



Patterns de mobilité

Thèse soutenue par Cédric du Mouza

le 12 octobre 2005

sous la direction de Philippe Rigaux

Equipe VERTIGO - Lab. CEDRIC - CNAM Paris



Contexte



Manipuler *efficacement* des données de nature *complexe* (objets mobiles)

Problématiques classiques:

- indépendance logique/physique: on souhaite exprimer de manière simple des opérateurs puissants
- performances: quelles structures adopter afin d'avoir une évaluation efficace des requêtes

Domaine applicatif choisi: problématique assez récente du suivi et de l'interrogation d'objets mobiles.



Motivations



Convergence de nouvelles technologies:

- Internet
- Communications sans fil
- GPS (et Galileo?)

⇒ nouveaux défis pour la recherche comme:

- interroger des données historisées représentant des trajectoires
- suivre des objets mobiles en temps réel



Contribution



Cette thèse propose:

- un langage de modélisation et d'interrogation de trajectoires: étude de l'expressivité et de la complexité
- un algorithme très efficace pour une partie bien définie du langage
- un modèle de représentation et d'interrogation multi-échelle
- un prototype



Problématique



Hypothèses:

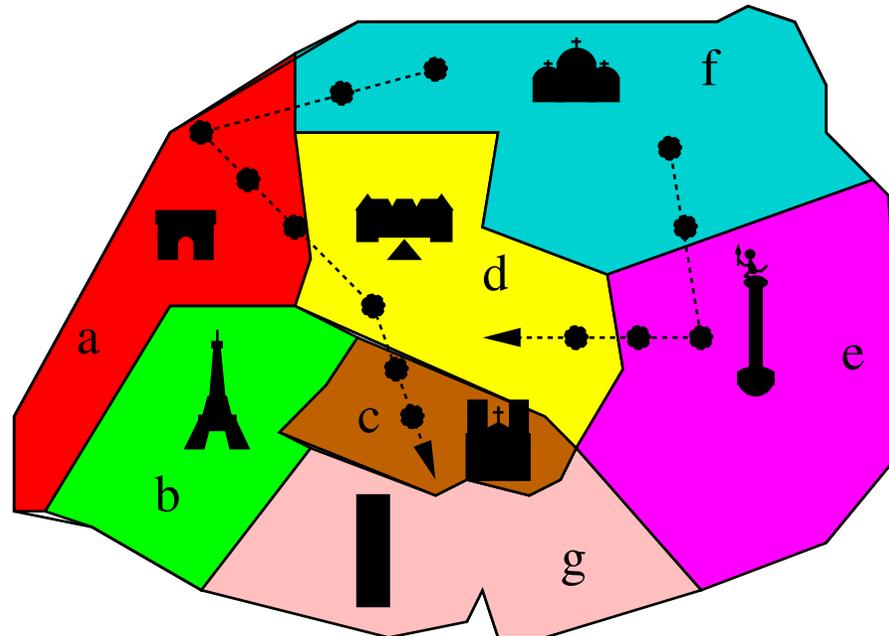
- l'espace de référence est une carte **partitionnée** en zones
- les trajectoires sont représentées par des **séquences** de zones traversées

Objectifs:

- définir un modèle de données et un langage de requêtes pour ces trajectoires
- développer des techniques pour une évaluation **continue** des requêtes

 *Intuition: utiliser des patterns pour l'interrogation.*

Données



- Une carte partitionnée en 7 zones: a, b, c, d, e, f et g
- chaque point représente un nouvel événement GPS, reçu à une fréquence donnée
- 2 trajectoires $f \rightarrow a \rightarrow d \rightarrow c$ et $f \rightarrow e \rightarrow d$



Requêtes



- **Q1.** Donner tous les objets qui ont voyagé de a à f , sont restés au moins deux minutes en f et ensuite ont voyagé de f à c .
- **Q2.** Donner tous les objets qui restent en a ou b excepté pendant une minute où ils furent dans une autre zone.
- **Q3.** Donner tous les objets qui sont passés par f pour atteindre une autre zone, puis sont allés en d ou c , et sont revenus en f via la *même* zone.



Variables



Une requête correspond donc à une spécification de zones successives traversées avec un certain degré de liberté et des contraintes temporelles.

Comment représenter ces « degrés de liberté » ?
⇒ utilisation de **variables**.

Une variable peut être associée à **n'importe quel** symbole.

Si une variable se répète: toutes les répétitions doivent être liées au **même symbole**



Variables



Exemple: Donner tous les objets qui restent en a ou b excepté pendant une minute où ils furent dans une zone $@x$ différente de a ou b .

Exemple: Donner tous les objets qui sont passés par f pour atteindre une autre zone $@x$, puis sont allés en d ou c , et sont revenus en f via la *même* zone $@x$.

Exemple: Donner tous les objets partant d'une zone $@x$ pour aller en a , puis traversant une zone $@y$ différente de $@x$ avant de revenir en a .



Patterns de mobilité

Langage: adaptation des expressions régulières à la sémantique de l'application.

Pourquoi pas toute E.R.?

$E = b . (a \mid @x) + . c$, alors la trajectoire $b . a . c$ a deux représentants dans $\mathcal{L}(E_1)$:

- $b . a . c$
- $b . @x . c$ (avec $@x$ instanciée à a)

⇒ non déterministe, et parfois n'a aucun sens.

