur densité. En effet, il suffit d'utiliser l'ordre des feuilles pour trier les segments transformés dans l'axe TR. Cette numérotation par proximité réduit considérablement l'éclatement en sous-requêtes, permettant une optimisation des traitements.

L'agrégat avec groupage, tel que dans les deux requêtes exemples, se traduit par une boucle invoquant cet algorithme pour chaque groupe, autrement dit pour chaque requête spatiotemporelle. Il est possible d'optimiser au-delà ces requêtes de type jointure spatiotemporelle si un index est défini sur les champs de pollution. Il suffirait d'adapter les algorithmes de jointure par parcours simultané de deux index.

### 4.3 Optimisation du stockage

Les index bitmap sont largement utilisés dans les applications OLAP de Chaudhuri et al. (1997). Le problème des index bitmap est leur coût de stockage. En les compressant, on peut réduire l'espace requis en mémoire secondaire et diminuer également le coût des entrées/sorties. Seulement, certaines opérations booléennes peuvent nécessiter la décompression ou le décodage coûteux du bitmap ainsi compressé. La technique de compression ne doit donc pas être appréciée uniquement par rapport à sa capacité d'optimisation du stockage, mais aussi par ses performances d'exécution des opérations sur le format binaire. Après l'étude de l'état de l'art, nous avons choisi la technique WAH (Word-Aligned Hybrid code) proposée par Wu et al. (2004) qui offre un bon compromis entre la compression et l'efficacité des opérations en binaire. Elle est bien adaptée aux index bitmap en bases de données.

## 5. Expérimentation

Cette section évalue expérimentalement l'index proposé *TTR-tree* en le comparant avec deux alternatives : (1) *l'indexation* notée ici *Pfoser* comme proposé par Pfoser et Jensen (2003) qui nous a servi de méthode de référence. Cette méthode fait passer une représentation de trajectoires de trois dimensions en deux dimensions et permet de les indexer par un simple index en 2D. (2) *3DR\*-tree* de Beckmann et al. (1990), qui est une méthode d'indexation spatiale en 3D. La section 5.1 examine le coût de stockage de l'index et discute la performance sur l'exécution de l'opérateur "OU". La section 5.2 évalue son efficacité dans la réponse aux requêtes décisionnelles.

### 5.1 Coût de stockage et de l'opérateur "OU"

En raison du manque de données réelles, les trajectoires de OMC sont créées par le générateur de données synthétiques de Brinkhoff dans Brinkhoff (2002). Ce générateur simule des trajectoires d'objets mobiles évoluant dans des réseaux routiers réels.

Jeux de données	Nombre d'objets/temps	Nombre d'unités de temps	Nombre d'objets	Taille des données	nombre de tronçons traversés	Réseau sous-jacent
DS1	20	500	10000	24.1 M	407419	Oldenburg
DS2	20	1000	20000	49.4 M	829103	Oldenburg
DS3	20	500	10000	57.4 M	804660	Joaquin San
DS4	20	1000	20000	114.1 M	1640815	Joaquin San

TAB 1 – Les configurations de paramètres pour différents jeux de données.

Dans l'expérimentation, nous avons utilisé deux réseaux routiers réels : *Oldenburg* et *San Joaquin*, comprenant respectivement 7035 et 24123 tronçons. Afin de produire différents jeux de données pour chacun de ces réseaux, nous avons fait varier le paramètre : *nombre d'unités de temps* et fixé à 20 le *nombre d'objets mobiles créés par unité de temps*. Ainsi, nous avons généré les jeux de données illustrés dans le tableau TAB 1 dont la taille varie de 24 à 114 Méga octets.

### 5.1.1 Coût du stockage du TTR-tree

Le *TTR-tree* est un *R-tree* dont les nœuds comportent des index bitmap fournissant l'identifiant des objets mobiles indexés par ces nœuds. Ces index bitmap sont essentiels car ils nous permettent d'éviter le double comptage et d'associer la trajectoire de l'objet avec ses attributs descriptifs (ex : l'âge, le sexe). En contrepartie, ils entraînent un surcoût de stockage par rapport à un index sans ces bitmaps en dépit de la technique de compression adoptée. C'est pourquoi la première expérience évalue l'espace disque consommé par le *TTR-tree* pour différents jeux de données et différentes implémentations avec ou sans compression. Le tableau présenté dans TAB 2 compare les tailles (en Kilo octets) des index sans bitmaps de *Pfoser* et *3DR-tree* et ceux avec bitmap normal ou compressé dans *TTR-tree*.

