UQAC

Université du Québec À Chicoutimi

Mémoire Présenté à

l'Université du Québec à Chicoutimi

comme exigence partielle

de la maîtrise en informatique

Par

Mohamed Tarik MOUTACALLI

# Une approche de reconnaissance d'activités utilisant la fouille de données temporelles

Août 2012

## Résumé

L'assistance d'une personne atteinte de la maladie d'Alzheimer est une tâche difficile, couteuse et complexe. La relation est souvent compliquée entre l'aidant et le patient qui souhaite préserver son intimité. Avec l'émergence du domaine de l'intelligence ambiante, les recherches se sont orientées vers une assistance automatisée qui consiste à remplacer l'assistant par un agent artificiel. Le plus grand défi de cette solution réside dans la reconnaissance, voire la prédiction, de l'activité du patient afin de l'aider, au besoin et au moment opportun, à son bon déroulement. Le nombre très élevé des activités de la vie quotidienne (AVQ) que le patient peut effectuer, ce que nous appelons aussi le nombre d'hypothèses, complique grandement cette recherche. En effet, comme chaque activité se compose de plusieurs actions, cette recherche se traduit donc par la recherche de la ou les actions effectuées par le patient parmi toutes les actions de toutes ses activités et ce, en un temps réel.

Ce mémoire de maîtrise explore les techniques de la fouille de données temporelles afin de proposer une réponse à cette problématique en essayant de réduire au maximum, à un instant précis, le nombre d'hypothèses. Le travail débute par une analyse de l'historique des actions du patient pour créer des plans d'activités. Des plans propres à chaque patient et qui spécifient la liste ordonnée des actions qui composent une activité. Ensuite, une segmentation temporelle est effectuée sur chaque plan créant un ou plusieurs intervalles temporels résumant les périodes de commencement de l'activité. La troisième étape consiste à implémenter un système de reconnaissance d'activité pour trouver, à tout instant, l'activité la plus probable. Ce travail se base essentiellement sur l'aspect temporel et n'offre pas juste une solution pour la reconnaissance d'activité, mais il répond aussi aux erreurs d'initiations. Des erreurs susceptibles d'être commises par les malades d'Alzheimer et qui n'ont jamais été traitées par aucune autre recherche. Les tests et validations de notre approche ont été effectués avec des données réelles enregistrées après l'observation de réels malades d'Alzheimer et les résultats sont très satisfaisants.

## Remerciements

En préambule à ce mémoire, je tiens à remercier mes directeurs de recherches, M.Abdenour BOUZOUANE et M.Bruno BOUCHARD, qui se sont toujours montrés à l'écoute et très disponible tout au long de la réalisation de ce mémoire, ainsi que pour le soutient, l'encouragement, l'inspiration, l'aide et le temps qu'ils ont bien voulu me consacrer.

Mes remerciements s'adressent également à Monsieur Stephen WHITNEY, pour sa disponibilité et son aide grandement apprécié. Je me dois aussi de souligner l'appui quotidien des membres du département d'informatique et de mathématique.

Je n'oublie pas mes parents pour leur contribution, leur soutien, leur patience et leur confiance qui m'ont alloué.

# Tables des Matières

# Liste des Figures

# Liste des Tableaux

# Liste des Algorithmes

# Chapitre 1

# Introduction

## 1.1. Contexte

Aujourd'hui au Canada, plus de 500.000 personnes souffrent de la maladie d'Alzheimer [1]. Un chiffre qui, d'après les mêmes études, ne cessera d'augmenter et pourra atteindre 1 à 1.5 millions après juste une génération causant une perte à l'économie canadienne d'environ 218.6 billion de dollars en 2038. La majeure partie de ces coûts vient du besoin intrinsèque des patients atteints de cette maladie d'une personne aidante pour les assister dans leurs activités de vie quotidienne. Ces chiffres alarmants ont incité les chercheurs à explorer plusieurs approches pour réduire l'impact de ce fléau sur la société canadienne. Comme le domaine de la santé n'arrive toujours pas à offrir une solution efficace à ce problème, les recherches se sont orientées vers un domaine émergeant qui est l'intelligence ambiante [2].

L'intelligence ambiante (IAm), vise à améliorer le quotidien des personnes en introduisant, d'une façon transparente à l'utilisateur, des outils technologiques miniaturisés dans leur vie de tous les jours. Donc, l'idée générale de ces recherches est de transformer l'habitat du

patient en un habitat dit intelligent [3] afin d'alléger la charge de travail des aidants. L'habitat est alors truffé de senseurs : des capteurs installés sur des objets et émettent des informations sur son état, une étiquette RFID sur une tasse qui envoie des informations sur sa position ou un capteur sur une lampe signalant si elle est allumée ou éteinte etc, et des effecteurs lumineux, audio ou vidéo qui servent à orienter l'utilisateur vers la forme d'aide que nous voulons lui apporter. Entre l'envoi des informations par les capteurs et l'aide proposée par les effecteurs vient la phase dite intelligente. Une phase où nous nous heurtons à une problématique importante, celle qui consiste à analyser les informations envoyées par les capteurs pour trouver l'activité entamée par le patient afin de l'aider, au besoin, à son bon déroulement. Le nombre très élevé des activités que le patient peut effectuer, que nous appelons aussi nombre d'hypothèses, rend la détection de l'activité en temps réel une mission très compliquée.

## 1.2. Problématique générale

Fournir une assistance automatisée et ponctuelle au patient passe forcément par la reconnaissance de l'activité qu'il est en train d'effectuer [3]. Cette problématique de reconnaissance a fait l'objet de plusieurs recherches [3, 7, 8, 9, 10, 11] et a été traitée de différentes façons. La façon la plus naturelle pour résoudre ce problème est de parcourir tous les plans d'activités jusqu'à ce que nous trouvions celui qui correspond le plus aux actions observées. Un plan d'activités spécifie la liste ordonnée des actions qui composent une activité; alors qu'une action peut se composer d'un ou plusieurs événements. Chacun de ces événements se traduit par un changement de valeurs ou d'état d'un capteur, par exemple le changement de la position de la tasse, signalé par l'étiquette RFID installée dessus, peut se traduire par l'action *prendre tasse* ou *remettre tasse* selon si la tasse se rapproche ou s'éloigne de la position du

patient. Les actions : *prendre thé*, *prendre tasse*, *ajouter de l'eau* peuvent par exemple définir le plan d'activité *préparer thé*, comme *prendre tasse*, *ajouter café* et *ajouter du sucre* peuvent définir le plan *préparer café*. Si nous ne possédons que ces deux plans et que l'action détectée est *prendre thé*, l'activité qui sera reconnue est la première : *préparer thé* et donc la prochaine action qui sera prédite correspondra à *ajouter de l'eau*. Malheureusement cet exemple est trop simpliste et le processus de la reconnaissance d'activité est beaucoup plus complexe. Cette complexité vient surtout du très grand nombre d'activités de la vie quotidienne (AVQ) qu'un patient peut effectuer dans sa journée, que nous appelons aussi nombre d'hypothèses, et qui rend impossible la détection de l'activité en temps réel. En effet, pour fournir une assistance au moment opportun, l'agent artificiel, appelé aussi agent ambiant, doit reconnaître instantanément l'activité entamée. Supposons que nous possédons **m** plans d'activités et que chacun d'entre eux est composé au maximum de **n** actions, le temps de recherche de l'activité sera donc dans l'ordre de **m** x **n**. alors, il est évident et indispensable de minimiser **m** pour accélérer cette recherche, ce qui se traduit par la réduction du nombre d'hypothèses. Il faut bien noter que cette réduction ne se réalise pas par l'élimination des activités que le patient a l'habitude d'effectuer, mais en diminuant le niveau de plausibilité de celles qui n'a pas l'habitude d'effectuer au moment où nous commençons cette recherche.

La réduction du nombre d'hypothèses peut généralement s'effectuer en se basant sur des informations spatiales ou temporelles. Si par exemple le patient se trouve au salon, nous pouvons ignorer les activités qui se déroulent ailleurs : dans la cuisine, la salle de bain etc. En ce qui concerne les informations temporelles, nous pouvons remarquer que la plus part des activités se déroulent pendant des périodes bien déterminées, par exemple l'activité *ranger lit* se déroule le matin entre 8h et 10h, alors elle peut ne pas être prise en considération, dans un premier lieu, si

l'heure actuelle est en dehors de cet intervalle. Il est à noter que chaque patient peut effectuer différentes activités, de différentes façons et à différents moments. En effet, Les plans d'activités d'un patient font parties de son profil. Chaque patient a son profil unique, décrivant, en autre, ses habitudes. Pour cette raison, ces intervalles sont créés après l'analyse des habitudes du patient et du temps de commencement de ses activités.

Plusieurs autres problèmes sont rencontrés pendant la création du système de reconnaissance d'activité dont la construction des plans d'activités. Si la plupart des recherches évitent ce problème en supposant qu'elles possèdent déjà ces informations capitales, nous avons choisi de traiter ce problème et de proposer une façon automatique et non supervisée basée sur l'analyse des données envoyées par les capteurs. Un autre problème, traité aussi dans notre travail, peut être constaté dans l'exemple cité ci-haut vient du fait qu'une ou plusieurs actions détectées peuvent appartenir à différents plans d'activités, *prendre tasse* appartient aux deux plans d'activités, et dans ce cas qu'elle activité doit être déduite ? En fin, les erreurs susceptibles d'être commises par les personnes assistées qui souffrent, rappelons-le, de la maladie d'Alzheimer causent d'additionnels problèmes à la reconnaissance d'activité.

## 1.3. La maladie d'Alzheimer

Décrite pour la première fois par Aloïs Alzheimer en 1907, la maladie d'Alzheimer est une dégénérescence progressive du cortex cérébral et d'autres zones du cerveau qui entraîne la démence [4]. Elle progresse lentement et conduit à la mort dans une période de huit à douze ans. La perte de mémoire est le symptôme précoce le plus frappant et le plus connu de cette maladie. D'autres effets plus désagréables se manifestent progressivement après, comme l'incapacité d'effectuer des AVQ, le changement rapide de la personnalité et du tempérament, la

désintégration lente de la personnalité, la perte graduelle de la maîtrise du corps, les hallucinations, le délire [4]...

Selon le nombre d'années, après le diagnostic évidemment, le patient est dans l'une des trois phases de la maladie : Initiale, Intermédiaire ou Avancée. Le Tableau 1.1, adapté de Gauthier [5], explique le niveau de dépendance du malade par phase.

| Phases | Durée (années) | Échelle de détérioration globale * | Échelle MEEM ** | Autonomie globale |
|---|---|---|---|---|
| Initiale | 2-3 | 3-4 | 26-18 | Vie autonome |
| Intermédiaire | 2 | 5 | 10-17 | Supervision requise |
| Avancée | 2-3 | 6-7 | 9-0 | Dépendance totale |

**Tableau 1.1: Évaluation de la dysfonction globale et cognitive associée aux trois phases de la maladie d'Alzheimer adapté de Gauthier [5]**

Dans la phase initiale, à la rubrique *Autonomie globale*, le Tableau 1.1 indique que le malade peut mener une vie autonome, mais les chiffres 3-4 dans la catégorie *Échelle de Détérioration globale* montrent que le malade a besoin d'aide pour ses activités quotidiennes (seul 1-2 n'en ont pas besoin). Durant cette phase de la maladie, le patient commence à avoir des petits problèmes de concentration diminuant son attention sur une tâche particulière sans vraiment l'empêcher de mener une vie normale.

---

*L'échelle de détérioration globale évalue le besoin d'aide progressif pour les activités quotidiennes (p.ex. choix des vêtements et aide pour se vêtir); le score varie de 1-2 (normal) à 6-7 (dysfonction grave).

**L'échelle du mini-examen de l'état mental (MEEM) en 22 points sert à évaluer la fonction cognitive; le score varie de 30 (fonctionnement excellent) à 0 (dysfonction grave).

Dans la phase intermédiaire, une supervision est requise car pendant cette période les troubles de mémoire sont plus graves chez le patient. Lors de l'exécution d'une tâche il est possible qu'il saute des étapes, ou qu'il les effectue en désordre. Il se peut également qu'il effectue d'autres actions n'ayant aucune relation avec la tâche entamée. L'étude faite par Baum et al.[6, 7] classifie ces erreurs en six catégories :

1. Erreurs d'**initiation** : Quand le patient n'arrive pas à amorcer une activité même si un thérapeute lui indique de la commencer. Par exemple, même s'il est conscient qu'il doit prendre ses médicaments, il n'arrive toujours pas à effectuer la première action de ce plan qui pourrait être *prendre un ver d'eau.*
2. Erreurs d'**organisation** : Quand le patient n'utilise pas les bons outils pour réaliser une activité. Le patient peut, par exemple, essayer d'utiliser un couteau au lieu d'un tire-bouchon pour ouvrir une bouteille de vin.
3. Erreurs de **réalisation** : Quand le patient oublie des étapes ou ajoute des actions qui n'ont rien à voir avec l'activité. En préparant son petit déjeuner, il arrive que le patient oublie de toaster le pain et le tartine directement.
4. Erreurs de **séquence** : Quand le patient réalise de façon désordonnée les différentes étapes de l'activité. Tartiner le pain puis le toaster est un des exemples de ces erreurs.
5. Erreurs de **jugement** : Quand le patient réalise l'activité d'une façon non sécuritaire. Ouvrir une bouteille de vin avec un couteau fait aussi partie des erreurs de jugement.
6. Erreurs de **complétion** : Quand le patient n'arrive pas à déterminer si l'activité en cours de réalisation est terminée. Le patient peut, par exemple, continuer à attendre devant la poêle alors que l'eau est déjà bouillante.

Dans la phase avancée, il n'est plus question de supervision mais le patient doit être complètement pris en charge. Il est incapable d'effectuer n'importe quelle tâche même avec assistance. Cela explique la dépendance totale mentionnée dans la rubrique *Autonomie globale* du Tableau 1.1.

Donc, d'après le Tableau 1.1, le patient a besoin d'assistance pendant quatre à cinq ans, soit la durée totale des deux premières phases. Il existe différentes formes d'assistance. Le patient peut rester chez lui et des membres de sa famille vont l'assister (aidants naturels). Ou bien la famille peut engager un professionnel pour veiller sur le patient (aidant professionnel). Sinon, Le patient peut intégrer une maison de retraite ou un établissement de soins de longue durée. Chacune de ces formes d'assistance a ses propres inconvénients, mais elles participent toutes d'une façon ou d'une autre à augmenter de façon spectaculaire les coûts globaux engendrés par la maladie. En plus elles habituent le patient à recevoir de l'aide d'une personne aidante et le rendent rapidement totalement dépendant d'elle [4].

L'assistance automatisée est donc proposée pour remédier à tous ces problèmes et aussi pour ses différents avantages : préserver l'intimité du malade et éviter la relation souvent complexe entre la personne aidante et le malade: Le grand défi de cette forme d'assistance est de pouvoir reconnaître l'activité entamée par le patient afin de lui fournir, au besoin, l'aide nécessaire au bon déroulement de l'activité. Différentes recherches essaient de répondre à cette problématique utilisant différentes approches : logiques [8,9], probabilistes [10, 16, 17] ou hybrides [11]. De récentes recherches se basent sur des techniques émergeantes, comme celles de la fouille de données temporelles, pour obtenir des résultats encore plus prometteurs.

## 1.4. La fouille de données temporelles

De nos jours, et dans tous les domaines, les bases de données chargées d'entreposer les informations sont gigantesques et de plus en plus inter-reliées. Nous ne parlons même plus de base de données mais d'entrepôt de données. Si les requêtes SQL sont parfaitement capables d'en retirer des informations, aussi compliquées que soient-elles, elles sont totalement inutiles pour en extraire des connaissances. L'exemple d'un entrepôt de données d'un supermarché illustre très bien ces propos : les requêtes SQL peuvent facilement répondre à des questions sur la quantité vendue d'un produit ou sur la recette d'un mois, mais elles sont incapables de construire une base de connaissance sur les habitudes de consommation des clients, par exemple de trouver les produits que les clients ont l'habitude d'acheter ensemble [12].

La fouille de données est donc apparue et a pour objectif d'extraire de la connaissance après l'analyse des données stockées dans l'entrepôt de données. Elle peut être définie plus précisément en étant un ensemble de méthodes et techniques automatiques ou semi-automatiques explorant les données en vue de détecter des règles, des associations, des tendances inconnues ou cachées, des structures particulières restituant l'essentielle de l'information utile pour la prise de décision [12]. Elle se divise en deux majeures catégories : La fouille de données descriptive qui ne fait que mettre en évidence des informations déjà existantes et qui considère toutes les données au même degré d'importance, et la fouille de données prédictive qui extrapole de nouvelles informations à partir de celles déjà présentes et qui considère une ou plusieurs données plus importantes que les autres, ce sont la cible de l'analyse.