Un **pattern de mobilité** est une expression régulière sur $\Sigma \cup \mathcal{V}$ où chaque variable joue un rôle dans l'évaluation.

Interrogation

Syntaxe: une *requête* est un pattern accompagné d'un ensemble de contraintes pour les variables.

Sémantique: un objet appartient au résultat d'une requête si les **derniers** déplacements correspondent au pattern tout en respectant les contraintes.

Exemple: donner tous les objets qui ont traversé une certaine zone différente de c pour aller en d et après 2 minutes sont allés en f en passant par cette *même* zone.

$$q = (@x . d [2] . @x . f, \{ @x \neq c \})$$

Alors l'objet de trajectoire $d . c [2] . a . d [2] . a . f$ appartient au résultat.

Évaluation



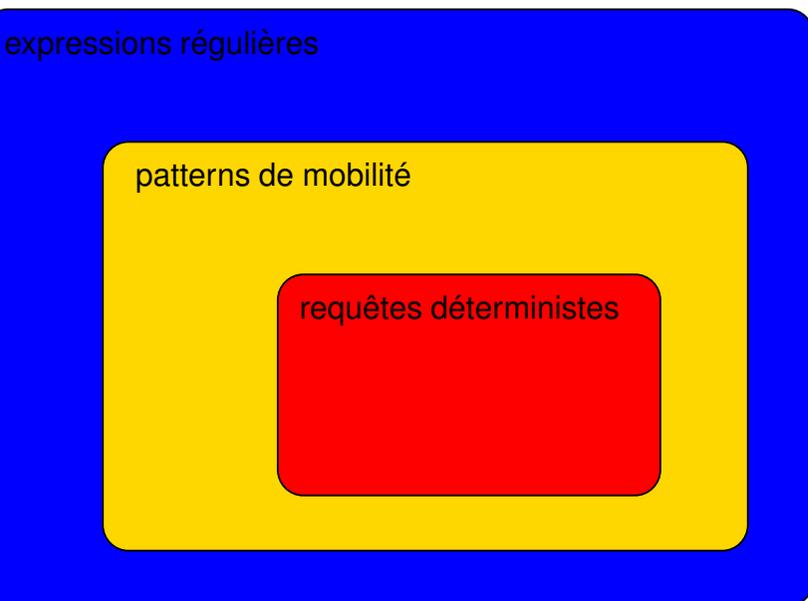
Évaluation de $(a|b)^+ . @x . (a|b)^+$

| entrée | chaîne possible | instanciation |
|----------------------|---|---------------------------------|
| a | a | @x=⊥ |
| a . a | a . a a . @x | @x=⊥, @x=a |
| a . a . b | a . a . b a . a . @x a . @x . b | @x=⊥, @x=b, @x=a |
| a . a . b . b | a . a . b . b a . a . b . @x a . a . @x . b a . @x . b . b | @x=⊥, @x=b, @x=a, @x=b |



Expression vs complexité

Dans le pire des cas, le nombre d'états à maintenir est **exponentiel** dans le nombre de variables.



expressions régulières: n'ont pas toujours de sens
patterns de mobilité: ont un sens mais grande complexité
requêtes déterministes: moins expressives mais peu complexes

rendant l'instanciation des variables déterministe, le besoin en mémoire devient $|P| + |var(P)|$.

Exemple d'évaluation

Évaluation de $((a|b)^+.\@x.(a|b)^+, \{\@x \neq a, \@x \neq b\})$

| entrée | chaîne possible | chaîne impossible |
|--------------------------|---------------------|---|
| a | a | |
| a . a | a . a | a . \@x car $a \notin \text{dom}(\@x)$ |
| a . a . b | a . a . b | a . \@x . b car $a \notin \text{dom}(\@x)$ a . a . \@x car $b \notin \text{dom}(\@x)$ |
| a . a . b . b | a . a . b . b | a . \@x . b . b car $a \notin \text{dom}(\@x)$ a . a . \@x . b car $b \notin \text{dom}(\@x)$ a . a . b . \@x car $b \notin \text{dom}(\@x)$ |
| a . a . b . b . c | a . a . b . b . \@x | |

Principaux résultats



- un modèle de représentation des trajectoires sur un espace discrétisé
- un langage d'interrogation, sous-ensemble décidable des expressions régulières avec variables, expressif mais gourmand en mémoire ($|etats(\mathcal{A})| \times |\Sigma^k|$ statuts possibles par objet et par requête)
- un sous-ensemble du langage bien déterminé (décidabilité) garantissant de faibles besoins en mémoire ($|P| + |var(P)|$)