Jeux de données	Pfoser	3DR-tree	TTR-tree	TTR-tree avec compression des bitmaps
DS1	8925 K	34753 K	9025 K	8790 K
DS2	18020 K	71351 K	18884 K	17899 K
DS3	18251 K	67616 K	18221 K	17771 K
DS4	36849 K	138601 K	38399 K	36397 K

TAB 2 – Comparatif des coûts de stockage des différents index.

On constate que le *3DR-tree* occupe beaucoup -plus de trois fois- plus d'espace disque que les autres méthodes. Le *TTR-tree* avec des bitmaps non compressés a presque la même taille de stockage que l'index *Pfoser*. Ceci s'explique par le fait que seuls les nœuds internes ont des bitmaps rattachés. En adoptant la technique de compression de bitmaps (ici, *WAH*), la consommation d'espace disque de *TTR-tree* reste linéaire et inférieure à celle de l'index *Pfoser*, même si la quantité d'objets mobiles augmente rapidement. En conclusion, l'expérimentation a montré que le surcoût de stockage des bitmaps est négligeable par rapport à la taille de la base de données. L'index *TTR-tree* est comparable en stockage à l'index de *Pfoser* et que la réduction de dimensionnalité dans ces deux index permet de réduire l'espace requis par rapport à l'index *3DR-tree*.

### 5.1.2 Coût de l'opérateur binaire "OU"

Les opérations sur des codes binaires comme le "OU" sont souvent coûteuses en temps d'exécution pour nombre de méthodes de compression qui nécessitent la décompression des bitmaps compressés. Dans notre prototype, la méthode de compression choisie *WAH* offre un bon compromis entre compression et opérations sur les bitmaps compression. De plus, comme on peut le voir dans l'algorithme en FIG. 4, l'utilisation de l'opérateur "OU" est généralement couplée avec la lecture d'un nœud de l'arbre *TTR-tree*. Par conséquent, son coût est négligeable par rapport au coût d'entrées /sorties lors du processus de recherche.

## 5.2 Requêtes d'agrégation

L'indexation du *3DR-tree*, ainsi que celle de *Pfoser* sont conçues dans un contexte transactionnel pour répondre aux requêtes sur la position individuelle d'objets mobiles dans l'espace et le temps. Dans ces méthodes, s'il y a intersection de la requête avec un nœud de l'arbre, la recherche se poursuit jusqu'aux feuilles pour déterminer le nombre d'objets situés dans cette requête. *TTR-tree* est une méthode développée dans un contexte, a priori, décisionnel et vise à optimiser les requêtes combinant agrégation (y compris le comptage distinct) et requête spatio-temporelle.

La mesure de performances de la plupart de méthodes d'indexation est principalement basée sur le coût des entrées/sorties, mesuré par le nombre de nœuds consultés pendant le test de montée en charge du volume de données. L'expression d'une requête d'agrégation spatiotemporelle, c'est-à-dire utilisant comme critère d'intersection spatiotemporelle  $q$ , implique deux paramètres qui affectent les performances : (i) la taille du facteur "espace" de la requête ( $qs$ ), représentée par le pourcentage de sa zone dans l'univers spatial et (ii) la longueur du facteur "temps" ( $qt$ ), représentée par le pourcentage de sa longueur (i.e. intervalle) par rapport au temps délimitant les données de la base. Ces paramètres ont des valeurs identiques pour toutes les requêtes du même test de montée en charge. De plus, nous avons adopté la technique de réduction de dimensionnalité pour transformer le réseau, à l'origine en 2D, en un axe de 1D composé par des intervalles connectés. Par conséquent, une requête spatiotemporelle en 3D doit être d'abord transformée en 2D et peut-être éclatée en plusieurs sous requêtes. Par conséquent, le coût d'une requête est également influencé par le nombre de sous requêtes résultats de la transformation, lequel est, à son tour, dépendant du regroupement par proximité des numéros de tronçons dans le réseau transformé.

Dans cette expérimentation, les requêtes sont générées en faisant varier (i) la position de la fenêtre de chaque requête  $q$  uniformément dans l'espace dans le but d'éviter des requêtes inutiles conduisant vers des zones vides; (ii) l'intervalle uniformément dans le temps. En total, nous générons un ensemble de montées en charge dont chacun contient 500 requêtes de même taille. Les valeurs retenues pour  $qs$  et  $qt$  sont respectivement 0.5 %, 2 %, 8 %, 16 % et 32% de l'espace et 1 %, 5 %, 10 %, 30 % et 50 % du temps. La taille du nœud est fixée à 1024 octets dans tous les cas. De plus, l'implémentation utilise la variante de *R-tree* qui améliore ses qualités, appelée *R\*-tree* dans Beckmann et al. (1990).