Les excellents résultats obtenus par le biais de la fouille de données lui ont permis de s'attaquer à des domaines tellement diversifiés et tellement nombreux que nous ne pouvons tous

les citer : la sécurité, le marché boursier et le commerce électronique, la santé, etc. L'exemple du supermarché, cité ci-haut, nous montre une de ses applications dans le domaine du commerce et comment, si nous gardons des informations sur le temps des transactions; elle peut nous aider dans la prise des décisions mercatiques; en découvrant par exemple les produits qui se vendent ensemble. Ces données temporelles augmentent considérablement la complexité de la fouille de données classique car on doit tenir compte des différents types de données temporelles, des relations ou opérateurs temporels et de la granularité temporelle [13] etc. C'est pour cette raison qu'un domaine à part entière a vu le jour, la fouille de données temporelles, avec des techniques et des tâches bien définies [14] :

- La caractérisation et comparaison de données temporelles
- La classification temporelle
- Les règles d'associations temporelles
- La prédiction temporelle et l'analyse des tendances
- L'analyse des motifs temporels : une de ses utilisation est la découverte des motifs ou motif qui se répète dans le temps. Encore une fois l'analyse du panier de la ménagère est un bon exemple où on cherche dans les différentes transactions les motifs qui reviennent le plus souvent; ce qui se traduit par les produits qui se vendent ensemble.
- La segmentation temporelle : vue le très grand nombre de données au sein d'un entrepôt de données, il est plus simple de travailler avec un nombre réduit de groupes d'objets homogènes. La segmentation nous permet de créer ces groupes homogènes en utilisant différents algorithmes et en se basant sur la distance minimale entre les membres du même groupe. Elle est utilisée dans plusieurs domaines : la bio-informatique, l'analyse des images, l'apprentissage automatique etc.

L'habitat intelligent, la solution proposée pour automatiser l'assistance des patients de la maladie d'Alzheimer, est un bon champ d'application pour la fouille de données temporelles parce qu'il peut être considérée comme un grand entrepôt de données composé des différentes informations envoyées en continu par les senseurs. Donc c'était tout à fait naturel à ce que les techniques de la fouille de données et la fouille de données temporelles prennent la relève et remplacent les approches classiques utilisées jusqu'à présent dans ce domaine.

## 1.5. Les travaux existants sur la reconnaissance d'activités

Schmidt et all. [15] définissent la reconnaissance d'activités, connue aussi sous la reconnaissance de plans, par le fait de : « prendre en entrée une séquence d'actions exécutées par un acteur et d'inférer le but poursuivi par l'acteur et d'organiser la séquence d'actions en terme d'une structure de plan ». L'acteur suit donc un plan d'actions pour atteindre un but bien déterminé, alors que l'agent, qui possède normalement tous les plans d'actions ou ce que nous appelons aussi les plans d'activités, essaie de trouver le plan adéquat en se basant sur les actions observées. Différentes approches ont été utilisées pour répondre à ce problème.

### 1.5.1. Les approches classiques

En parcourant la vaste littérature sur le sujet, les approches classiques peuvent se classer sous deux majeures catégories :

Les approches logiques [8, 9]: visent à sélectionner, parmi les plans d'activités, par une série de déductions logiques, l'ensemble des activités possibles qui peuvent expliquer un ensemble d'actions observées. Le grand avantage de ces approches est leurs efficacités sur une très large

base de connaissance. Elles permettent d'ignorer rapidement la majorité des activités et de n'en garder qu'un petit ensemble. Les traitements et calculs se font alors sur un ensemble réduit, ce qui donne un meilleur temps de réponse. Par contre, elles peuvent facilement éliminer l'activité recherchée. Ce sont des approches qui supposent que l'entité observée agit de manière rationnelle et que toute action détectée est en cohérence avec l'objectif visé. Une supposition qui n'est pas réaliste dans notre cas. Une action détectée qui n'appartient pas au plan d'une activité élimine automatiquement cette dernière des activités possibles, alors que le patient peut être en train d'effectuer la même activité et l'action détectée n'est dû qu'à une erreur de type réalisation de sa part. Cette action peut aussi être expliquée par le commencement d'une deuxième activité avant l'achèvement de la première : un plan entrecroisé. Le problème d'équiprobabilité est un autre problème des approches logiques. Certes elles permettent de ne garder qu'un petit ensemble des activités possibles mais elles ne permettent pas de favoriser une parmi cet ensemble. Toutes les activités de l'ensemble réduit ont la même probabilité d'être celle recherchée.

Les approches probabilistes [10] sont basées sur des modèles markoviens [17] ou des réseaux bayésiens [16]. Pour utiliser ces approches, une probabilité est assignée manuellement à chaque activité, puis ces probabilités sont mises à jour en fonction de l'ensemble des nouvelles observations. Ces approches probabilistes règlent le problème des approches logiques. Elles permettent de résoudre le problème d'irrationalité de l'entité observée. Une action, qui n'a aucun rapport avec l'objectif visé, n'éliminerai plus l'activité la plus probable. Elle est ignorée si elle n'est contenue dans aucune activité de la base de connaissance. Sinon elle augmentera d'une façon non significative la probabilité d'une ou de plusieurs activités qui la contiennent. Elle ne gênera pas la prise de décision car la probabilité de ces activités restera faible vue qu'elles ne

contiennent pas les autres actions observées. Le grand inconvénient de ces approches est la lourdeur du calcul et du traitement. Pour chaque action détectée c'est l'ensemble de toutes les activités de la base de connaissance qui est mis à jour.

Également, il existe des approches basées sur des théories hybrides comme la logique probabiliste (nillson) ou la logique possibiliste (dubois et prade) [53]. L'idée générale consiste à diminuer l'ensemble des activités sur lequel les traitements vont s'effectuer à l'aide d'une approche logique. Ensuite appliquer sur cet ensemble réduit la méthode probabiliste. Des approches qui héritent tout de même des inconvénients des approches logiques.

### 1.5.2. Les approches récentes

La diversité des techniques de la fouille de données temporelles a permis de traiter la problématique de la reconnaissance d'activité de différentes manières. Le travail de Jakkula et al. [18] par exemple, se base sur la logique temporelle d'Allen [19] et trouve des relations temporelles entre les actions du patient pour pouvoir prédire une action prochaine ou juste détecter une erreur temporelle commise par le patient quand les contraintes temporelles dans une activité ne sont pas respectées. Si nous parlons d'action alors que dans leur travail ils parlent d'évènement, c'est juste pour mettre l'accent sur la durée de cette action qui peut être vue comme un intervalle temporelle avec un temps de début et un temps de fin bien déterminés. Concernant les relations temporelles d'Allen, elles sont du genre X avant Y, X termine Y etc. il en existe treize et elles seront développées au prochain Chapitre.

Leur travail commence par une définition mathématique de ces relations pour faciliter leurs découvertes durant le parcours de l'entrepôt de données. Par exemple X avant Y est défini par : Temps début (X) < Temps début (Y) et Temps fin (X) < Temps début (Y).

Une relation de ce type est très utile car elle peut servir à prédire l'action Y si l'action X est détectée. De plus, elle permet de détecter qu'une erreur a été commise si l'action Y est détectée sans que l'action X ne le soit.

Le problème de cette approche est qu'elle est appliquée directement sur l'entrepôt de données, ce qui fait que toutes les relations temporelles découvertes sont entre évènements et non pas entre activités. Comme un événement peut appartenir à plusieurs activités, nous aurons plusieurs relations pour le même évènement, et nous ne saurons laquelle appliquer pour la prédiction de la prochaine action.

Les types de senseurs utilisés dans l'habitat intelligent influencent beaucoup le choix de la technique de la fouille de données temporelles à utiliser. Dans le travail de Spriggs *et al.*[20] des caméras vidéo ont été utilisées pour observer le patient. Même si leur entrepôt de données est sous forme de séquence vidéo, le processus de la reconnaissance de l'activité reste le même. Après la détection d'une ou plusieurs actions, il faut essayer de trouver l'activité entamée pour prédire la prochaine action sauf qu'ici une action est elle aussi une séquence vidéo de plus petite taille.

Ce dernier travail est basé, comme le nôtre, sur la segmentation temporelle. En effet la séquence vidéo qui représente l'activité est considérée comme un grand intervalle temporel qui peut être segmenté en plusieurs petits intervalles représentant des actions. La segmentation est basée sur

les motifs qui reviennent plus fréquemment. Les actions ainsi découvertes sont classifiées pour créer des plans de mouvements qui seront parcourus au futur afin de trouver l'activité entamée.

Les résultats de cette approche, d'après ses auteurs, étaient difficiles à évaluer, mais nous pouvons imaginer la lourdeur du traitement des séquences vidéo et son impact sur des applications qui doivent donner des réponses en temps réel. De plus, l'utilisation des caméras vidéo fait l'objet de plusieurs critiques parce qu'elle ne préserve pas l'intimité du patient.

Nous avons aussi utilisé la segmentation temporelle mais pas de cette manière. Nous l'avons utilisé d'une façon plus proche à celle présentée dans le travail de Harvey *et al.*[21] sur l'analyse de l'historique du travail des développeurs. Au départ il traite le temps comme une séquence de points et le rôle de la segmentation est de combiner des points consécutifs pour créer des segments. Le but étant de travailler sur un nombre réduit de segments plutôt que sur un nombre beaucoup plus grands de points.

Le problème de cette technique est que les segments ne représentent pas entièrement les données d'entrées. Les segments sont créés en essayant de minimiser les erreurs. Des erreurs qui se traduisent par la perte de certaines données, chose que nous ne pouvons pas nous permettre dans notre cas.

## 1.6. Contribution

Notre travail s'inscrit dans cette nouvelle tendance à vouloir exploiter les techniques de la fouille de données temporelles pour proposer une solution à la problématique de la reconnaissance d'activité au sein d'un habitat intelligent afin d'offrir une assistance automatisée aux patients de la maladie d'Alzheimer. Quant à notre contribution, elle consiste à concevoir une

nouvelle approche, basée sur le temps, et à répondre aux différents problèmes rencontrés pour sa mise en pratique.

Comme nous l'avons déjà expliqué, le plus grand problème dans le processus de la reconnaissance d'activité réside dans le nombre très élevé d'hypothèses. Notre approche utilise la technique de la segmentation temporelle pour réduire, au moment de la recherche d'activité, ce nombre d'hypothèses. Plus précisément, la segmentation temporelle nous permet de créer pour chaque plan d'activité les périodes, ou les intervalles de temps, où elle débute d'habitude. De cette façon, nous pouvons diminuer le niveau de plausibilité de toute activité qui, d'habitude, n'est pas commencé au moment de la recherche. Autrement dit, si le moment de recherche n'est inclus dans aucun des intervalles temporels du plan d'activité, elle peut être ignorée dans un premier lieu.

Un plan d'activité, comme expliqué auparavant, définie une activité par la liste ordonnée des actions qui la composent. Comme notre segmentation temporelle s'applique sur ces plans, nous avons également proposé dans ce mémoire une méthode pour les créer. Une méthode qui utilise elle aussi une des techniques de la fouille de données temporelles, l'extraction des motifs séquentiels [22]. Nous avons choisi d'utiliser l'algorithme BIDE proposé par Wang *et al.*[23] dans le domaine de l'analyse du panier de la ménagère. Dans notre cas, BIDE analyse notre entrepôt de données, contenant les évènements envoyés par les capteurs, pour découvrir les motifs les plus fréquents qui vont constituer nos plans d'activités. Donc, un plan d'activités est constitué d'un ensemble de motifs séquentiels d'évènement capteur, lesquels corresponds à des actions.

Le but ultime de notre travail est de trouver l'activité entamée par le patient. Après avoir créé les plans d'activités et après la réduction du nombre d'hypothèses, nous avons conçu une méthode qui nous donne à chaque instant l'activité la plus probable. Une méthode qui se base sur les résultats obtenus après la segmentation temporelle et qui se met à jour après chaque détection d'une nouvelle action du patient.

Cette dernière phase de notre travail ne permet pas juste de trouver l'activité entamée; mais, elle peut également prédire l'activité qui doit être commencée. En effet, elle peut désigner l'activité la plus probable avant même la détection d'aucune action du patient. Dans ce mémoire, on explique comment cette faculté offre une réponse aux erreurs d'initiation susceptibles d'être commises par notre acteur atteint de la maladie d'Alzheimer. Des erreurs qui, à notre connaissance, n'ont jamais été traitées auparavant.

Malgré que les tests de notre approche, avec des données enregistrées après l'observation d'un réel patient de la maladie d'Alzheimer pendant vingt-huit jours au sein d'un autre habitat intelligent, celui de Kasteren *et al.*[10], ont été plus que satisfaisante, nous estimons que plusieurs améliorations s'imposent pour rendre cette approche encore plus efficace. En fin, nous pensons que ce travail présente la reconnaissance d'activité et l'utilisation des données temporelles sous un nouvel angle et peut servir de base pour des prochains travaux. Il est à noter aussi qu'il très facile de lui intégrer des travaux existants, celui de Jakkula *et al.*[18] par exemple, pour un rendement meilleur. C'est dans cet objectif que notre prochain travail s'orientera au sein du laboratoire d'intelligence ambiante pour la reconnaissance d'activités (LIARA) à l'université du Québec à Chicoutimi.

## 1.7. Objectifs et méthodologie de la recherché

Ce travail de recherche a été effectué en trois majeures étapes :

La première étape avait pour premier objectif de comprendre le comportement des patients de la maladie d'Alzheimer afin de pouvoir anticiper les différentes erreurs qu'ils peuvent commettre et leur offrir ainsi une assistance plus adaptée à leur situation. C'est pour cette raison que nous avons commencé ce travail par une étude détaillée de cette maladie et des différentes erreurs qu'elle peut provoquer [4, 6, 7].

Comme nous sommes parties de l'idée d'utiliser la fouille de données temporelles pour résoudre les problèmes reliés à la reconnaissance d'activité, une autre étude sur les techniques de la fouille de données temporelles [12, 13, 14] s'imposait et a fait l'objet de la deuxième étape. Une étape où la nature des données temporelles et les opérations de la fouille de données temporelles y sont bien expliquées.

Quant à la troisième étape, elle définit plus précisément la reconnaissance d'activité et réalisé un état d'art sur les travaux déjà existants qui la traitent. L'objectif de ce travail étant de nous aider à conceptualiser notre approche en profitant de leurs avantages et en corrigeant ou évitant leurs faiblesses. Ces différents travaux seront détaillés au Chapitre 3, surtout ceux qui utilisent, comme nous, la fouille de données temporelles.

La quatrième étape avait pour objectif de créer une toute nouvelle approche. Celle-ci est basée sur le temps; elle est composée de trois parties. La première a été consacrée à la création des plans d'activités en utilisant la technique de l'extraction des motifs temporelles. La deuxième s'appuie sur la technique de la segmentation temporelle pour réduire au maximum le nombre

d'hypothèses afin de rendre faisable la recherche de l'activité en temps réel. La troisième partie présente un système de reconnaissance de l'activité capable de désigner l'activité la plus probable à chaque moment.

La dernière étape concerne la validation de notre approche afin de vérifier son opérationnalité. Les tests ont été réalisés sur des données proposées par van Kasteren *et al.*[10]. Des données réelles, enregistrées après l'observation d'un patient de la maladie d'Alzheimer au sein d'un habitat intelligent pour une période de vingt-huit jours. Comme notre approche se compose de trois parties, les résultats des tests sont évalués après chaque partie, de cette façon les points forts et faibles de cette approche sont plus pointus. Les améliorations et travaux à entrevoir sont aussi plus précis.

Notons que ce travail a été présenté lors du 79$^{ème}$ congrès de l'ACFAS à Sherbrooke. Il a aussi été publié sous la forme d'un article de six pages à la conférence internationale SMART 2012 à Stuttgart en Allemagne. Cette reconnaissance vient appuyer l'originalité de notre approche et les bons résultats obtenus.

## 1.8. Organisation de ce mémoire

Ce mémoire est organisé en trois chapitres qui viennent détailler chacune des étapes décrites dans la méthodologie de recherche.

Le Chapitre 3 définira plus profondément la reconnaissance de l'activité. Ensuite nous y présenterons les travaux déjà existants portant sur cette problématique de recherche. Les travaux utilisant la fouille de données temporelles seront vus plus en profondeur parce que nous utilisons ces mêmes techniques dans notre approche.

Dans le Chapitre 4 nous allons expliquer en détail notre approche proposée pour répondre à la problématique de la reconnaissance d'activité. Nous commencerons par l'explication de l'algorithme BIDE proposé par Wang et *al.*[23] que nous utilisons pour la création des plans d'activités. Ensuite, nous énoncerons en détail la technique de la segmentation temporelle que nous avons utilisée dans le but de réduire le nombre d'hypothèses. En fin, nous terminerons par l'explication du système de reconnaissance de l'activité qui se base sur les résultats de la segmentation temporelle pour trouver l'activité la plus probable. Au cours de ce même Chapitre, les tests et validation de notre approche sont bien décrits. Comme elle se compose de trois parties, les résultats des tests seront évalués après chaque partie.

Nous terminons ce travail par une conclusion générale où nous montrons les avantages et les limites de notre approche. Nous y discutons également des améliorations à entrevoir et des travaux existants qui peuvent s'intégrer à notre approche pour un meilleur rendement.