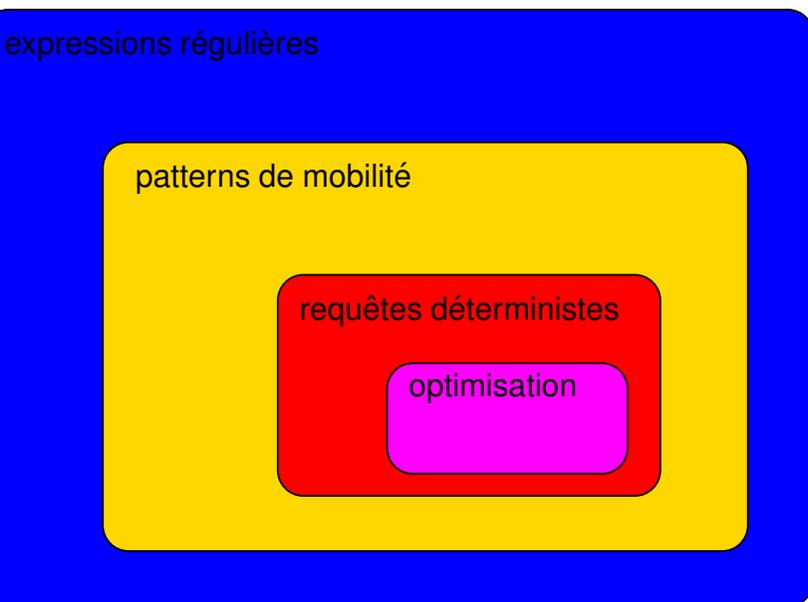
Ces travaux ont été présentés à *STDBM'04* et *GeoInformatica'05*.



Optimisation

Motivation: réception d'un flux *continu* d'information, pas le temps de re-tester des symboles.

Objectif: évaluation d'une requête en ne testant qu'une fois chaque symbole.



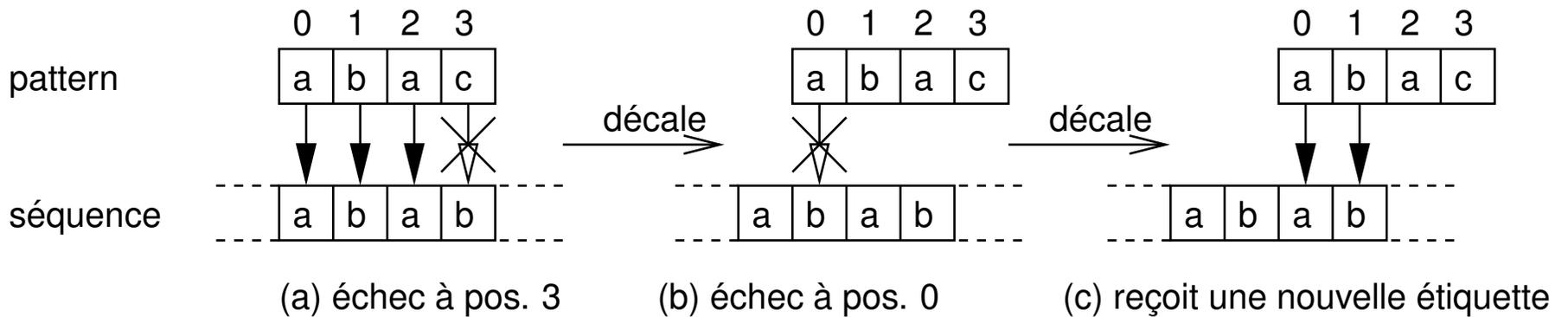
On évalue les patterns de la forme $t_1.t_2 \cdots t_n$ où $t_i \in (\Sigma \cup \mathcal{V})$.

Problème: on sait le faire sans variable (KMP), comment le faire avec des variables?

Évaluation sans variable



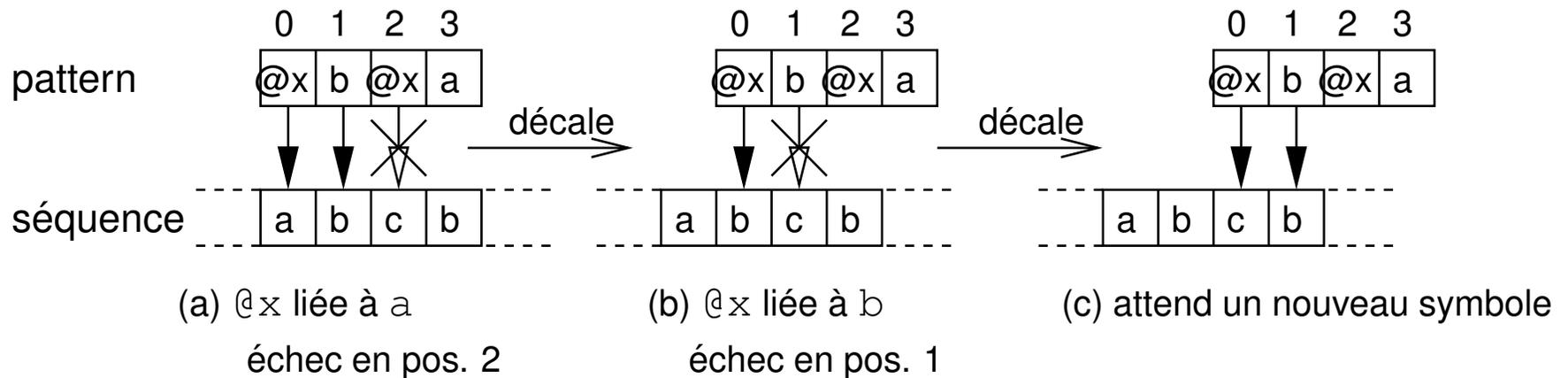
Technique naïve sans variable: en cas d'échec on décale de 1 et on recommence.