En résultat, la FIG. 5 compare le coût de requêtes d'agrégation de chaque méthode d'indexation en faisant varier la taille du facteur "espace"  $qs$ , tant dis que les résultats illustrés dans la FIG. 6, illustrent l'impact de la variation du facteur temps  $qt$ . On peut noter que le *TTR-tree* est toujours plus performant que le *3DR-tree* et est également plus performant que l'index *Pfoser* pour tous les jeux de données.

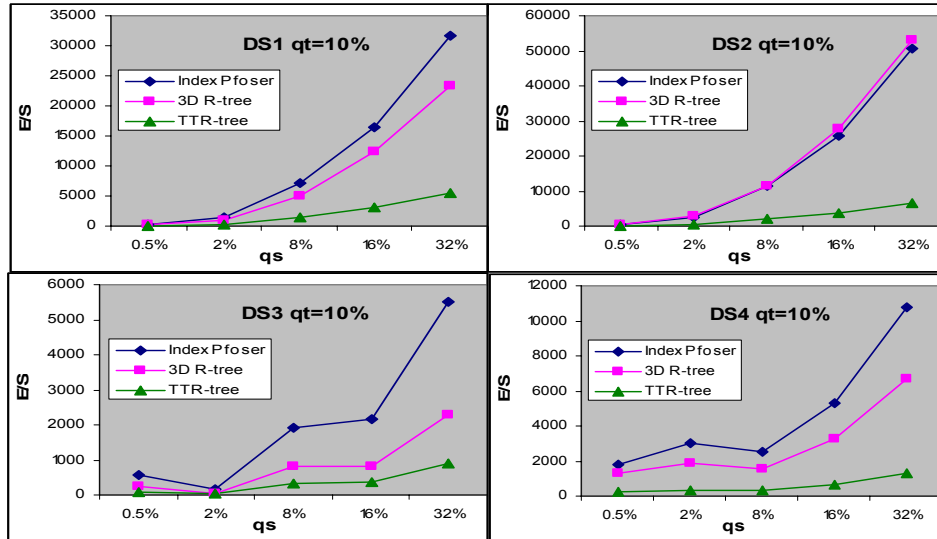


FIG. 5 – Coût de requêtes agrégats spatiotemporelles en faisant varier l'espace.

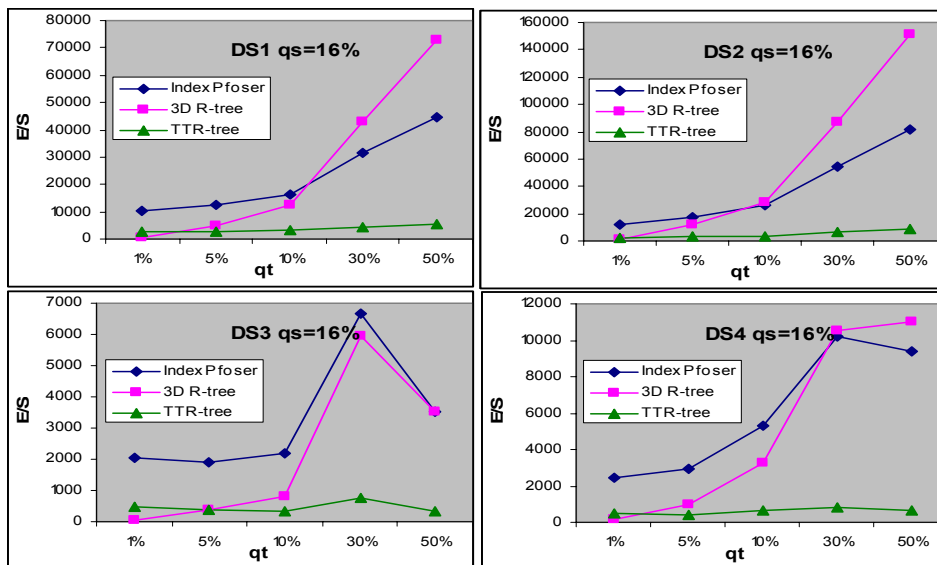


FIG. 6 – Coût de requêtes agrégats spatiotemporelles en faisant varier le temps.