# Chapitre 2

# La fouille de données temporelles

## 2.1. Introduction

Dans le domaine de la reconnaissance d'activités, les nouvelles recherches s'orientent vers l'utilisation de la fouille de données, surtout après la définition de Patterson *et al.*[25] de la problématique de la reconnaissance d'activité comme «la tâche d'inférer l'activité exécutée par l'entité observée à partir de données fournies par des capteurs de bas niveau ». C'est une définition qui ressemble beaucoup aux précédentes tout en ajoutant une information cruciale qui précise que la reconnaissance est basée sur les données envoyées par les capteurs et non pas sur des actions. Cette définition est parfaitement adaptée à l'habitat intelligent où nous plaçons plusieurs capteurs pour détecter les changements dans l'environnement qui peuvent se traduire par des actions de la personne observée. L'analyse de l'énorme base de données, qui enregistre la quantité massive des données envoyées par les capteurs, cause un vrai problème aux méthodes classiques et favorise l'utilisation de la fouille de données qui a été créée justement pour ce

propos. De plus, la fouille de données est composée d'un ensemble de techniques variées qui nous permettent d'aborder cette problématique selon différents angles, ce qui explique la diversité des travaux qui exploitent la fouille de données dans la reconnaissance d'activité.

Les résultats obtenus après l'utilisation de la fouille de données, dans différents domaines, sont très satisfaisantes ce qui nous encourage encore plus à l'utiliser. Parmi ces domaines nous pouvons citer: la sécurité, le marché boursier et le commerce électronique, la santé, etc.

Par contre, la fouille de donnée pèche à traiter les données temporelles. Des données, primordiales dans notre cas d'étude, qui peuvent révéler des dépendances et des relations pertinentes à la reconnaissance d'activité ou à la détection d'erreurs commises pendant l'exécution de l'activité. Même si nous avons essayé d'adapter ces techniques à gérer ce genre de données, mais leurs types de représentation, les relations temporelles qui existent entre elles et la granularité temporelle ont fait en sorte qu'un domaine à part entière naisse [13], la fouille de données temporelles.

Au cours de ce Chapitre, nous allons commencer par définir la fouille de données, puis les types de données temporelles, pour finir avec la fouille de données temporelles et ses différentes techniques.

## 2.2. La fouille de données

Avant 1960 et à l'aube de l'ère informatique, l'analyse des données était une tâche attribuée à une personne ayant des connaissances d'expert et une formation dans les statistiques. Son travail consistait à parcourir et analyser les données brutes et trouver des motifs, faire des extrapolations, et à localiser d'intéressantes informations qui va ensuite véhiculer via des

rapports écrits, graphiques et diagrammes. Mais aujourd'hui, la tâche est trop compliquée pour un seul expert [25]. L'information est répartie sur de multiples plateformes et stockées dans une grande variété de formats. Certaines sont bien structurées et d'autres ne le sont pas. Les données sont souvent incomplètes. Parfois, elles sont continues et d'autres fois discrètes. Par contre, la quantité de données à analyser est toujours énorme. En fait, d'après Dunham [26] les données doublent chaque année et pourtant la quantité d'informations utiles dont nous disposons est à la baisse. La fouille de données est donc apparue pour essayer d'automatiser cette tâche très complexe d'analyse et d'extraction de connaissances. Oui, nous disons bien, essayer d'automatiser, parce que jusqu'à aujourd'hui, il existe des problèmes pour lesquels nous n'avons pas pu faire mieux que de proposer des techniques semi-automatiques de la fouille de données.

### 2.2.1. Les origines de la fouille de données

Les premières briques qui ont été posées pour la création de la fouille de données d'aujourd'hui remontent aux années 50 quand les travaux des mathématiciens, des logiciens et des informaticiens ont été combinés pour créer l'intelligence artificielle et l'apprentissage automatique [27].

Le domaine de l'intelligence artificielle, par la suite, avec celui des statistiques ont connu le développement de nouveaux intéressants algorithmes, utilisés de nos jours dans la fouille de données, comme l'analyse de régression, les réseaux de neurones et les modèles linéaires de classification, etc [26]. Le terme fouille de données «Data Mining» est apparu dans cette même période, les années 60, pour désigner la pratique de fouiller les données pour trouver des motifs qui n'avaient aucune signification statistique [25]. Le domaine de la récupération d'informations

«information retrieval» a fait sa contribution lui aussi dans cette même période par les techniques de segmentation et de mesures de similarité [26].

L'année 1971 est une année très importante dans l'histoire de la fouille de données, parce que c'est durant cette année que Gerard Salton a publié ses travaux innovateurs sur le système «SMART Information Retrieval» et a présenté une nouvelle approche de recherche d'information qui a utilisé un modèle basé sur l'algèbre d'espace vectoriel. Ce modèle se révélera être un ingrédient clé dans la boîte à outils de la fouille de données [26].

Les années d'après ont connu le développement d'autres importants algorithmes comme les algorithmes génétiques, le k-means [28] pour la segmentation et les algorithmes des arbres de décisions, etc. Les années 90 ont vu, pour la première fois, l'utilisation du terme entrepôt de données «Data Warehouse» afin de décrire une grande base de données (composé d'un schéma unique), créé à partir de la consolidation des données opérationnelles et transactionnelles de la base de données. C'est durant cette même période que la fouille de données a connu son grand lancement en cessant d'être juste une technologie intéressante et devient une partie intégrante des pratiques standards du monde des affaires [29]. En effet, les entreprises ont commencé à utiliser la fouille de données pour les aider à gérer toutes les phases du cycle de vie des clients, à augmenter les recettes provenant des clients existants, à conserver les bons clients et même à la recherche de nouveaux clients. Plusieurs facteurs ont participé à ce que les entreprises adoptent la fouille de données, la chute des coûts relatifs au stockage des données sur les disques informatiques, l'augmentation de la puissance du traitement des informations et surtout, les avantages de l'exploration des données sont devenus plus apparents.

De nos jours, la fouille de données est utilisée dans des domaines très variés, la télécommunication, la sécurité, le marketing, la finance, le marché boursier et le commerce

électronique, la santé, etc. Même les gouvernements n'hésitent pas à l'utiliser dans différents projets. En 2004, dans le rapport des activités fédérales sur la fouille de données, le bureau de comptabilité générale des États-Unis [30] a signalé qu'il y a 199 opérations de fouille de données en cours ou prévues dans les divers organismes fédéraux, et cette liste n'inclue pas les activités secrètes de la fouille de données comme MATRIX ou le système d'écoute de la NSA [31].

### 2.2.2. Définition de la fouille de données

La définition la plus complète de la fouille de données, à notre avis, est celle présentée par Benoît [12]: La fouille de données est un processus, à plusieurs étapes, d'extraction de connaissances, préalablement imprévus, de grandes bases de données, et d'application des résultats pour la prise de décision. Les outils de la fouille de données détectent des motifs à partir des données et déduisent des associations et des règles. Les informations extraites peuvent ensuite être appliquées à la prédiction ou à la création de modèles de classification, en identifiant les relations au sein des enregistrements de données ou entre des bases de données. Ces motifs et règles peuvent alors guider à la prise de décision et à la prévision des effets de ces décisions.

Concrètement, la fouille de données est comme tout autre programme informatique. Elle prend des entrées, normalement de grandes tailles. Elle les analyse et les traite pour en extraire, non pas des données déjà existantes, mais une connaissance. En fin, elle formule cette connaissance sous un format compréhensible et la présente comme sortie à l'utilisateur.

Les entrées peuvent être d'une grande variété de types de données, les bases de données, textes, données spatiales, des données temporelles, des images et autres données complexes [32]. Nous sommes allés même jusqu'à créer des domaines à part entière pour certains types d'entrées, pour les textes nous parlons de la fouille de textes «text mining», pour le Web nous parlons de la

fouille du Web «Web mining» et pour les données temporelles nous parlons de la fouille de données temporelles « temporal data mining».

Il est très important de préciser que même si la fouille de données travaille sur des bases de données, ou plutôt entrepôt de données, par exemple celui montré au Tableau 2.1 tiré du livre « Data Mining Practical Machine Learning Tools and Techniques » [33], elle est très différente des requêtes, qui interrogent, elles aussi, les bases de données. Les requêtes répondent aux questions du genre : "Combien de fois la pluie est tombé alors que la température était élevée?" alors que la fouille de données répond aux questions du genre : " est ce qu'il pleuvra si la température est élevée et l'humidité est normale ?".

| Outlook | Temperature | Humidity | Windy | Play |
|---|---|---|---|---|
| sunny | hot | high | false | no |
| sunny | hot | high | true | no |
| overcast | hot | high | false | yes |
| rainy | mild | high | false | yes |
| rainy | cool | normal | false | yes |
| rainy | cool | normal | true | no |
| overcast | cool | normal | true | yes |
| sunny | mild | high | false | no |
| sunny | cool | normal | false | yes |
| rainy | mild | normal | false | yes |
| sunny | mild | normal | true | yes |
| overcast | mild | high | true | yes |
| overcast | hot | normal | false | yes |
| rainy | mild | high | true | no |

**Tableau 0.1: Un exemple d'entrepôt de données par Witten [33]**

Les sorties aussi peuvent être de différentes formes. Elles sont la représentation finale de la connaissance ou du savoir extrait des données. En effet, une fois les motifs ont été identifiés, ils doivent être transmis à l'utilisateur final d'une manière qui permet à l'utilisateur d'agir sur eux et de fournir une rétroaction au système [34].

En se basant sur l'exemple d'entrepôt de données présenté au Tableau 2.1, et selon l'algorithme de la fouille de données appliqué dessus, nous pouvons citer quelques exemples des sorties possibles toujours d'après le livre cité ci-haut [33]:

✓ Arbre de décision

C'est une représentation, schématisée à la Figure 2.1, que nous pouvons obtenir en appliquant un algorithme de diviser pour régner "divide-and-conquer" [35]. Pour trouver un résultat, il suffit de suive les branches de l'arbre sachant qu'une branche, celle avec un "y", est à suivre quand la condition dans le nœud est vérifiée, et l'autre branche quand elle ne l'est pas, jusqu'à ce que nous tombons sur une feuille. Son avantage est qu'elle est très lisible pour un humain.

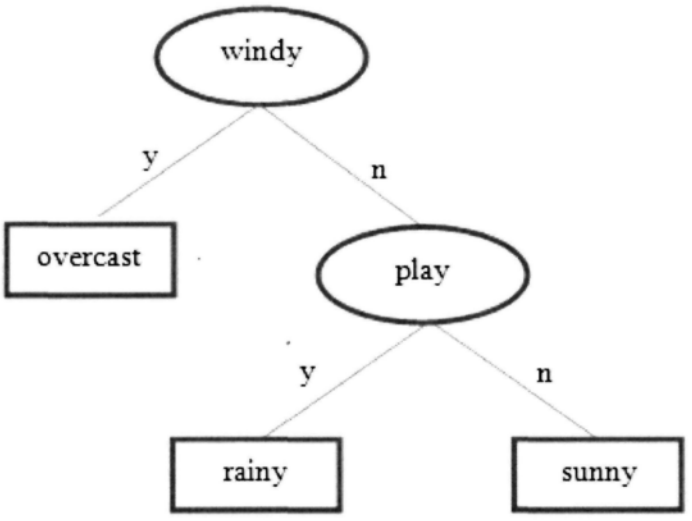



**Figure 0.1: Exemple d'arbre de décision**

✓ Règles de classification

Facile à obtenir à partir d'un arbre de décision. Elle a l'avantage d'être plus compacte. De plus, chaque règle est indépendante de l'autre ce qui donne une facilité d'ajout d'autres règles.

If windy= false and humidity=high then play = yes

If temperature=hot1 and outlook=sunny then play = no

Otherwise class = b

✓ Règles d'association

Ressemble beaucoup aux règles de classification, mais ils peuvent prédire n'importe quel attribut, des combinaisons d'attributs ou à chaque fois différentes valeurs du même attribut.

If temperature = cool then humidity = normal

If windy = false and play = no then outlook = sunny and humidity = high

If windy = false and play = no then outlook = sunny

If windy = false and play = no then humidity = high

✓ Règles avec exceptions

Ce sont des règles comme les précédentes, sauf qu'elles permettent des exceptions.

If windy = false and play = no then outlook = sunny

EXCEPT If temperature = cool then outlook = rainy

Elles permettent même des exceptions des exceptions.

If windy = false then outlook = sunny

EXCEPT If temperature = cool then outlook = rainy

EXCEPT If play = no then outlook = overcast

C'est une idéale représentation pour les règles complexe, mais reste très difficile à lire et à comprendre.

✓ Règles impliquant des relations

Comme son nom l'indique, nous cherchons des relations entre les attributs et non pas entre les valeurs. L'exemple de détecter si des figures sont debout ou couché illustre très bien ce cas :

| Largeur | Hauteur | Classe |
|---|---|---|
| 5 | 2 | couché |
| 3 | 12 | debout |
| 2 | 11 | debout |
| 6 | 4 | couché |

Au lieu d'avoir comme les règles précédentes : If Hauteur > 7 then debout

Nous allons trouver des règles encore plus intéressantes qui impliquent des relations entre les attributs, de la sorte : If Hauteur > Largeur then debout

✓ Les arbres pour les prédictions numériques

Ce sont des arbres qui ressemblent beaucoup aux arbres de décision, sauf que la condition au sein d'un nœud n'est plus booléenne mais numérique, du genre : Hauteur > 7. Ces arbres sont appelés aussi "regression trees" parce qu'elles contiennent des équations de régression.

✓ Segments

Quand nous appliquons un algorithme de segmentation, le K-means par exemple [28], le résultat obtenu est un ensemble de clusters ou segments. Tous les membres du même segment sont homogènes entre eux et différents des membres des autres segments.

Quant au processus d'analyse, de traitement et d'extraction de connaissances, comme cité dans la définition en haut, il se compose de plusieurs étapes. Des étapes que nous allons détaillées dans la prochaine section.

### 2.2.3. Les étapes du processus de la fouille de données

Avant de commencer le processus, l'opération de la fouille de données doit être bien comprise: L'analyste doit connaitre le problème à résoudre et les questions auxquelles nous devons répondre. Il peut travailler avec un spécialiste du domaine pour affiner le problème à résoudre. Benoît [12] considère ce travail comme la première étape du processus de la fouille de données, et qu'au cours de la seconde, l'analyste doit voir avec l'utilisateur final quelles données doivent être analysées afin de répondre à leurs questions et vérifier la disponibilité de ces données. Les prochaines étapes de ce processus peuvent être classées comme suit:

✓ La sélection et le nettoyage des données

Dans cette étape, nous devons faire face à deux éventuels problèmes [12, 25] :

1. Des données manquantes: Parfois l'opérateur de saisi oublie de remplir des champs non obligatoires ou il ne possède pas l'information au moment de la saisi, etc. Au cours de cette étape, nous devons trouver une solution qui peut consister, par exemple, à supprimer les enregistrements incomplets, les remplir manuellement, entrer une constante pour chaque valeur manquante ou l'estimer, etc.
2. Des données bruitées: Les données sont complètes mais erronées à cause d'un bruit. Nous pensons, par exemple, à un capteur du type étiquette RFID qui envoie des informations qui arrivent erronées à cause d'interférences. Dans ce cas aussi nous devons trouver une solution qui peut ressembler à celles décrites plus haut.

✓ La transformation des données

Après avoir géré les données manquantes, nous devons préparer les données en les transformant dans un format plus approprié à la fouille de données [36]. Cette procédure de transformation peut inclure [34] :

1. Le lissage de données «smoothing» : Par exemple, utiliser les moyennes pour remplacer les données erronées.
2. L'agrégation : Par exemple, afficher les données mensuellement plutôt que quotidiennement.
3. La généralisation : Par exemple, au lieu d'utiliser l'âge exact des personnes, nous pouvons les définir comme, petits, jeunes ou vieux.
4. La normalisation : Changer des valeurs pour qu'elles soient toujours dans une fourchette fixe. Par exemple, recalculer les valeurs d'un champ pour qu'il corresponde à un pourcentage entre 0 et 100.
5. La construction d'attributs : Ajouter de nouveaux attributs à l'ensemble des données.

✓ La réduction des données

Vu que les entrées sont généralement trop grandes, les données ont besoin d'être réduites pour rendre le processus d'analyse gérable, rentable et plus rapide. Plusieurs techniques peuvent être utilisées pour réduire les données dont [34] :

1. L'agrégation : C'est la même technique utilisée, et expliquée, dans l'étape de la transformation (afficher les données mensuellement plutôt que quotidiennement).
2. La réduction de dimension : Les attributs non pertinents ou redondants sont supprimés.
3. La compression des données : Les données sont codées afin de réduire leur taille.

4. La réduction de numérosité : Des modèles ou des échantillons sont utilisés au lieu des données réelles.

✓ La découverte de motifs

Dans cette étape, les données sont itérativement parcourues par les algorithmes de la fouille de données, dans un effort pour trouver les relations ou les motifs intéressants et utiles [12]. Certains motifs sont plus intéressants que d'autres, et cet "intérêt" est l'une des mesures utilisées pour déterminer l'efficacité de l'algorithme [25, 37, 34].