Évaluation avec variables



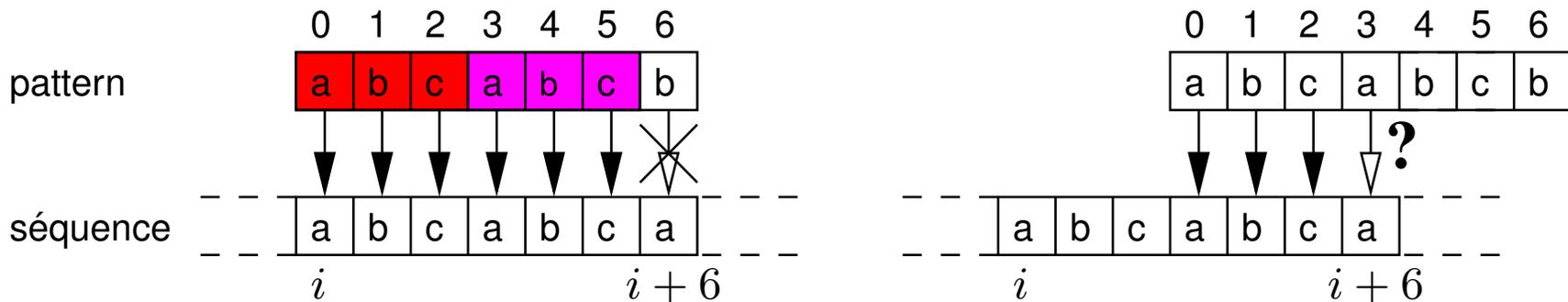
Technique naïve avec variable: en cas d'échec on décale de 1 et on recommence.



Intuition de l'optimisation



En cas d'échec, certaines comparaisons peuvent être évitées (**KMP**).



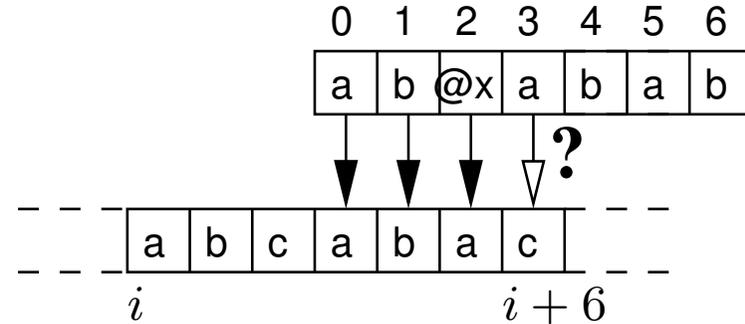
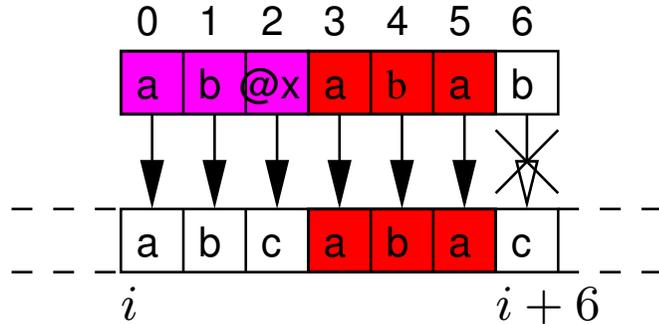
⇒ avant l'exécution il est possible de connaître les décalages à faire en cas d'échec.



KMP étendu



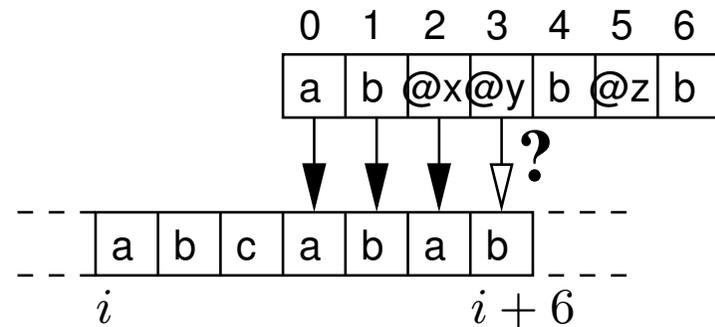
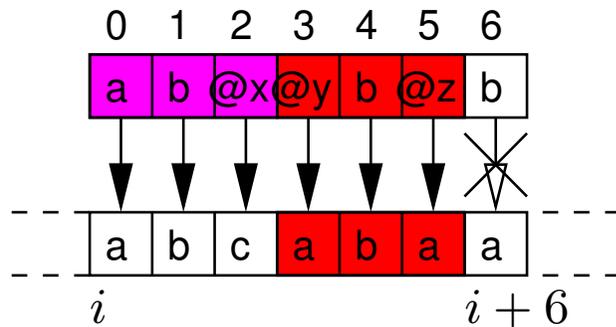
Comment adapter KMP pour nos patterns?



L'instanciation des variables du **préfixe** peut changer après un décalage.



KMP étendu



L'instanciation des variables du **suffixe** à une importance pour trouver le bon décalage.