En effet, le coût d'accès du 3D R-tree et de P foser augmente beaucoup plus que TTR-tree lorsque la taille de la fenêtre ( $qs$  ou  $qt$ ) augmente. La raison à cela est qu'une fenêtre plus grande englobe plus d'objets indexés et que les deux premières méthodes accèdent jusqu'au niveau le plus bas de l'arbre, tandis que le TTR-tree met fin à sa recherche dès que le nœud intermédiaire est contenu dans la fenêtre. On constate par ailleurs, que le coût d'exécution

des requêtes de *TTR-tree* varie très peu avec la variation de l'espace  $qs$ . Cependant, la méthode de *Pfoser* explose lorsque la taille de  $qs$  est supérieure à 32 %. La raison est que, contrairement à *TTR-tree*, la méthode de *Pfoser* ne permet pas le regroupement des codifications des tronçons lors de la transformation 1D car les codes de Hilbert qu'il utilise ne produisent pas des numéros de tronçons consécutifs. Donc, plus l'espace couvert par la requête est grand, plus celle-ci est traduite par une multitude de sous-requêtes et entraîne une multiplication des temps d'exécution.

## 6. Conclusion et perspectives

Cet article a proposé un nouveau modèle multidimensionnel pour objets mobiles contraints par le réseau. Ce modèle capture les objets mobiles continus et permet de les analyser conjointement avec leurs attributs. Il exploite la localisation sur le réseau pour réduire d'une dimension la représentation des trajectoires mobiles réduisant également l'espace de stockage. Nous avons étendu les notions de dimension et de fait à la variation continue. Nous avons ensuite explicité la représentation interne en terme d'intervalles et proposé une indexation et une implémentation de l'algorithme d'agrégat spatiotemporel dans ce contexte. L'index proposé est une combinaison d'un index spatial et des index bitmap. Une optimisation du stockage par compression de bitmaps a été discutée. L'efficacité de cette implémentation optimisée a été montrée par l'étude expérimentale. Il n'existe pas d'approche similaire dans l'état de l'art actuel. La combinaison des dimensions discrètes et spatiotemporelle dans les requêtes d'agrégation est tout à fait possible et serait optimisée si l'on génère au préalable des index bitmap sur ces dimensions discrètes. Faute de données adéquates, nous n'avons pu le tester pour le moment.

Une première perspective est justement d'étendre le générateur de Brinkhoff par une simulation d'attributs d'objets mobiles afin d'appliquer ces tests. Une seconde perspective est d'étendre ce modèle d'entrepôt d'objets mobiles aux fonctions agrégats complexes portant sur les trajectoires mobiles elles-mêmes et non par fenêtre spatiotemporelle. Il faudra définir la sémantique de telles fonctions, le type d'objet retourné et leur implémentation. Egaleme nt identifiée dans Fernando et al. (2004), nous pensons que le cadre OLAP pour objets mobiles est tout à fait approprié pour cette perspective de recherche.

## Références

- Agrawal, R., A. Gupta, and S. Sarawagi (1997). Modelling multidimensional databases. ICDE, 232-243.
- Ahmed, T., M. Miquel and R. Laurini (2004). Continuous Data Warehouse: Concepts, Challenges and Potentials. 12th International Conference on Geoinformatics, Geospatial Information Research: Bridging the Pacific and Atlantic.
- Beckmann, N., H. Kriegel, R. Schneider, B. Seeger (1990). The R\*-tree: an Efficient and Robust Access Method for Points and Rectangles. SIGMOD.
- Brinkhoff, T. (2002). Network-based Generator of Moving Objects. <http://fh-oow.de/institute/iapg/personen/brinkhoff/generator/>