✓ L'interprétation et la visualisation des résultats

Durant cette étape, nous tenons à ce que les résultats soient bien présentés et facilement compris par l'utilisateur final. Les outils de l'infographie et de la conception graphique peuvent être utilisés pour une meilleure visualisation des résultats.

Enfin, Les résultats peuvent être utilisés ou servir comme des entrées pour un autre algorithme de fouille de données.

Après avoir détaillé les étapes du processus de la fouille de données, nous soulignerons le rôle important de l'étape de transformation qui prépare les données pour une meilleure utilisation. Parmi les données qui nécessitent souvent des transformations plus particulières, nous pouvons citer les données temporelles. Des données de différents types que nous allons aborder dans la prochaine section; et, qui possèdent différentes représentations; ce qui complique leur analyse par les algorithmes de la fouille de données.

## 2.3. Les types de données temporelles

Conceptuellement, les données temporelles peuvent être classifiées en deux types différents: les séquences et les séries temporelles [38].

### 2.3.1. Les séquences

Une séquence peut être définie comme un ensemble ordonné d'évènements souvent représentée par une série de symboles nominaux [14]. Dans le contexte de l'habitat intelligent qui nous concerne, la matinée de la personne observée peut être vue comme une séquence d'activité. Une séquence qui peut ressembler à celle schématisée dans la Figure 2.2.

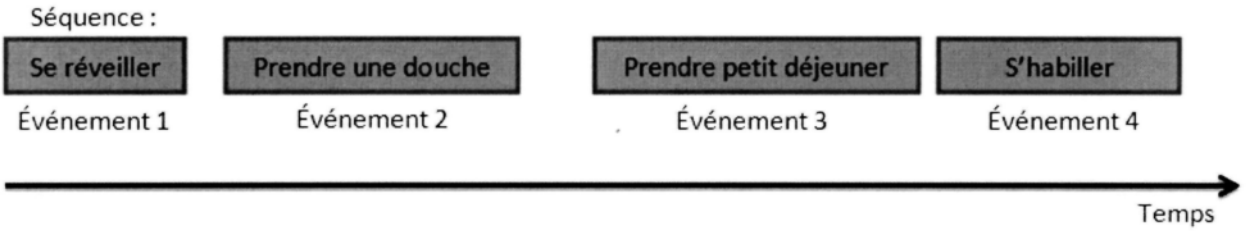



**Figure 0.2: Une séquence ordonnée**

La seule information temporelle qu'on peut ressortir de ce type de séquence, les séquences ordonnées, est que les évènements sont ordonnés dans le temps et se produisent séquentiellement. Par exemple, l'évènement 2 se produit après l'évènement 1, etc. Pour certaines applications, nous aurons besoin d'informations temporelles encore plus précises. Pour cette raison, nous pouvons décider d'ajouter le temps réel où l'évènement s'est produit. De cette façon, à la place d'une séquence ordonnée, nous allons avoir une séquence horodatée qui peut être schématisée dans la figure 2.3.

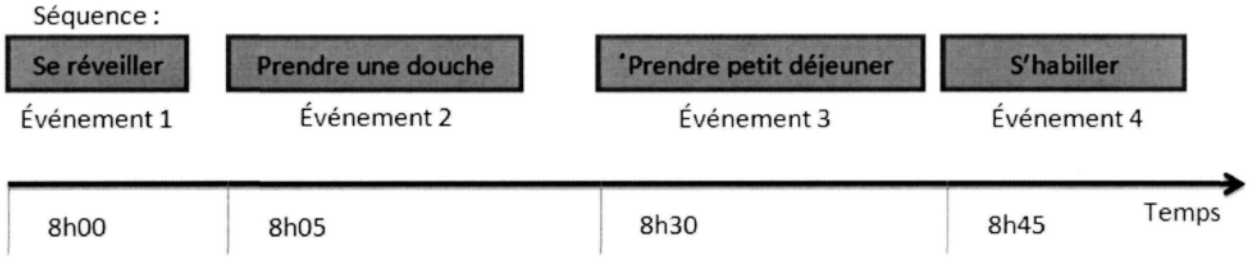



**Figure 0.3: Une séquence horodatée**

## 2.3.2. Les séries temporelles

Une série temporelle est une séquence de valeurs continues d'éléments [14]. Pour comprendre ce type de données temporelles, il suffit de penser à un capteur qui envoie en continu des mesures de températures, de pression ou la position d'une tasse, etc. Par conséquent, ce que nous enregistrons dans la base de données, ce sont des mesures prises par le capteur ainsi que le temps réel où les mesures étaient faites. La Figure 2.4 montre un cas typique d'une série temporelle.

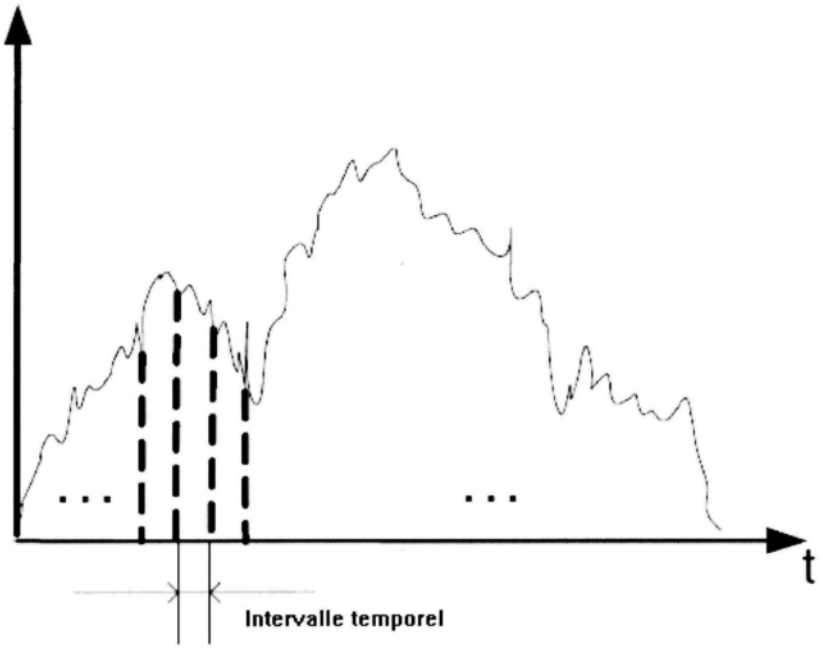



**Figure 0.4: Une série temporelle**

C'est ce type de données, les séries temporelles, qui agrandi d'une façon spectaculaire les bases de données et rend les techniques d'extraction de connaissance impraticables. Pour remédier à ce problème, la fouille de données temporelles effectue une transformation pour mieux présenter ces types de données. Une étape qui sera expliquée dans la prochaine section.

## 2.4. La fouille de données temporelles

La fouille de données temporelles est un domaine relativement nouveau. Il est devenu plus populaire dans la dernière décennie en raison de la capacité accrue des systèmes informatiques modernes qui sont devenus capable de stocker et de traiter de grandes quantités de données de plus en plus complexes. Il n'est pas inhabituel de voir des équipements modernes générer des mégaoctets de données temporelles pendant leur surveillance constante des différents paramètres. Même une simple puce d'un capteur peut produire une quantité énorme d'informations temporelles, ce qui rend son analyse très difficile sans l'utilisation des techniques de la fouille de données temporelle.

La fouille de données temporelles peut donc être définie exactement comme la fouille de donnée avec la particularité de bien traiter les données temporelles. Les étapes qui la composent sont aussi les mêmes sauf l'étape de transformation qui est un peu plus poussée pour pouvoir gérer la complication des données temporelles. Cette étape sera décrite dans la prochaine section, alors que la section d'après sera réservée à la présentation des tâches de la fouille de données temporelles.

### 2.4.1. L'étape de transformation dans le processus de FDT

L'étape de transformation dans le processus de la fouille de données temporelles a pour objectif de réduire la complexité des données originales et de les présenter sous un format plus adéquat à l'analyse. En tout, il existe six techniques de transformation dont quatre sont spécifiques aux séries temporelles :

- Pas de transformation
- Série temporelle divisée
- Point dans un espace de dimension N
- Langage

Tandis que les deux autres s'alignent pour les deux types [38] :

- Séquence
- Modèle

#### 2.4.1.1.Pas de transformation

C'est une technique rarement utilisée à cause d'un potentiel problème de compatibilité. Il est clair que comparer, par exemple deux signaux chacun avec sa propre amplitude et fréquence donnera généralement de fausses résultats. Néanmoins, il existe des techniques qui ont été conçues pour travailler directement sur les données originales [39, 40]. Certaines divisent le signal en différents segments puis effectuent une mise à l'échelle, etc.

### 2.4.1.2.Série temporelle divisée «Jagged Time-Series»

Cette technique essaie de trouver une fonction linéaire par morceaux pour approximer les données d'origines. La figure 2.5 montre graphiquement cette approximation. Le grand défi de cette technique est d'élaborer un algorithme qui effectue une bonne approximation. Keogh *et al.*[41] en 1997 ont proposé une solution basée sur la segmentation mais qui avait l'inconvénient de devoir spécifier manuellement et à l'avance le nombre de segments. L'année 2000 a connu la présentation de leur nouvelle approche, «Piecewise Aggregate Approximation», qui a subi plusieurs améliorations par la suite [61].

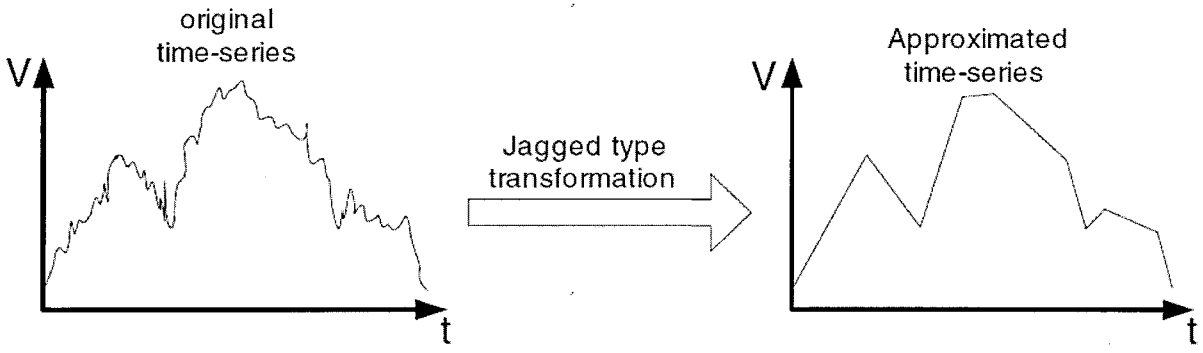



**Figure 0.5: Un exemple typique d'une transformation série temporelle divisée**

### 2.4.1.3.Point dans un espace de dimension N

Cette technique essaie de transformer le domaine d'appartenance des données d'un domaine temporel à un domaine de fréquence. Pour effectuer cette transformation, ces techniques se basent généralement sur les transformations discrètes de Fourier (DFT) [42] ou celles de Wavelet les transformations en ondelettes discrètes (DWT) [43]. Comme le montre la Figure 2.6, une donnée sera donc représentée par trois coefficients de fréquence. Une représentation qui

permettra la réduction de la dimension de l'espace et le calcul métrique d'une distance entre deux données par exemple.

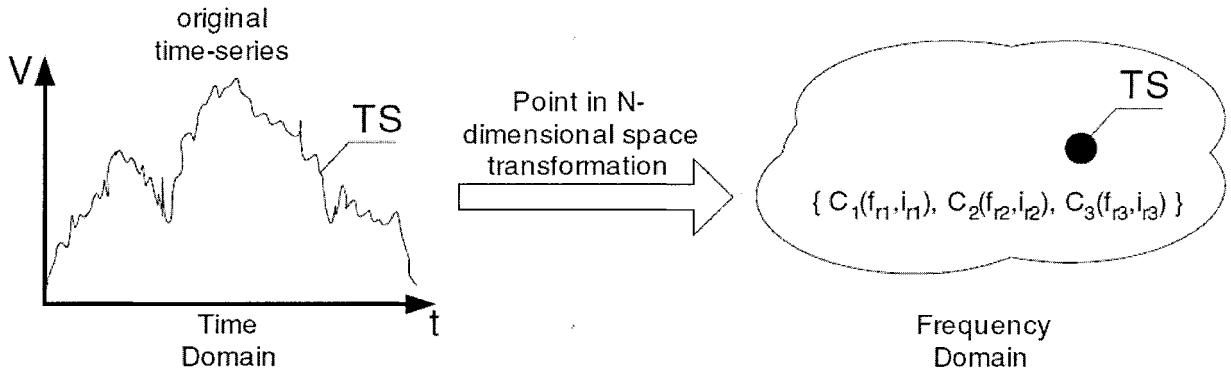



**Figure 0.6: Un exemple d'une transformation Point dans un espace de dimension N**

## 2.4.1.4.Langage

La Figure 2.7, illustre bien cette technique où nous essayons de remplacer une série temporelle par un ensemble de définitions qui décrivent un changement dans sa forme. L'utilisation de cette transformation permet une comparaison facile du comportement général des séries temporelles.

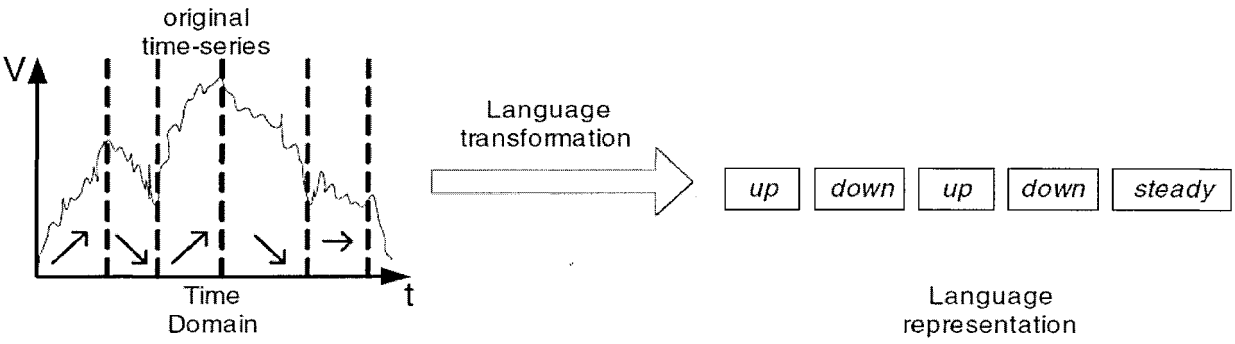



**Figure 0.7: Un exemple typique d'une transformation Langage**

### 2.4.1.5.Séquence

Comme le montre bien la Figure 2.8, cette technique transforme les séries temporelles en séquences ordonnées ou horodatées. C'est ce type de transformation que nous estimons adéquat avant d'appliquer notre algorithme de reconnaissance d'activité.

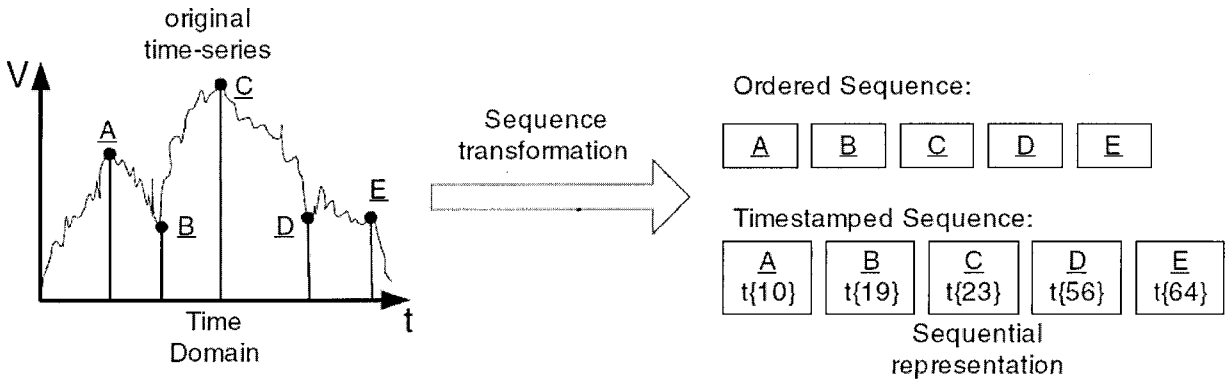



**Figure 0.8: Un exemple d'une transformation Séquence**

### 2.4.1.6.Modèle

La Figure 2.9 schématise bien cette technique qui essaie de transformer une collection de séries temporelles en une sorte de modèle. Pour expliquer brièvement cette technique, on peut dire qu'elle essaie d'extraire les éléments importants des données temporelles et établie des relations entre eux. Son plus grand avantage est que les relations cachées et les interactions peuvent facilement être découvertes.