Bords



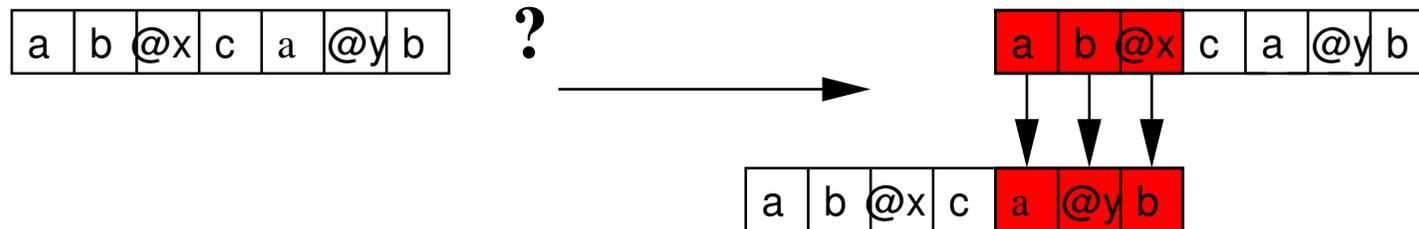
Un **bord** d'un pattern est l'information composée de:

- les conditions sur l'instanciation des variables pour effectuer le décalage
- la longueur du décalage à réaliser
- l'instanciation en sortie des variables du pattern

Exemple: si $(3, \{@y = b\}, \{@x = b\})$ est un bord pour un pattern P de longueur m , alors en cas d'échec, si $@y$ est instanciée à b , alors on peut décaler pour reprendre au 4ème symbole et $@x$ devient instanciée à $b \rightarrow$ bord

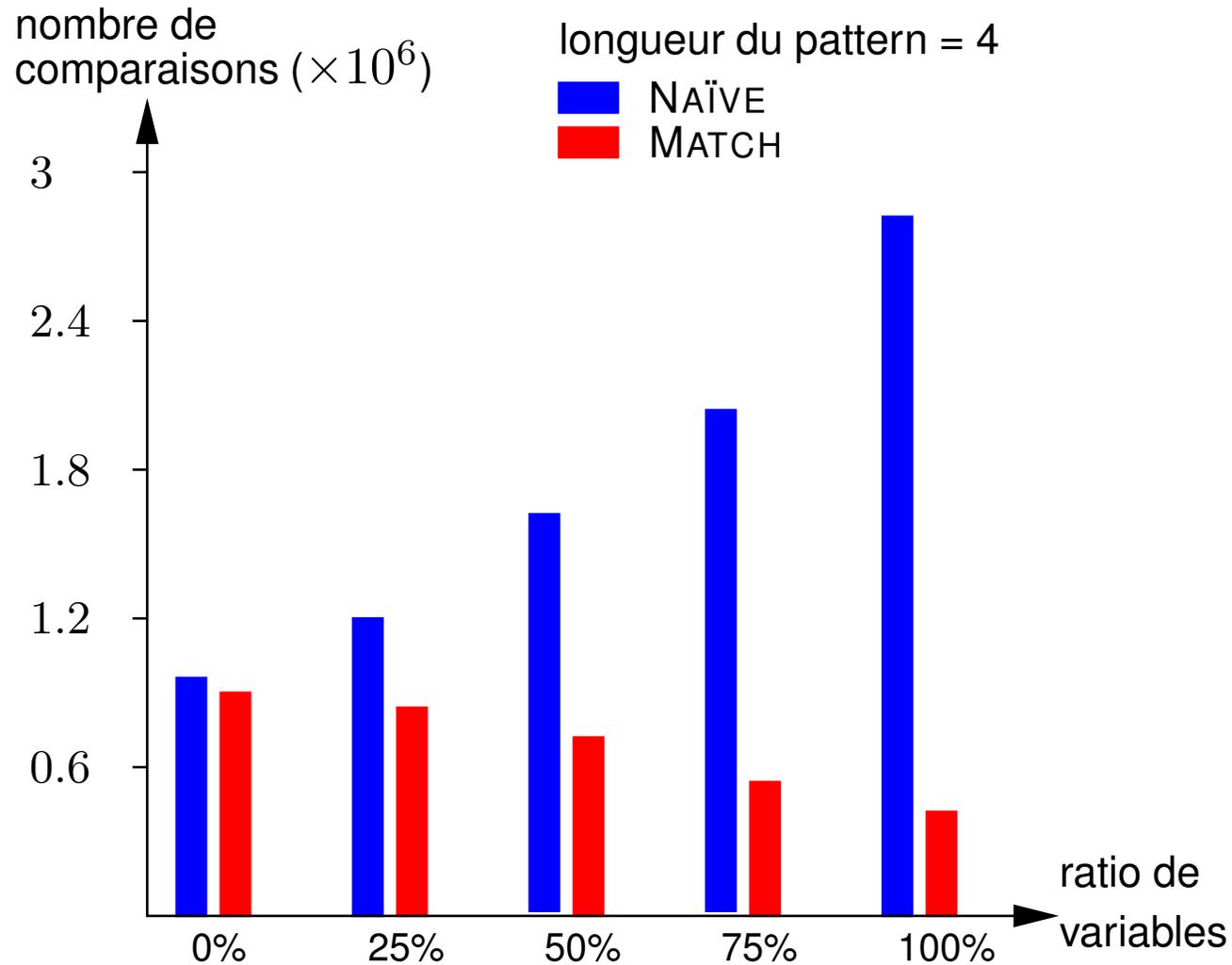


Exemples de bords



- $(0, \emptyset, \emptyset)$ bord par défaut (décalage de tout le pattern);
- pas de bord de longueur 1 existant car $a \neq b$
- $(2, \{\text{@y/a}\}, \emptyset)$, car si `@y` était instanciée à `a`, alors on peut faire le décalage de 2
- $(3, \{\text{@y/b}\}, \{\text{@x/b}\})$, car si `@y` était instanciée à `b`, alors on peut faire le décalage de 3 et `@x` sera alors instanciée à `b`