- S. Chaudhuri and U. Dayal (1997). An Overview of Data Warehousing and OLAP Technology. *ACM SIGMOD*, 26(1): 65-74.
- Chon, H. D., D. Agrawal, A. El Abbadi (2002). Query Processing for Moving Objects with Space-Time Grid Storage Model. *Mobile Data Management*, 121.
- Fernando, I., V. Lopez, R. T. Snodgrass, and B. Moon (2004). Spatiotemporal Aggregate Computation: A Survey, *TIMECENTER TR-77*.
- Gray, J., A. Bosworth, A. Layman, and H. Pirahesh (1996). Data cube: A relational aggregation operator generalizing group-by, cross-tab, and sub-total. *ICDE*, 152–159.
- Güting, R., M. Böhlen, M. Erwig, C. Jensen, N. Lorentzos, M. Schneider, M. Z. Vazirgiannis (2000). Foundation for Representing and Querying Moving Objects. *ACM Transactions on Database Systems*, 25(1):1–42.
- Ho, C., R. Agrawal and N. Megiddo, R. Srikant (1999). Range Queries in OLAP Data Cubes. *ACM SIGMOD*, 73-88.
- Christian S. Jensen, A. Kligys, T. B. Pedersen, I. Timko (2004). Multidimensional data modelling for location-based services. *The VLDB Journal*, 13(1):1-21.
- Jurgens, M. and H. J. Lenz. The R a –tree (1998). An Improved R-tree with Materialized Data for Supporting Range Queries on OLAP-Data. *DEXA Workshop*.
- Marchand, P., A. Brisebois, Y. Bédard and G. Edwards (2004). Implementation and evaluation of a hypercube-based method for spatio-temporal exploration and analysis. *International Society for Photogrammetry and Remote Sensing (ISPRS) Journal*, 59(1-2): 6-20.
- Ooi, B.C., K.J. Mcdonell., R. Sack-davis. Spatial kd-tree (1987) An indexing mechanism for spatial databases. *IEEE COMPSAC – Comp. Software&Applications Conf.* 433-438.
- Papadias, D., Y. Tao, P. Kalnis, J. Zhang (2002). Indexing Spatio-Temporal Data Warehouses. *ICDE*, 166-175.
- Pfoser, D., C. S. Jensen, Y. Theodoridis (2000). Novel Approaches in Query Processing for Moving Object Trajectories. *VLDB*, 395-406.
- Pfoser, D. and C.S. Jensen (2003). Indexing of Network-constrained MOs. *ACM GIS*, 25-32.
- Saltenis, S., C. S. Jensen, S. T. Leutenegger, M. A. Lopez (2000). Indexing the Positions of Continuously Moving Objects. *ACM SIGMOD*, 331-342.
- Shanmugasundaram, J., U. Fayyad, and P. Bradley (1999). Compressed data cubes for OLAP aggregate query approximation on continuous dimensions. *KDD*, 223-232.
- Stefanovic, N., J. Han, and K. Koperski (2000). Object-based selective materialization for efficient implementation of spatial data cubes. *IEEE Transactions on Knowledge and Data Engineering*, 12(6): 938-958.
- Tao, Y., D. Papadias (2002). Time-Parameterized Queries in Spatio-Temporal Databases. *ACM SIGMOD*, 334-345.

## Représentation et indexation d'objets mobiles dans un entrepôt

- Tao, Y., G. Kollios, J. Considine, F. Li and D. Papadias (2004). Spatiotemporal Aggregation Using Sketches. ICDE, 214-225.
- Vazirgiannis, M. and O. Wolfson (2001). A Spatiotemporal Query Language for Moving Objects. Conference on Spatial and Temporal Databases, 20-35.
- Wan, T. et K. Zeitouni (2005). Modélisation d'objets mobiles dans un entrepôt de données. 5èmes EGC, Edition CEPADUES.
- Wu, K., J. Otoo Ekow, A. Shoshani (2004). An efficient compression scheme for bitmap indices. Lawrence Berkeley National Laboratory Report LBNL-49626.
- Zhang, D., A. Markowetz, V.J. Tsotras, D. Gunopulos and B. Seeger (2001). Efficient Computation of Temporal Aggregates with Range Predicates, SIGMOD PODS, 237–245.
- Zhang, V. J. Tsotras and D. Gunopulos (2002). Efficient Aggregation over Objects with Extent, SIGMOD PODS, 121-132.

## Summary

The rapid growth of geo-location techniques and mobile devices led to the profusion of Mobile Objects (MO) databases. This raises a new issue concerning their use for decision support. While conventional On-Line Analytical Processing (OLAP) systems are efficiently used in data analysis thanks to their multidimensional modelling, they are not adapted to MOs which consider information that evolves continuously over time (e.g., position). In this paper, we consider network constrained MOs and make the three following contributions. We first propose a conceptual model that extends the multidimensional model to continuous dimensions and facts. Then, we derive an efficient data structure. Third, we describe an index structure and algorithms which optimizes typical spatiotemporal OLAP queries for such moving objects. A prototype has been implemented and the experimental results, detailed in the paper, show the efficiency of the proposed methods.