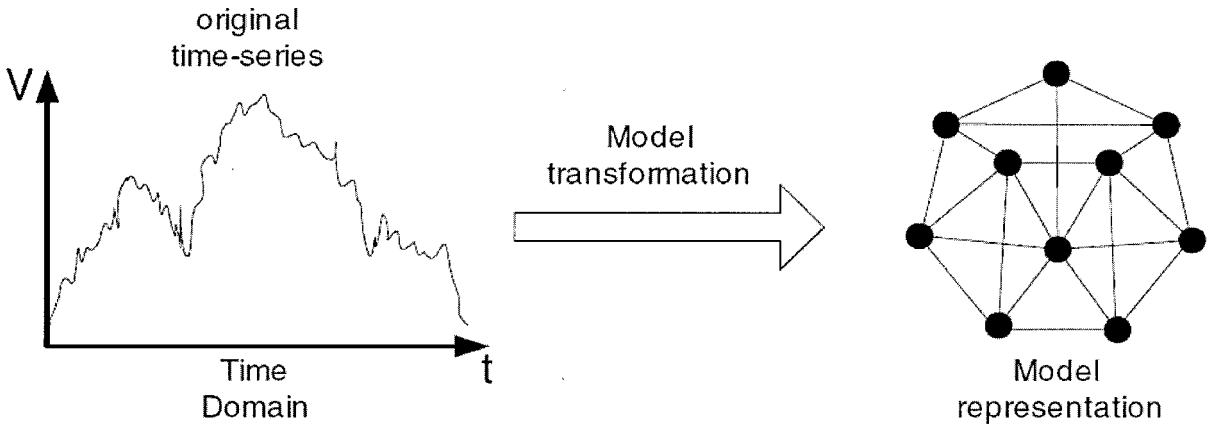



**Figure 0.9: Un exemple d'une transformation Modèle**

## 2.5. Les opérations de la fouille de données temporelles

La fouille de données temporelles englobe différents algorithmes et techniques qui se spécialisent dans l'analyse des données et l'extraction de la connaissance. Ces techniques peuvent être catégorisées en trois groupes : La classification, la segmentation et l'association.

### 2.5.1. La classification

La classification reste l'opération la plus fréquente de la fouille de données temporelles. L'objectif de la classification est de classer une instance inconnue dans une des classes déjà définies. Cette opération s'effectue en deux étapes [33] illustrées dans la Figure 2.10, une première étape d'apprentissage, puis l'étape de classification.

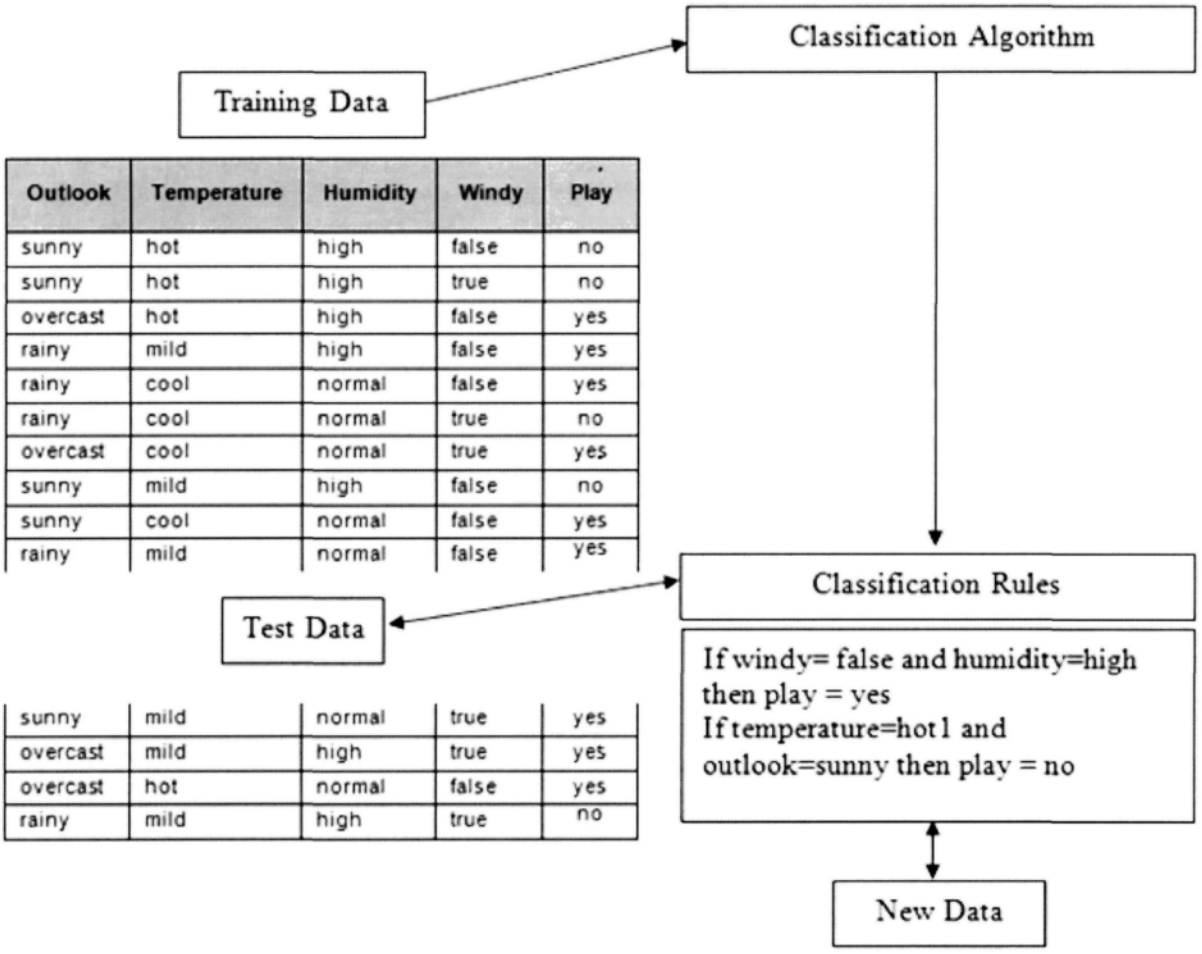



**Figure 0.10: Les deux étapes du processus de classification**

- L'étape d'apprentissage : Dans cette étape, l'algorithme de classification essaie de créer un classificateur en analysant et en apprenant d'un ensemble de données d'apprentissage «Training Data» crées des tuples de l'entrepôt de données et de leur classe associée. Un tuple, X, est représenté par un vecteur d'attributs de dimension n, X = (x1,x2,…xn), correspondant aux n valeurs des attributs de l'entrepôt de données A1, A2, …An. Chaque tuple, X, est supposé appartenir à une classe prédéfinie et déterminée par un autre attribut de l'entrepôt de données. Si nous revenons à la Figure 2.10, l'attribut qui détermine la

classe est « Play», et il existe deux classes, «yes» et «no». Les tuples sont de dimension 4. Par exemple, le tuple, X = (sunny, hot, high, false) appartient à la classe «no».

Le classificateur, ainsi créé, peut être représenté par des règles de classification, des arbres de décision ou des formules mathématiques de la forme y = f(X) où y est la classe trouvée après l'application de la fonction f sur le tuple X.

- L'étape de classification : Avant d'utiliser ce classificateur pour trouver la classe de nouvelles données, il faut avoir une estimation de sa précision. Pour cette raison, nous l'appliquons d'abord sur des données dites de tests. Il est inutile d'utiliser les mêmes données d'apprentissage pour les tests parce que les résultats seront optimaux. Plusieurs techniques existent pour diviser l'entrepôt de données en données d'entrainement et de tests. La Figure 2.10, montre la technique deux tiers un tiers, où deux tiers sont réservés à l'entrainement et un tiers aux tests. Nous pouvons aussi utiliser une technique d'échantillonnage où les enregistrements –données- sont tirées aléatoirement, technique de « boostrap » [44], etc.

  Si les tests sont satisfaisants, nous pouvons utiliser le classificateur pour les nouvelles données, sinon on peut changer d'algorithme de classification, le raffiner ou changer les données d'entrainement.

Le type de classification expliqué ci-haut est dit supervisé parce que nous connaissons les classes à l'avance. L'autre type est dit non-supervisé et les classes ne sont pas connues à l'avance. Dans ce cas, nous pouvons utiliser, au début du processus de classification, la technique de la segmentation pour créer des groupes qui représenteront les classes. Une technique qui sera expliquée dans la prochaine section.

### 2.5.2. La segmentation

L'opération de segmentation, appelée aussi clustering, a pour but de grouper les objets similaires dans un même groupe, ou cluster, d'une façon à ce que les membres de chaque cluster soient le plus homogène possible et très hétérogène des membres des autres clusters. Elle peut servir à créer des classes, comme mentionné dans la précédente section, comme elle peut également servir à compresser les données puisque nous pourrons travailler directement avec les clusters créés au lieu des membres qui les composent. Tout comme la classification, la segmentation essaie de trouver le cluster qui correspond aux nouvelles données. La grande différence entre les deux opérations est que la classification se base sur un apprentissage par les exemples tandis que la segmentation se base sur un apprentissage par observation [33]. Des observations qui permettent de calculer une distance métrique entre les différentes entrées.

Les méthodes de segmentation peuvent être groupées en deux principales catégories, les méthodes de partitionnement et les méthodes hiérarchiques [45].

- Les méthodes de partitionnements : ayant des entrées sous forme de n-tuples, ces méthodes créent k partitions avec $k \leq n$. Chaque partition représente un cluster avec les deux conditions : Un cluster possède au moins un objet et un objet appartient à un et un seul cluster. Il faut noter que la deuxième condition n'est pas respectée dans la segmentation floue «Fuzzy Segmentation» [46]. Le k est soit définie manuellement à l'avance, ou trouvé automatiquement.

  Dans ce type de segmentation, nous commençons par créer k partitions initiales, puis on améliore ce partitionnement itérativement en déplaçant les objets d'une partition à une autre de telle sorte qu'un objet soit le plus proche aux autres objets de son groupe et le

plus loin des objets des autres groupes. L'algorithme K-means [28], qui sera détaillé au chapitre 4, fait partie de ce type de segmentation.

- Les méthodes hiérarchiques : Une pareille méthode crée une décomposition hiérarchique de l'ensemble des objets de données. Elle peut être une méthode d'agglomération ou de division selon comment la décomposition hiérarchique est formée. Les méthodes d'agglomération commencent en considérant chaque objet comme un cluster. Ensuite, elles commencent à fusionner successivement les clusters qui sont proches l'un de l'autre jusqu'à ce qu'on n'ait qu'un seul cluster représentant le niveau supérieur ou jusqu'à ce qu'une condition d'arrêt soit satisfaite. Par contre, les méthodes de division commencent par un seul cluster qui englobe tous les objets. Ensuite, dans chaque itération un cluster est divisé en plus petits clusters jusqu'à ce que chaque cluster n'est plus composé que d'un objet ou jusqu'à ce qu'une condition d'arrêt soit satisfaite.

Il est à noter que pour des raisons de performances, nous avons choisi, dans notre approche proposée dans ce mémoire, d'utiliser une méthode de division qui commence par un seul cluster représentant les vingt-quatre heures de la journée, puis le diviser en plusieurs clusters de plus petites tailles où chaque cluster représente un intervalle de temps contenant les différentes heures de débuts d'une activité. Cette segmentation sera expliquée plus en détail dans le Chapitre 4.

### 2.5.3. La découverte de règles d'association

La découverte de règles d'association, une des plus importantes opérations de la fouille de données temporelles, a été introduite pour la première fois, dans le travail de Agrawal *et al.*[39]. Elle vise à extraire des corrélations intéressantes, des motifs fréquents, des motifs, des associations ou des structures informelles parmi les ensembles d'articles sauvegardés dans une base de données transactionnelle, schématisée dans le Tableau 2.2, ou d'autres sources de données [48].

| Numéro – id du Client | Le temps de la transaction | Les produits achetés |
|---|---|---|
| 1 | Octobre 23' 02 | 30 |
| 1 | Octobre 23' 02 | 90 |
| 2 | Octobre 23' 02 | 10, 20 |
| 2 | Octobre 23' 02 | 30 |
| 2 | Octobre 23' 02 | 40, 60, 70 |
| 3 | Octobre 23' 02 | 30, 50, 70 |
| 4 | Octobre 23' 02 | 30 |
| 4 | Octobre 23' 02 | 40, 70 |
| 4 | Octobre 23' 02 | 90 |
| 5 | Octobre 23' 02 | 90 |

**Tableau 0.2: Un exemple d'une BD transactionnelle.**

Le Tableau 2.2, représente une base de données créée en enregistrant les transactions d'achat de livres en ligne. En appliquant les techniques de découverte de règles d'association sur cette base de données, nous pouvons extraire des règles très intéressantes. Par exemple, Quand le livre 70 est acheté, à 66% le livre 40 est aussi acheté, et à 33% le livre 30 est également acheté.

Des règles qui permettent aux sites de commerces électroniques de proposer aux clients, lors de l'achat du livre 70, des livres qu'ils sont susceptibles d'acheter, le 40 puis le 30, ce qui permet dans l'éventualité d'augmenter les ventes.

Ce genre de règles est extrait en appliquant nos algorithmes sur des séquences temporelles. Des règles encore plus précises peuvent être extraites si on les applique sur des séries temporelles. Par exemple, si un client achète le livre 30, il est susceptible d'acheter le livre 90 au cours du même mois.

La plus part des algorithmes de cette opération sont basés sur l'algorithme Apriori [76]. Un algorithme qui est créé sur les observations suivantes :

- Un article est fréquent si sa fréquence est supérieure ou égale à la fréquence minimale précisée par l'utilisateur. L'utilisateur est donc appelé à spécifier la fréquence minimale dès le départ.
- Si un article n'est pas fréquent, alors sa jointure avec un autre article ne sera pas fréquente elle aussi (selon la propriété Apriori). Une observation très importante qui nous évite de perdre le temps en prenant en considération tous les articles et non pas juste les fréquents.

Pour mieux comprendre cet algorithme et ses différentes étapes, prenons l'exemple cité dans le livre de Han et al.[34]. Le Tableau 2.3 représente la base de données transactionnelle sur laquelle on a appliqué l'algorithme Apriori. Alors que, La Figure 2.11 représente les différentes étapes de cet algorithme sachant que la fréquence minimale (le support minimal) est fixée à 2.

| *TID* | *Les items* |
|---|---|
| T100 | I1, I2, I5 |
| T200 | I2, I4 |
| T300 | I2, I3 |
| T400 | I1, I2, I4 |
| T500 | I1, I3 |
| T600 | I2, I3 |
| T700 | I1, I3 |
| T800 | I1, I2, I3, I5 |
| T900 | I1, I2, I3 |

**Tableau 0.3: Exemple de base de données transactionnelle**

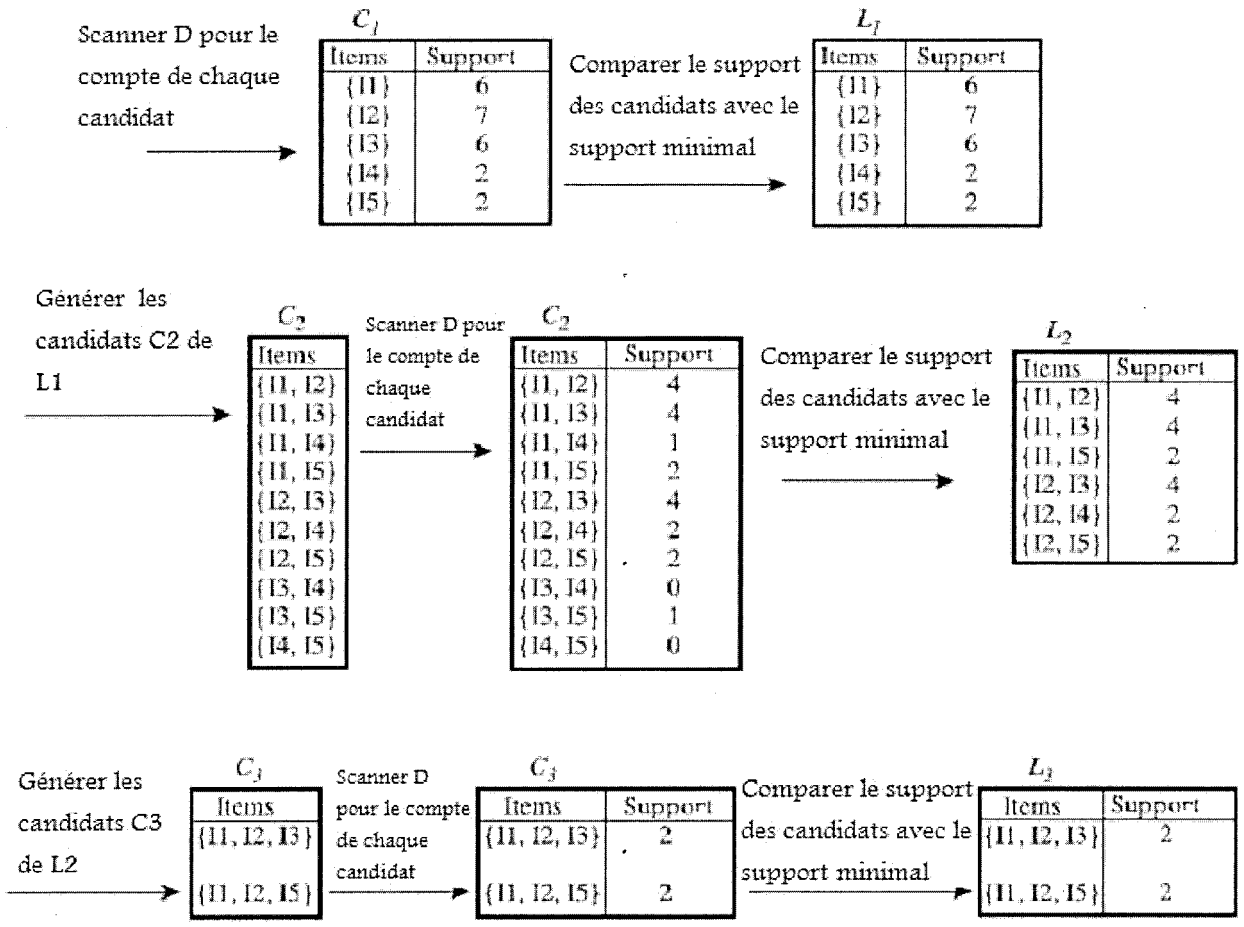



**Figure 0.11: Les étapes de l'algorithme Apriori**

La première étape, C1, consistait à parcourir la base de données D pour trouver tous les articles ainsi que leur fréquence. Pour créer L1, Tous les articles dont la fréquence est inférieure à 2 sont éliminés. C2 est créée en faisant toutes les jointures possibles entre L1 et L1, pour ne travailler qu'avec les articles fréquents. Nous supposons qu'un ordre lexicographique existe, I1 I2, de cette façon nous éviterons la jointure I2 I1 qui a la même signification que la première. Ainsi de suite jusqu'à L3.