Expérimentations



Résumé des contributions



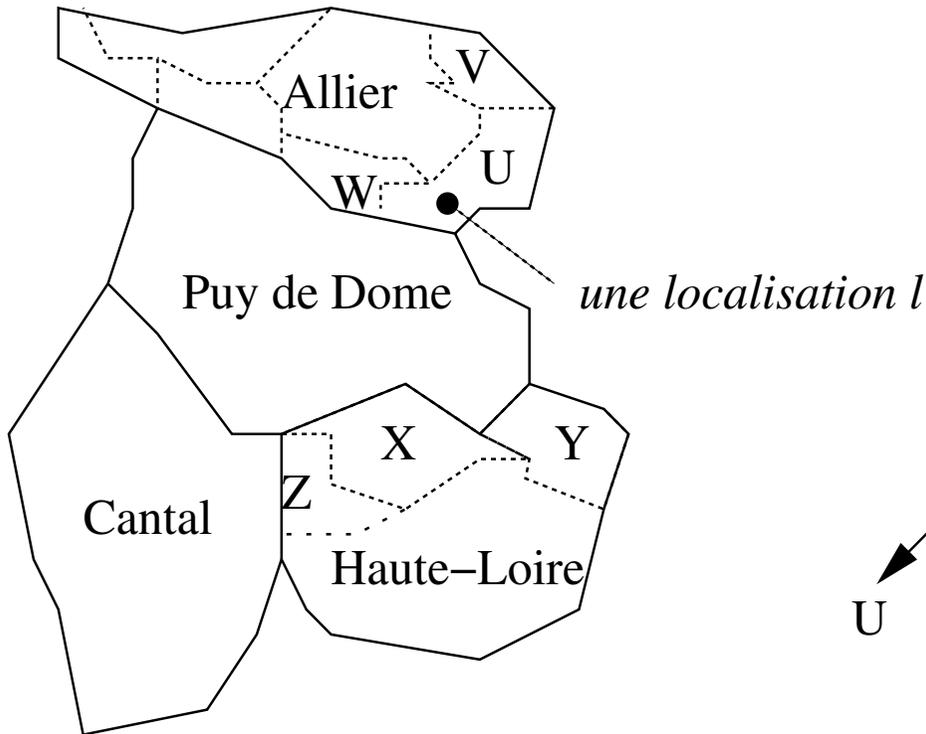
- identification d'un sous-ensemble de patterns de mobilité
- définition d'un bord
- algorithme de construction quadratique de la table des bords
- complexité linéaire de l'évaluation au lieu d'une évaluation naïve quadratique
- expérimentations

Ces résultats ont été présentés à *CIKM'05*.

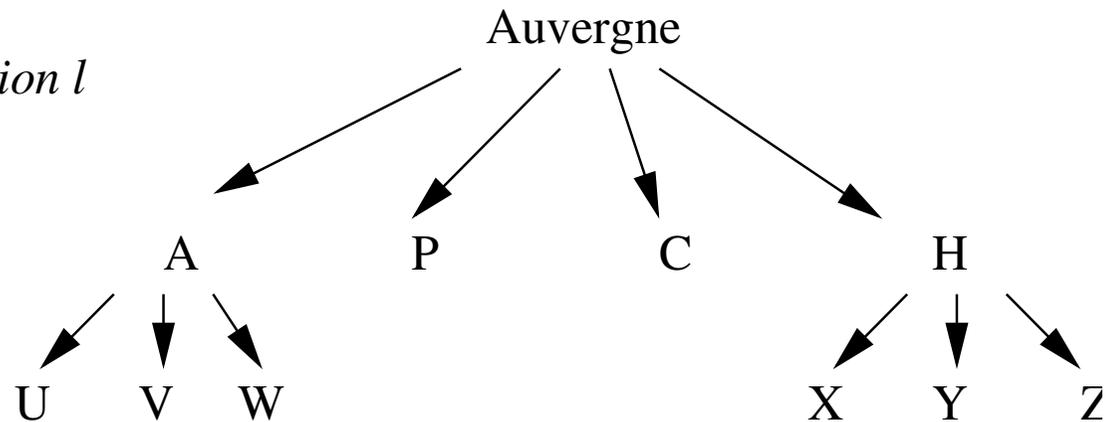


Autre aspect abordé: multi-échelle

L'espace considéré est partitionné à différents niveaux de granularité \Rightarrow arbre d'inclusion pour les zones.



a. l'espace de référence



b. sa représentation

Intérêt des niveaux d'échelle



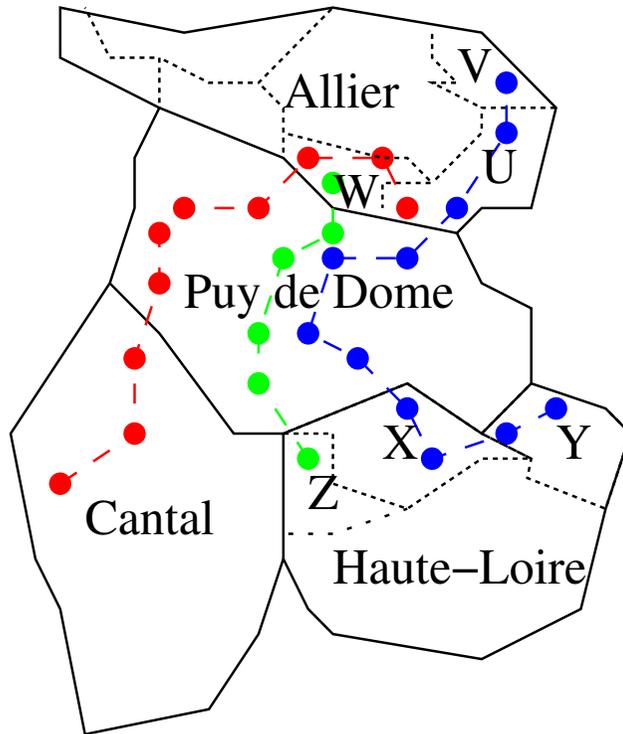
Les niveaux d'échelle permettent:

- d'adapter l'espace considéré aux besoins de l'utilisateur
- en passant à un niveau plus grossier, d'agréger des trajectoires ayant un comportement proche mais distinct à un niveau plus fin
- en passant à un niveau plus fin, de distinguer des trajectoires ayant un comportement semblable à un niveau plus grossier



Exemples

Soit $N1$ le niveau d'échelle des départements et $N3$ celui mixte villes/départements.



$pattern(t1, N1) = A.P.C$

$pattern(t1, N3) = U.W.P.C$

$pattern(t2, N1) = H.P.A$

$pattern(t2, N3) = Y.X.P.U.V$

$pattern(t3, N1) = H.P.A$

$pattern(t3, N3) = Z.P.W$

Clustering



Objectif: utiliser un pattern plus *générique* pour filtrer les événements à tester sur un ensemble de patterns.

Pourquoi? Si une trajectoire ne satisfait pas le pattern *générique*, aucune chance de satisfaire les patterns plus précis.

Exemple: si une trajectoire ne satisfait pas le pattern *H.P.A* elle ne peut satisfaire ni *Brioude-Riom-Allier*, ni *Haute-Loire - Puy de Dôme - Vichy*, ni *Haute-Loire - Issoire - Montluçon*