Il reste à noter que le résultat final sera composé de la liste des séquences d'articles dont la fréquence est supérieure ou égale à 2 : ({I1}, {I2}, {I3}, {I4}, {I5}, {I1, I2}, {I1, I3}, {I1, I5},

{I2, I3}, {I2, I4}, {I2, I5}, {I1, I2, I3}, {I1, I2, I5}). Parfois, nous sommes intéressés juste par les séquences maximales qu'on appelle aussi « closed frequent pattern» [49, 71], qui sont les séquences qui n'appartiennent pas à des séquences plus longues qui apparaissent le même nombre de fois. Le résultat du même exemple sera alors : ({I1}, {I2}, {I3}, {I1, I2}, {I1, I3}, {I2, I3}, {I2, I4}, {I1, I2, I3}, {I1, I2, I5}).

De nos jours, l'opération de découverte de motifs est utilisée dans différents domaines, découvrir les motifs d'accès d'un utilisateur aux sites Web, l'utilisation de l'historique des symptômes pour prédire un certain type de maladies ou juste faciliter le contrôle de stock, etc. Pour la reconnaissance d'activités au sein d'un habitat intelligent, cette opération peut s'avérer très utile. Nous avons essayé de l'utiliser pour créer les plans d'activités. La méthode ainsi que l'algorithme que nous avons choisi seront expliqués dans le Chapitre 4.

## 2.6. Conclusion

Ce chapitre avait pour objectif de définir et d'expliquer les différentes techniques de la fouille de données temporelles. Des techniques que nous utilisons pour implémenter notre approche qui propose une solution à la problématique de la réduction du nombre d'hypothèses. Nous avons commencé par définir la fouille de donnée en générale; puis, nous avons expliqué que c'est à cause de la complexité des données temporelles, des données que nous avons détaillées d'ailleurs, qu'un nouveau domaine a vu le jour portant le nom de la fouille de données temporelles. Il a aussitôt fait ces preuves dans plusieurs domaines; ce qui nous a encouragé à l'utiliser. Le reste du chapitre était consacré à ses différentes étapes et opérations.

Plusieurs livres ont été consacrés à la fouille de données temporelles; ce chapitre se veut d'expliquer brièvement le processus d'un programme de la fouille de données temporelles ainsi que les différentes possibilités et opérations qu'offre ce domaine.

# Chapitre 3

# Les principaux travaux sur la reconnaissance d'activités utilisant la FDT

## 3.1. Introduction

Reconnaitre l'activité est un acte inné chez l'être humain. En effet, à chaque fois qu'on observe une personne, on ne peut s'empêcher de penser à ce qu'elle est en train de faire ou ce qu'elle a l'intention de faire. En voyant, par exemple, une personne à 9h00 se saisir d'une tasse, nous pouvons associer cette action, avec une certaine certitude, à l'activité *prendre petit déjeuner*. Dans le domaine de l'intelligence ambiante, nous essayons de doter un agent artificiel de cette faculté d'inférer l'activité entamée à partir des actions observées. Comme l'être humain ne peut penser qu'aux activités qu'il connait déjà, avec la possibilité d'en apprendre des nouvelles, l'agent artificiel, que nous appelons aussi agent ambiant, doit posséder des plans d'activités et doit être capable d'en créer des nouveaux. La reconnaissance d'activité pour l'agent ambiant revient donc à sélectionner, avec une certaine certitude, une activité parmi ces plans

d'activités. Cette sélection peut se baser sur différentes informations et peut être effectuée de différentes manières; ce qui explique la diversité des recherches et la vaste littérature concernant ce sujet. Durant ce chapitre, nous allons présenter les différentes approches existantes, tout en se focalisant sur celles utilisant les techniques de la fouille de données temporelles. Les données temporelles s'avèrent d'une grande importance dans le processus de recherche d'activité. Si nous revenons à l'exemple cité ci-haut, et que l'action *prendre tasse* était détecté à 22h00, l'activité qui peut se dégager de ces deux observations ne serait plus *prendre petit déjeuner* mais plutôt *faire la vaisselle*.

Ce chapitre sera donc divisé en deux parties. La première présentera plus en détail la reconnaissance d'activités et abordera les approches classiques utilisées pour répondre à cette problématique. La deuxième partie, quant à elle, sera consacrée aux principales approches utilisant la fouille de données temporelles pour la reconnaissance d'activité.

## 3.2. La reconnaissance d'activité

La reconnaissance d'activité est un domaine très actif de l'intelligence ambiante, pourtant il a été définit depuis longtemps dans le domaine de l'intelligence artificielle et a été connu sous le nom de reconnaissance de plans. C'est Schmidt [15], qui l'a défini pour la première fois par le fait de « prendre en entrée une séquence d'actions effectuée par un acteur et d'inférer le but envisagé par l'acteur et organiser la séquence d'actions en une structure de plan». De cette définition nous pouvons ressortir la relation évidente entre la reconnaissance d'activité et la planification qui d'après P. Roy [50] « consiste en l'élaboration d'une stratégie prenant la forme d'une séquence d'actions-plans permettant de solutionner un problème donné, de façon à atteindre un objectif visé ». En effet, depuis 2003 Russel et al. [52] ont considéré le problème de

la reconnaissance d'activité comme l'inverse de celui de la planification. Si dans cette dernière l'agent artificiel connait le but ou l'objectif visé et essaie de trouver le plan d'actions qui permettra d'atteindre cet objectif, dans la reconnaissance d'activité, par contre, l'agent connait quelques actions et essaie de trouver l'objectif visé ainsi que les autres actions qui permettent son accomplissement. La figure 3.1, prise du travail de Bouchard [3], illustre bien ces propos et montre d'une vue globale le processus de reconnaissance d'activité.

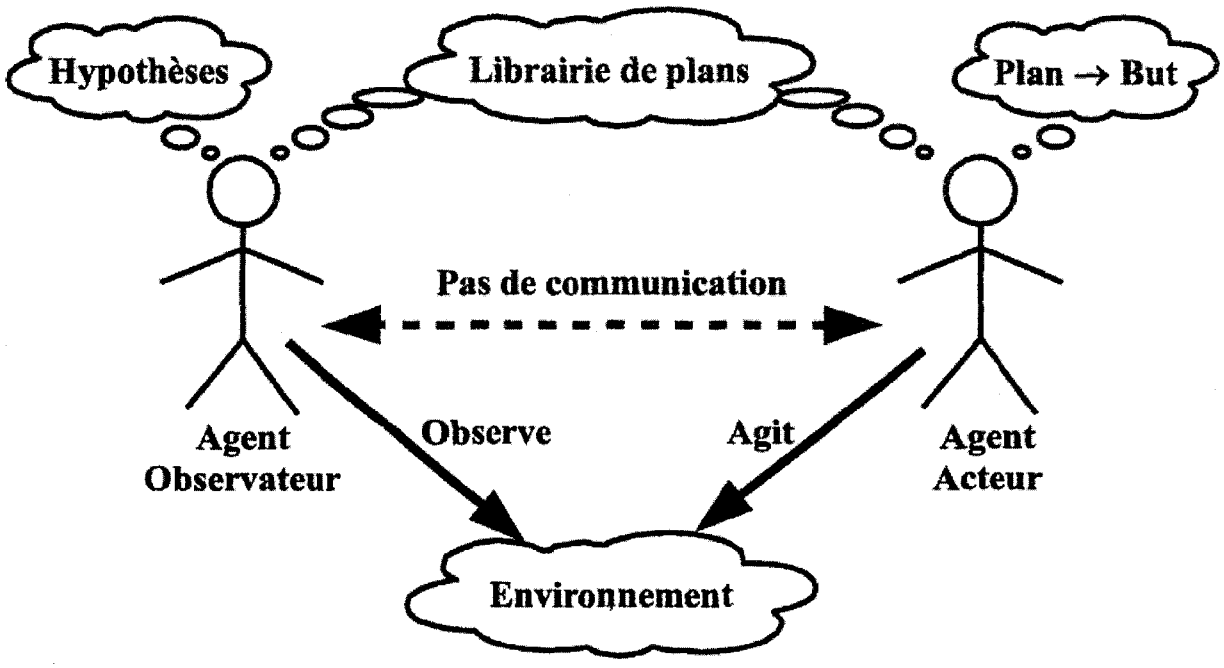



**Figure 3.1: Vue globale du processus de la reconnaissance d'activité**

Pour mieux comprendre cette Figure nous allons la projeter sur notre cas d'étude qui est, rappelons-le, la reconnaissance d'activités d'un patient de la maladie d'Alzheimer (Agent Acteur) au sein d'un habitat intelligent (Environnement) par un agent ambiant (Agent Observateur). L'agent acteur décide du but à accomplir à partir de la librairie de plan et de son plan d'actions. En effectuant ses premières actions, il agit sur l'environnement, le seul point partagé avec l'agent observateur parce qu'il n'y a pas de communication directe entre les deux ce

qui nous ramène à une reconnaissance à l'insu qui sera détaillée dans la section 3.3.2. L'agent observateur s'aperçoit des changements de l'environnement, grâce aux différents capteurs installés dans l'habitat, et les traduits en actions. À partir des actions détectées, de la librairie de plans et de son système d'inférence, l'agent ambiant crée des hypothèses sur l'objectif visé par l'agent acteur. Il reste juste à préciser que la librairie de plans est composée des différentes activités, ainsi que la liste ordonnée des actions qui composent chacune d'elle, que le patient peut effectuer à l'intérieur de l'habitat intelligent et nous sommes plus précisément intéressés par les activités de la vie quotidienne.

### 3.2.1. Les activités de la vie quotidienne (AVQ)

Dans ce mémoire nous utilisons souvent le terme d'activité alors que pour être plus précis nous devons utiliser activité de vie quotidienne (AVQ). La notion d'AVQ a été définie en 1963 par Katz *et al.*[54] comme étant l'ensemble des activités qu'un individu effectue comme routine pour prendre soin de lui-même, par exemple, préparer à manger, s'habiller, se laver, etc. La définition des AVQ s'est avérée d'une très grande importance parce qu'elle a permis de mesurer le niveau d'autonomie et le fonctionnement physique des personnes âgés et des sujets souffrants de maladies chroniques par le biais de l'index de Katz. Cet index est calculé, dans sa version originale, par l'attribution d'un 1 au cas où le sujet effectue avec succès et sans assistance chacune des AVQ suivantes :

- Se laver
- S'habiller
- Se rendre aux toilettes
- Les transferts
- La marche
- L'aide pour l'alimentation

Selon le score de l'index de Katz nous pouvons détecter le niveau de dépendance du sujet et lui offrir ainsi le type d'assistance adéquate. Au fil des ans et avec l'évolution du domaine de la santé, d'autres tests du niveau d'autonomie ont vu le jour [6, 55]; mais, ils restent tous baser sur les AVQ qui sont, de nos jours, classés en trois différents types [56]:

- Les activités de vie quotidienne basiques (AVQB) : c'est l'ensemble des activités fondamentales et obligatoires pour combler les besoins primaires de la personne. Des activités qui sont composées seulement de quelques étapes et n'exigent pas de planification réelle, par exemple, aller à la salle de bain, se déplacer sans appareils aidantes, manger, etc.
- Les activités de vie quotidienne instrumentales (AVQI) : ce sont des activités qui demandent une forme basique de planification et la manipulation d'objets. Une personne capable d'effectuer les AVQI veut dire qu'elle est autonome et peut vivre toute seule chez elle. Les AVQI sont plus complexe et se composent de plus d'étape que les AVQB. Dans cette catégorie nous trouvons des activités du genre : préparer un repas, appeler avec un téléphone, gérer son argent, faire des achats, etc.
- Les activités de vie quotidienne augmentées (AVQA) : correspondent aux tâches qui exigent une forme d'adaptation de la part de l'individu à cause de la nature de l'environnement. Par exemple, faire ces achats par Internet sur un site qui peut modifier son design et la position des liens.

Dans le cadre de notre étude, et dans toutes les études relatives aux habitats intelligents, nous nous intéressons plus précisément aux AVQI surtout que c'est la réussite d'exécution de ces activités qui fait en sorte que le sujet soit autonome. De plus, si le sujet est incapable d'effectuer les AVQB, il aura besoin d'une assistance classique avec une personne aidante à ces côtés.

Quant aux AVQA, elles sont beaucoup plus complexes et changent parfois de plan d'actions; alors, il est très difficile d'assister le sujet pour effectuer de pareils activités surtout avec les moyens technologiques utilisés jusqu'à présent.

### 3.2.2. Les types de reconnaissance d'activité

Dans la section précédente, nous venons de voir l'importance de connaître le niveau d'autonomie de l'agent acteur afin de lui adapter la forme d'assistance adéquate. Comme la reconnaissance d'activité est l'élément clé du processus d'assistance, il faut aussi, dès le départ, décider du type de reconnaissance que nous allons utiliser. C'est le type de relation entre l'agent acteur et l'agent observateur qui précise le type de la reconnaissance d'activité. Dans le travail de Geib et Goldman [58] nous trouvons trois types de reconnaissance d'activité : la reconnaissance communicative, contradictoire et la reconnaissance à l'insu.

- La reconnaissance d'activité communicative : Comme son nom l'indique, dans ce type de reconnaissance, il existe une sorte de communication entre l'agent acteur et l'agent observateur. L'acteur est donc au courant et consentant du fait qu'il est observé afin de reconnaître ses activités. Il peut même aller jusqu'à adapter son comportement pour faciliter la reconnaissance. Nous pensons, par exemple, à la phase d'essai de notre système de reconnaissance d'activité au laboratoire d'intelligence ambiante pour la reconnaissance de l'activité (LIARA) à l'université du Québec à Chicoutimi, où nous refaisions une action plusieurs fois jusqu'à ce que nous soyons sûrs que le système l'a détectée.

Ce type de reconnaissance est facile à installer; malheureusement, il n'est pas adapté à notre cas d'étude. Le patient observé oublie parfois l'activité qu'il est en train de faire, alors comment lui demander de se rappeler de nous aider à la reconnaître.

- La reconnaissance d'activité contradictoire : Ce type de reconnaissance est exactement le contraire du premier. Oui l'acteur est au courant qu'il est observé, mais il n'est pas consentant à reconnaître ses activités. Loin de là, il va même jusqu'à agir délibérément pour induire l'observateur en erreur dans sa reconnaissance. Ce type de reconnaissance est très utilisé dans le domaine des jeux vidéo où entre ennemi nous savons que nous sommes observés et nous faisons des actions juste pour dissimuler notre stratégie d'attaque par exemple.

  Ce type de reconnaissance n'est pas adapté, non plus, à notre cas d'étude. Même si, parfois le patient effectue des actions hors du plan d'action et qui peuvent introduire à l'erreur l'agent ambiant, mais il le fait non pas délibérément mais sans s'en rendre compte.

- La reconnaissance d'activité à l'insu : Ce type de reconnaissance correspond au dernier cas de figure restant où l'agent acteur n'est pas au courant qu'il est observé. Cela veut dire qu'il ne va pas influencer la décision de l'agent observateur, ni en l'aidant, ni en l'induisant à l'erreur.