Résumé des contributions

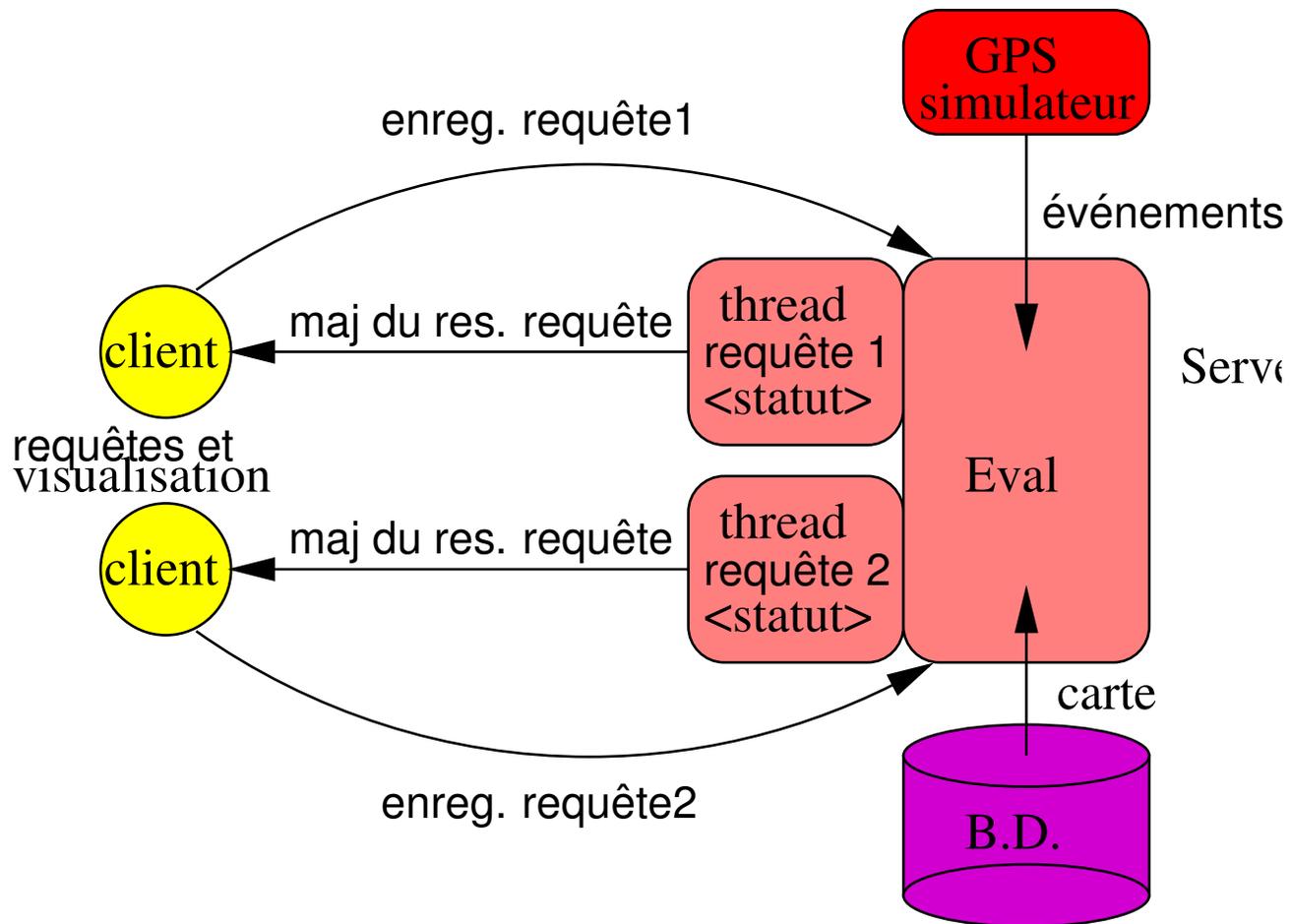


- description d'une carte multi-échelle et définition d'un niveau d'échelle
- relation d'ordre partielle entre patterns et calcul de *lub*
- technique de classification et de clustering de trajectoires s'appuyant sur le multi-échelle

Ces travaux ont été présentés à *SSDBM'04*.



Implantation



Perspectives



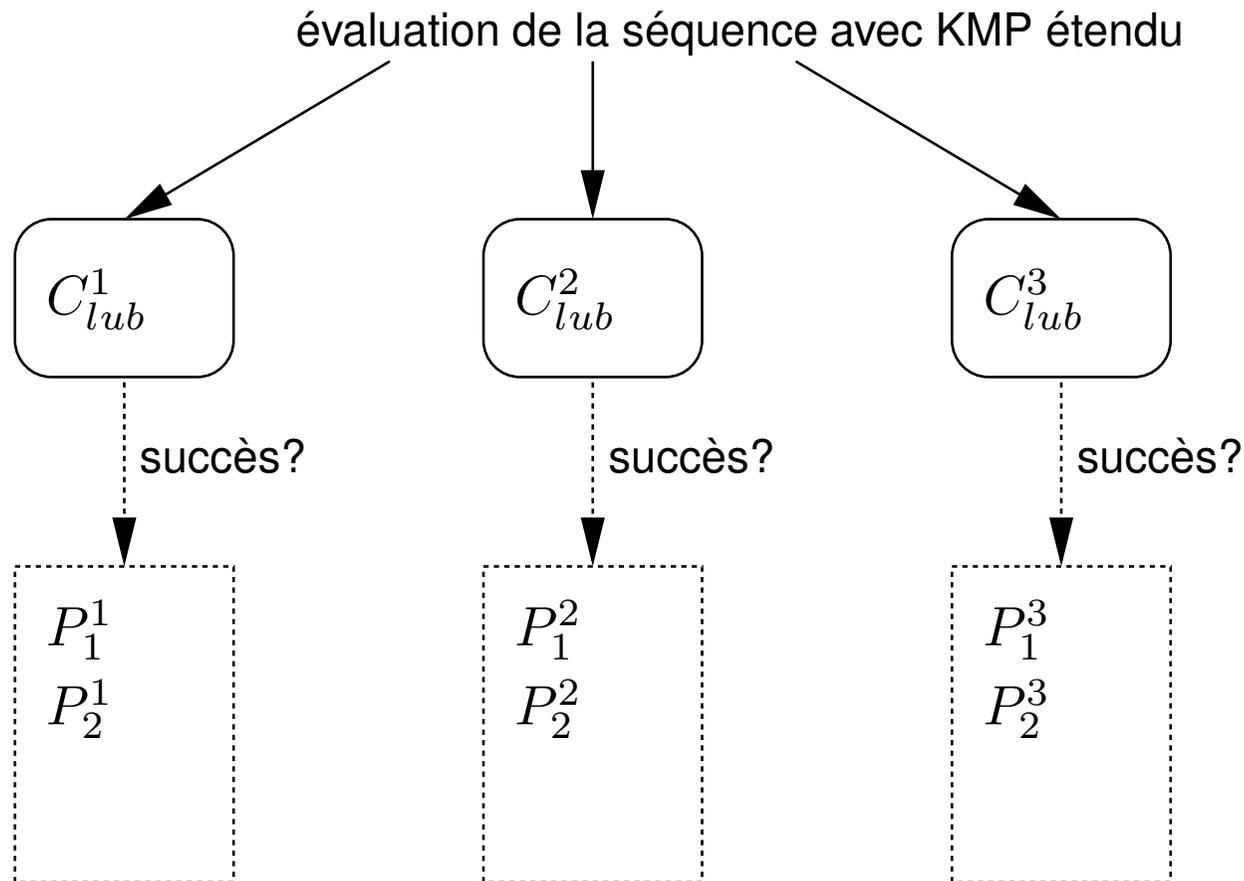
Les trois axes de ma recherche actuelle et future sont:

- le clustering de patterns avec variables
- un nouvel index: le SDR tree
- l'indexation d'images



Perspectives: clustering

Objectif: trouver une structure pour optimiser l'évaluation quand il y a beaucoup de patterns



Perspectives: SDRtree



Travail sur un index spatial ayant les propriétés suivantes:

- **piloté par les données**, avec un algorithme d'éclatement semblable au Rtree
- **binaire** car chaque nœud interne couvre exactement 2 nœuds
- **équilibré**
- **distribué** parce qu'il y a une correspondance un-à-un entre les feuilles et les sites
- **problème**: minimiser le nombre de messages échangés entre clients/serveurs et serveurs/serveurs pour maintenir et utiliser la structure



Merci



Merci pour votre attention.