  Ce dernier type de reconnaissance est le plus adapté à l'assistance cognitive au sein d'un habitat intelligent, mais pour des raisons d'éthique le patient doit être informé qu'il est observé.

  Le patient est donc averti qu'il est observé comme il est avisé de ne pas influencer, d'une façon intentionnelle, le processus de reconnaissance.

Maintenant que nous avons défini la reconnaissance d'activité, les activités de la vie quotidienne et les types de reconnaissance d'activité, nous pouvons présenter les principales approches existantes qui essaient de répondre à cette problématique de reconnaissance d'activité au sein d'un habitat intelligent et qui utilisent des techniques de la fouille de données temporelles.

## 3.3. Les principales approches utilisant la fouille de données temporelles

Plusieurs travaux ont été menés pour répondre à cette problématique de reconnaissance d'activité. Différentes approches en ont découlé. Ces approches peuvent être classifiées selon les techniques qu'elles utilisent. Les approches logiques basées sur la logique du premier ordre. Elles visent à sélectionner, parmi les activités enregistrées dans les plans d'activités, par une série de déductions logiques, l'ensemble des activités possibles qui peuvent expliquer un ensemble d'actions observées. Nous pouvons considérer les travaux de Kautz [8] comme les travaux fondateurs de ce type d'approches. Les approches probabilistes basées sur des modèles markoviens [17] ou des réseaux bayésiens [16] où une probabilité initiale est assignée manuellement à chaque activité; puis, ces probabilités sont mises à jour en fonction de l'ensemble des nouvelles observations. Les récentes approches, Quant à elles, exploitent les puissantes techniques de la fouille de données et leur capacité à analyser un grand volume de données tout en profitant de l'aspect spatial ou temporel pour réduire la complexité de la reconnaissance d'activité.

Dans notre étude, nous allons nous focaliser sur les approches utilisant la fouille de données temporelles. Des approches qui peuvent être classifiées en deux catégories selon les types de senseurs installés dans l'habitat intelligent, la reconnaissance d'activité basée sur la

vision «Vision-based activity recognition» et la reconnaissance d'activité basée sur les senseurs «Sensor-based activity recognition» [58].

### 3.3.1. La reconnaissance d'activité basée sur la vision

Les approches de reconnaissance d'activité basée sur la vision sont caractérisées par l'utilisation de caméras pour détecter les changements de comportement de l'agent acteur ou de son environnement [58]. En effet, les caméras fournissent des données, sous forme d'images, qui permettent la détection des actions primitives qui composent les activités. L'utilisation des caméras permet d'exploiter les techniques du domaine de la vision par ordinateur [59] pour l'analyse des observations visuelles afin de reconnaitre des motifs. Ces dernières techniques ont déjà fait leur preuve dans des domaines comme, Interface Homme-Machine, le Design d'Interface Utilisateur, l'apprentissage automatique et la surveillance.

Les premiers travaux de la reconnaissance d'activité basée sur la vision évitaient de reconnaitre chaque action primitive, une tâche très complexe avec les techniques traditionnelles, et essayaient plutôt d'extraire quelques caractéristiques qui peuvent être reliées aux actions primitives et aider ainsi à la reconnaissance d'activité. Le travail de Duong *et al.* [60] en est un bon exemple. Dans ce travail, plusieurs caméras installées aux coins de la chambre, observent l'agent acteur effectuant ces activités. La chambre est divisée en plusieurs cellules d'un mètre carré. Les caméras sont utilisées juste pour détecter le mouvement et retourner la liste des cellules visitées par l'agent acteur. Comme certaines cellules comportent des objets intéressants (micro-onde, four, etc), la visite de telles cellules peut être reliée à une action primitive (ouvrir le four, utilisation du micro-onde,etc). Donc, aucune reconnaissance des actions primitives n'est

effectuée; mais, la liste des cellules visitées est utilisée comme entrée pour la reconnaissance d'activité.

Avec le développement des techniques utilisées et l'émergence de la fouille de données temporelles, les nouvelles approches s'attaquent à la détection directe des actions primitives des données enregistrées par les caméras. Le travail de Spriggs *et al.*[20], détaillé dans la prochaine section, présente une approche qui utilise les techniques de la fouille de données temporelles pour reconnaitre l'activité à partir de séquences vidéo enregistrées par une caméra.

#### 3.3.1.1.**Approche de Spriggs et al.**

Dans le travail de Spriggs *et al.*[20], une caméra portable ainsi qu'une unité de mesure inertielle (IMUs) sont utilisées pour observer l'agent acteur dans un contexte de préparation de recettes dans un environnement normal. Les données envoyées par ces senseurs sont stockées puis, analysées afin d'effectuer une segmentation temporelle où chaque segment représente une action. Ces segments sont ensuite classifiés pour permettre la reconnaissance d'activité entamée. La Figure 3.2 montre les senseurs utilisés ainsi que le résultat de la segmentation temporelle sur les deux différents types de données.

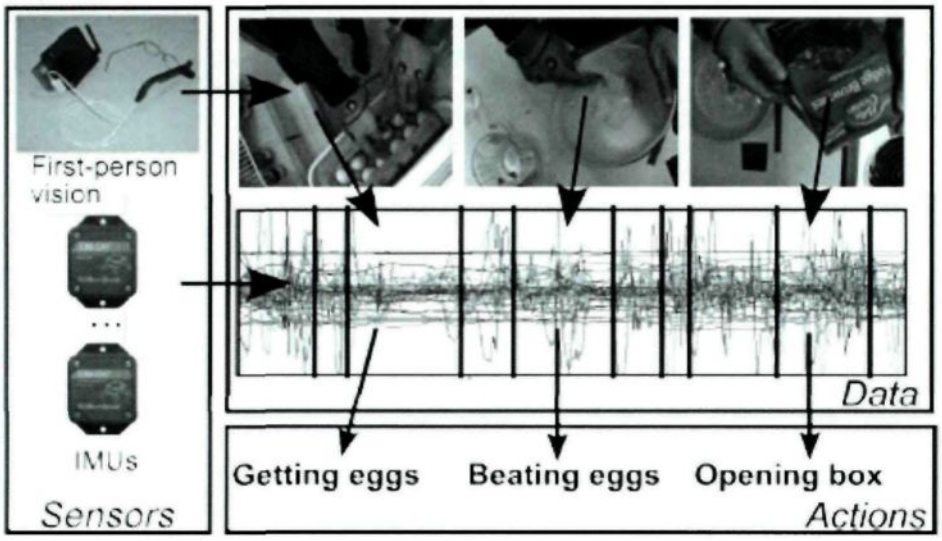



**Figure 3.2: Segmentation temporelle sur vidéo et signaux**

Comme le montre la Figure 3.2, la caméra offre une vision à la première personne; ce qui aide à comprendre les intentions de l'agent acteur. Normalement, ce qu'une personne voit a une relation avec ce qu'elle est en train de faire. La segmentation temporelle revient donc à attribuer chaque trame de la séquence vidéo, enregistrée par la caméra à la première personne, à l'étape adéquate de la préparation de la recette. Cette segmentation est effectuée en plusieurs étapes :

- L'étape de préparation des données : Comme une action peut se dérouler sur des arrières plans différents, des trames peuvent sembler très différentes même si elles représentent la même action. Pour cette raison, il faut prendre l'idée générale du contenu essentiel de la trame «gist of frame».

- L'étape de transformation et de réduction des données : le «gist of frame» est discrétisé en blocs de 4x4. Ils sont ensuite transformés en un 512 dimensionnel vecteur pour chaque trame vidéo. Pour réduire la dimensionnalité une analyse en composantes principales, présentée dans le travail de Barbic et al. [62], est performée afin de garder des vecteurs de plus petites

tailles (32 ou moins). Les données sont enfin normalisées à une moyenne de 0 et une variance de 1.

- L'étape de segmentation : la segmentation est effectuée d'une manière non supervisée en utilisant une estimation par le modèle de mélange gaussien (GMM). Le GMM est une somme pondérée de M densités des composantes gaussiennes donnée par la formule suivante :

$$p(\mathbf{x}|\lambda) = \sum_{i=1}^{M} w_i \, g(\mathbf{x}|\mu_i, \Sigma_i)$$

Où X est notre vecteur de dimension 32, wi, i = 1, ... , M, sont les poids du mélange, et g(X|μi,Σi), i = 1,... , M, sont les densités des composantes gaussiennes. La densité de chaque composante est une fonction à 32-variables gaussiennes de la forme :

$$g(\mathbf{x}|\mu_i, \Sigma_i) = \frac{1}{(2\pi)^{D/2}|\Sigma_i|^{1/2}} \exp\left\{-\frac{1}{2}(\mathbf{x}-\mu_i)' \, \Sigma_i^{-1} \, (\mathbf{x}-\mu_i)\right\}$$

Avec $\mu_i$ le vecteur moyen, $\Sigma_i$ la matrice de covariance et le mélange des poids satisfait la contrainte $\sum_{i=1}^{M} w_i = 1$

Après la segmentation vient l'étape de classification qui permet de trouver la classe, ou le segment, d'une nouvelle trame enregistrée afin de reconnaitre l'activité entamée et pouvoir prédire la prochaine action. Pour la classification, Spriggs et *al.*[20] ont utilisé la méthode des k proches voisins «K-Nearest Neighbor». Le principe de cette méthode est simple, il suffit de calculer la distance entre la nouvelle trame et les centres des segments voisins pour l'attribuer au segment avec la distance minimale. Plusieurs formules permettent de calculer cette distance. Dans leur travail, ils ont choisi la distance Euclidienne qui se calcule par la formule :

$$\mathrm{d}(\mathbf{p},\mathbf{q}) = \mathrm{d}(\mathbf{q},\mathbf{p}) = \sqrt{(q_1 - p_1)^2 + (q_2 - p_2)^2 + \cdots + (q_n - p_n)^2} = \sqrt{\sum_{i=1}^{n} (q_i - p_i)^2}.$$

Où $p = (p_1,\ldots,p_n)$ et $q = (q_1,\ldots,q_n)$ deux points dans l'espace Euclidien de dimension n.

En fin, pour valider leur approche, Spriggs *et al.*[20] ont demandé à sept personnes différentes de préparer deux recettes, une omelette et des Brownies. Chacune des deux parties de l'approche a été validée séparément. Pour l'étape de segmentation, les segments créés ont été comparés à des segments créés manuellement. La Figure 3.3 montre un exemple du résultat de cette comparaison.

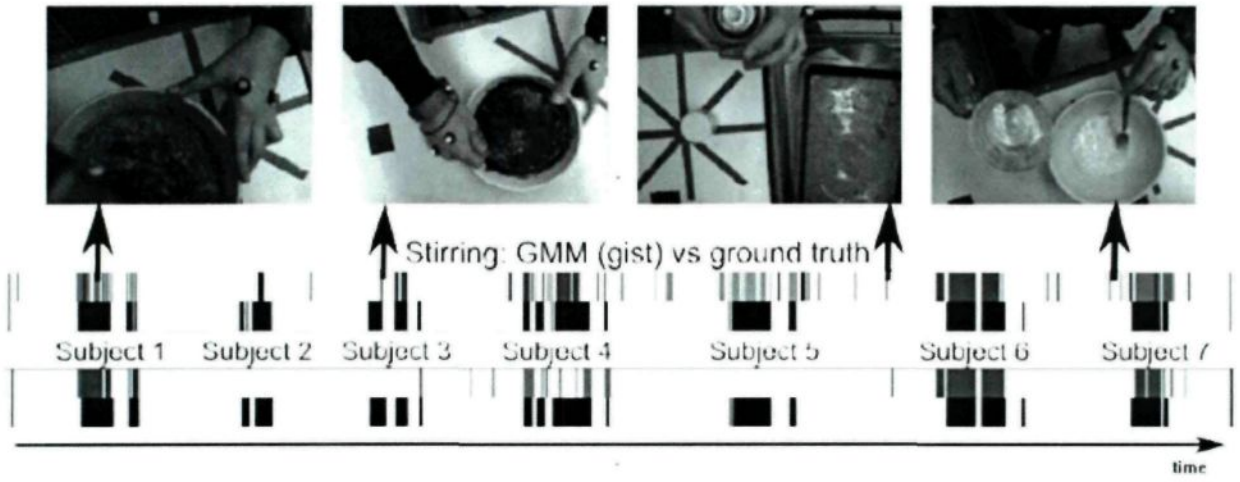



**Figure 3.3: Comparaison des segments automatiques et manuels**

La Figure 3.3 présente une comparaison entre les segments créés manuellement, en noir, et ceux créés par la segmentation non supervisée pour l'action *Agiter* pour les sept différentes personnes. 20401 trames ont été utilisées pour un taux de réussite de 70%.

En ce qui concerne l'étape de classification, la Figure 3.4 présente le résultat de la classification des trames de l'activité *préparer brownies* effectuée par la troisième personne.

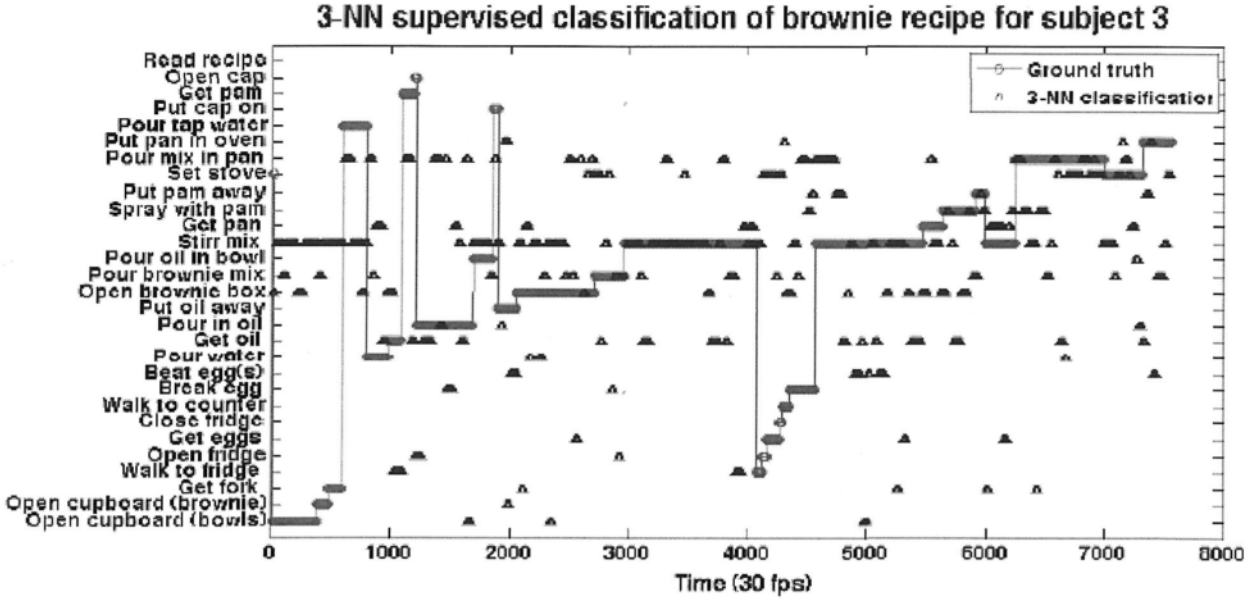



**Figure 3.4: Résultat de classification des trames**

Comme le montre la Figure 3.4, 61% des trames, en bleu, ont été bien classifiées. Nous devons noter que pour avoir un taux aussi élevé, la taille des trames n'a pas été réduite et les vecteurs auxquels elles étaient transformées sont de dimension 557.

### 3.3.1.2.Évaluation de l'approche de Spriggs et al.

Le travail de Spriggs *et al.*[20] est un exemple parfait de la puissance apportée par l'utilisation des techniques de la fouille de données temporelles. En fait, une telle approche est irréalisable avec les approches classiques. Toutefois, ce travail doit être pris comme une base pour des futures recherches qui doivent répondre à plusieurs questions. Par exemple, comment une trame va être classifiée si l'agent acteur change les outils utilisés pour effectuer l'action. Nous pensons à l'action *Agiter*, si l'agent acteur utilise un bol de différente forme et de différente couleur, la trame sera très différente de celles classifiées pour la même action. La deuxième question qui se pose concerne le temps de réponse de cette approche surtout qu'on sait que le

traitement des séquences vidéo est une lourde tâche pour des applications en temps réel. D'ailleurs, Spriggs *et al.*[20] ne l'ont pas essayée dans le cadre d'une assistance instantanée.

### 3.3.2. La reconnaissance d'activité basée sur les senseurs

L'utilisation des caméras pour la reconnaissance d'activité est une solution qui essaie de doter la machine d'une forme d'intelligence humaine très complexe. Elle est basée sur une surveillance visuelle et des données, sous forme d'images, que la machine a beaucoup de mal à traiter. Par contre, l'utilisation des senseurs est une solution plus adaptée à la machine; puisque, les données à traiter sont soit booléennes, des senseurs à ON ou OFF, soit numériques, des mesures de distance par exemple. C'est donc tout à fait normal que les travaux de la reconnaissance d'activité basée sur les senseurs soient plus nombreux et leurs résultats plus pertinents. Ces travaux peuvent à leur tour être divisés en deux différentes catégories, les approches basées sur les senseurs portables et les approches basées sur les objets.

#### 3.3.2.1.Les approches basées sur les senseurs portables

Les approches basées sur les senseurs portables sont des approches où l'agent acteur porte sur lui une collection de senseurs. Les senseurs peuvent être mis dans ces vêtements, dans une poche ou une trousse, ou directement placés sur son corps, sur le poignet, la hanche ou le torse. Le choix de la position des senseurs doit être bien étudié parce qu'il doit assurer une bonne utilisabilité des senseurs; tout, en offrant un maximum de confort à l'agent qui doit les porter [63].

Les senseurs portables peuvent être de différents types, des accéléromètres, des gyroscopes, des magnétomètres, des étiquettes RFID, etc. Les données qu'ils émettent fournissent principalement des informations sur la pose et les mouvements de l'agent acteur. Ces informations sont typiquement utilisées pour reconnaitre des types de mouvements, marcher, courir, etc; cela n'empêche qu'elles peuvent être très utile pour la reconnaissance de certaines AVQ, se brosser les dents, écrire, utiliser un ordinateur, etc. La Figure 3.5, prise du travail de Ravi et *al.*[64], montre comment un senseur accéléromètre triaxial placé sur le corps peut aider à la reconnaissance de certaines activités.

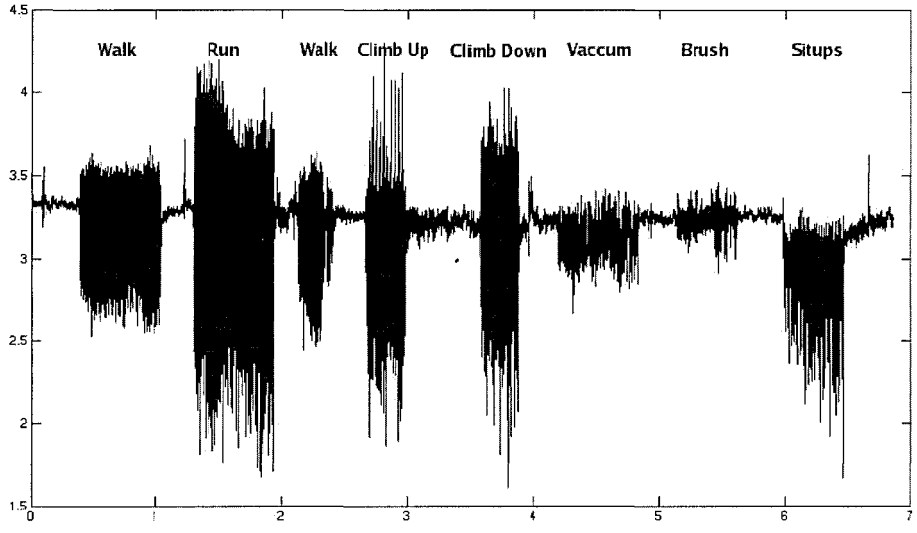



**Figure 3.5: reconnaissance d'activités par un senseur porté**

La reconnaissance d'activité basée sur les senseurs portés souffre de deux majeurs inconvénients. Premièrement, la plus part des senseurs portables ne sont pas applicables dans la vie de tous les jours à cause de problèmes techniques dus à leur taille, la durée de vie de la batterie ou la facilité d'utilisation, en plus de la question générale de l'acceptabilité ou la volonté de l'agent acteur à les porter. Le second inconvénient réside dans la complexité de la plus part des AVQ qui impliquent des mouvements physiques compliqués et une interaction encore plus

complexe avec des objets du monde réel. Pour toutes ces raisons, la reconnaissance d'activité basée sur les senseurs portables est souvent combinée avec celle basée sur les objets que nous allons détailler dans la prochaine section.

### 3.3.2.2.Les approches basées sur les objets

Les approches basées sur les objets semblent les plus adéquates à notre cas d'étude puisqu'un patient de la maladie d'Alzheimer risque fort d'oublier de porter les senseurs, en plus nous avons déjà opté pour une reconnaissance d'activité à l'insu. Dans ces approches, les senseurs sont posés sur les objets d'une façon totalement transparente à l'agent acteur. Ils peuvent être de différents types et effectuer différentes mesures, des commutateurs de contact pour donner l'état, fermer/ouvert, des portes et des armoires, des tapis de pression pour indiquer la position de la personne dans l'habitat ou détecter s'il est assis sur un canapé ou allongé sur son lit, des étiquettes RFID pour donner l'emplacement d'objets comme une tasse ou un bol, des capteurs de température, d'humidité ou à flotteur pour détecter si le four, la douche ou les toilettes sont utilisés, etc [65].

Chaque senseur effectue ses propres mesures et les envoie à une station centrale pour y être sauvegardées, ce qui fait de cette dernière un nœud d'un réseau sans fil. Les données peuvent passer d'un nœud à un autre jusqu'à la station centrale. Nous parlons d'un réseau de senseur de type ad hoc [66]. Une étude approfondie doit normalement être faite avant le choix des senseurs et de leur emplacement.

Plusieurs travaux ont essayé de répondre à la problématique de reconnaissance d'activité en se basant sur ce genre d'approche. Le grand volume de données envoyées par les différents

senseurs rend l'utilisation des techniques de la fouille de données temporelles presque indispensable. Le travail de Jakkula et Cook [67], qui sera détaillé dans la prochaine section, montre le résultat de l'utilisation de ces techniques.

#### 3.3.2.2.1 Approche de Jakkula et Cook

Le travail présenté par Jakkula et Cook [67] exploite les techniques de la fouille de données temporelles pour faire des prédictions sur les activités et pour la détection des anomalies. Leur approche, comme toutes les applications de la fouille de données temporelles, travaille directement sur les données envoyées par les senseurs. L'analyse de ces données permet l'extraction d'importantes relations entre les actions de l'agent acteur. Les relations sont de la forme, l'action *allumer télé* s'effectue après l'action *s'assoir sur le divan.* Donc, si nous détectons que l'action *s'assoir sur le divan* vient de s'effectuer, nous pouvons prédire que la prochaine action sera *allumer télé*. Par contre, si c'est l'action *allumer télé* qui a été détectée, on peut juger qu'il y a eu erreur d'exécution parce que l'action *s'assoir sur le divan* devait la précéder.

Cette approche est donc basée sur les différentes relations temporelles définies par Allen [68] et se déroule en plusieurs étapes :

- Étape de transformation : Dans cette étape, les données envoyées par les senseurs, qui sont sous la forme présentée dans la Figure 3.6, sont transformées pour créer des intervalles temporels pour chaque action.

| Raw Sensor Data | | |
|---|---|---|
| Timestamp | Sensor State | Sensor ID |
| 3/3/2003 11:18:00 AM | OFF | E16 |
| 3/3/2003 11:23:00 AM | ON | G12 |
| 3/3/2003 11:23:00 AM | ON | G11 |
| 3/3/2003 11:24:00 AM | OFF | G12 |
| 3/3/2003 11:24:00 AM | OFF | G11 |
| 3/3/2003 11:24:00 AM | ON | G13 |
| 3/3/2003 11:33:00 AM | ON | E16 |
| 3/3/2003 11:34:00 AM | ON | D16 |
| 3/3/2003 11:34:00 AM | OFF | E16 |

**Figure 3.6: Exemple de données envoyées par les senseurs**

La transformation est effectuée en considérant le changement d'état (ON/OFF) des senseurs comme borne de l'intervalle. Le résultat de cette transformation est présenté dans la Figure 3.7.

| Identify Time Intervals | | | |
|---|---|---|---|
| Date | Sensor ID | Start Time | End time. |
| 03/02/2003 | G11 | 01:44:00 | 01:48:00 |
| 03/02/2003 | G19 | 02:57:00 | 01:48:00 |
| 03/02/2003 | G13 | 04:06:00 | 01:48:00 |
| 03/02/2003 | G19 | 04:43:00 | 01:48:00 |
| 03/02/2003 | H9 | 06:04:00 | 06:05:00 |
| 03/03/2003 | P1 | 10:55:00 | 17:28:00 |
| 03/03/2003 | E16 | 11:18:00 | 11:34:00 |
| 03/03/2003 | G12 | 11:23:00 | 11:24:00 |

**Figure 3.7: Les différents intervalles créés**

Le senseur G11 peut être associé à la télé, Donc la première ligne peut être lue, la télé a été allumée le 03/02/2003 de 01h44 jusqu'à 01h48.

- L'étape de recherche de relations : Durant cette étape, on se base sur la comparaison des bornes des deux intervalles temporels pour décider du type de relation existante entre les deux. La Figure 3.8 montre les treize relations temporelles proposées par Allen, ainsi que les formules mathématiques qui les définissent.

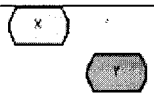
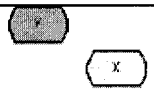
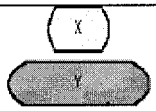
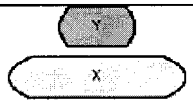
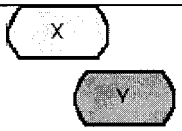
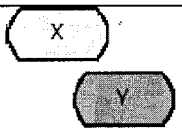
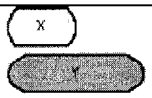
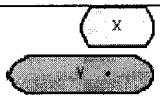
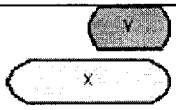
| Temporal Relations | Pictorial Representation | Interval constraints |
|---|---|---|
| X Before Y |  | Start(X)<Start(Y); End(X)<Start(Y) |
| X After Y |  | Start(X)>Start(Y); End(Y)<Start(X) |
| X During Y |  | Start(X)>Start(Y); End(X)<End(Y) |
| X Contains Y |  | Start(X)<Start(Y); End(X)>End(Y) |
| X Overlaps Y |  | Start(X)<Start(Y); Start(Y)<End(X); End(X)<End(Y) |
| X Overlapped-By Y |  | Start(Y)<Start(X); Start(X)<End(Y); End(Y)<End(X) |
| X Meets Y |  | Start(Y) = End(X) |
| X Met-by Y |  | Start(X)= End (Y) |
| X Starts Y |  | Start(X)=Start(Y); End(X)≠End(Y) |
| X started-by Y |  | Start(Y)=Start(X); End(X)≠End(Y) |
| X Finishes Y |  | Start(X)≠start(Y); End(X) = End (Y) |
| X Finished-by Y |  | Start(X)≠Start(Y); End(X)=End(Y) |
| X Equals Y |  | Start(X)=Start(Y); End(X)=End(Y) |

**Figure 3.8: Les relations temporelles d'Allen**

Comme ces relations vont être utilisées pour la prédiction, Jakkula et Cook [67] jugent que les quatre relations, Before, Contains, Overlaps et Meets ne seront pas utile pour cette tâche. Donc, pour alléger la procédure de recherche et pour gagner du temps, il vaut mieux les ignorer. Pour les relations restantes, et comme le montre la Figure 3.4, il suffit de comparer les bornes des deux intervalles X et Y pour décider de la relation existante entre eux. Par exemple, si le temps de début de X est inférieur à celui de Y et que le temps de fin de X est inférieur au temps de début de Y, la relation que nous pouvons déduire est que X s'effectue avant Y.

L'algorithme utilisé pour ces deux premières étapes est montré à l'Algorithme 3.1, tandis qu'un exemple des relations trouvées est présenté à la Figure 3.9.

**Input**: data timestamp, event name and state
  **Repeat**
    **While** [Event && Event + 1 found]
      Find paired "ON" or "OFF" events in data to
          determine temporal range.
      Read next event and find temporal range.
      Identify relation type between event pair from
          possible relation types (see Table 1).
      Record relation type and related data.
      Increment Event Pointer
**Loop Until** end of input.

**Algorithme 3.1: Trouver les intervalles ainsi que les relations temporelles**

**Associated Temporal Relations**

| Date time | Sensor ID | Temporal Relation | Sensor ID |
|---|---|---|---|
| 3/3/2003 12:00:00 AM | G12 | DURING | E16 |
| 3/3/2003 12:00:00 AM | E16 | BEFORE | I14 |
| 3/2/2003 12:00:00 AM | G11 | FINISHESBY | G11 |
| 4/2/2003 12:00:00 AM | J10 | STARTSBY | J12 |

**Figure 3.9: Exemple de relations trouvées**

- L'étape de découverte des fréquentes relations : à l'étape précédente, un très grand nombre de relations est généré, beaucoup trop grand pour les utiliser toutes d'une façon efficace pour la prédiction. En plus, certaines relations sont créées après des erreurs d'exécution de la part de l'agent acteur ou après l'analyse de données erronées causées par des bruits de senseurs. De plus, les relations doivent résumer les habitudes de l'agent acteur. Pour toutes ces raisons, nous ne devons sélectionner que les relations qui reviennent assez souvent. L'algorithme Apriori [76], un algorithme de la découverte de motifs expliqué dans le Chapitre 2, est utilisé pour ce propos. L'algorithme 3.2 présente la façon dont Jakkula et Cook [67] l'ont implémenté.

---

$C_k$ : Candidate itemset of size k
$L_k$ : Frequent itemset of size k

L1 = { frequent items };
For (k=1; $L_k$ != ∅; k++)
    $C_{k+1}$ = candidates generated from $L_k$;
    For each day t in datasets do
        Increment the count of all candidates in $C_{k+1}$ that are in t;
    $L_{k+1}$ = candidates in $C_{k+1}$ with min_support
Return $U_k L_k$

---

**Algorithme 3.2: Sélectionner les fréquentes relations**

L'application de cet algorithme ne sert pas juste à sélectionner les relations les plus fréquentes, mais permet aussi de calculer la fréquence des relations. Cette fréquence peut être transformée pour désigner la probabilité qu'un membre de la relation arrive si on détecte l'autre membre. Cette information est très utile pour la prochaine étape.

- L'étape de prédiction et de détection d'anomalies: Dans cette étape, il suffit de calculer la probabilité qu'une action X arrive après la détection d'une action Y et ce, en additionnant la probabilité des relations qui peuvent unir les deux actions. La formule (1) est utilisée pour faire ce calcul :

(1) $$P(X|Y) = |After(Y,X)| + |During(Y,X)| + |OverlappedBy(Y,X)| + |MetBy(Y,X)| + |Starts(Y,X)| + |StartedBy(Y,X)| + |Equals(Y,X)| \,/\, |Y|$$

Ce qui rend ce calcul encore plus intéressant, c'est de pouvoir le faire après la détection de plusieurs actions. La formule (2) nous montre comment calculer la probabilité après la détection de deux actions Y et Z :

(2) $$P(X|Z \cup Y) = \frac{P(X \cap (Z \cup Y))}{P(Z \cup Y)} = P(X \cap Z) \cup \frac{P(X \cap Y)}{P(Z)} + P(Y) - P(Z \cap Y)$$
$$= P(X|Z).P(Z) + P(X|Y).\frac{P(Y)}{P(Z)} + P(Y) - P(Z \cap Y)$$

L'interprétation de cette probabilité est très simple et peut facilement nous aider à prédire la prochaine action ou détecter une erreur d'exécution. Effectivement, si cette probabilité est proche de 1, cela veut dire que l'action X a une forte probabilité qu'elle soit la prochaine à s'effectuer. En fait, nous pouvons calculer cette probabilité pour toutes les actions et prédire celle avec la plus grande probabilité. Sinon, si cette probabilité est proche de 0 et que nous détectons que c'est

l'action X qui s'est effectuée après Y, nous pouvons dire qu'il y avait erreur d'exécution de la part de l'agent acteur.

L'algorithme utilisé dans cette étape de prédiction par Jakkula et Cook [67] est l'algorithme prédicteur ALZ, présenté à l'Algorithme 3.3, qui prend en entrée les relations trouvées dans les précédentes étapes. C'est un algorithme séquentiel qui emploie l'analyse incrémentale et utilise des modèles Markoviens pour la prédiction.

```
Input: Output of ALZ a, Best rules r, Temporal dataset
While a!=null Loop
  Repeat
    Set r1 to the first event in the first best rule
    If (r1==a) Then
      If (Relation!="After") Then
        Calculate evidence (EQ 1)
        If evidence>(Mean+2 Std. Dev.) Then
          Make event in the best.rule as next predictor output
        Else
        *Get next predicted event
         Look for temporal relations based on frequency
         Calculate evidence, store in a buffer

        If again relation is after Then goto *
        Until no more "After"
        Calculate evidence
        If evidence>(Mean+2 Std. Dev.) Then predict
        Else
        Calculate evidence
        If evidence>(Mean+2 Std. Dev.) Then
          predict this event based on relation
      End If
  Until end of rules
End While
```

**Algorithme 3.3: L'algorithme ALZ de prédiction**

Il faut signaler que cet algorithme peut être aussi utilisé pour la détection d'erreurs. Ceci peut être facilement réalisé en comparant les prédictions effectuées par l'algorithme avec les actions détectées.

Pour valider leur algorithme, il a été appliqué sur des données synthétiques et des données réelles enregistrées après l'observation d'un agent acteur pendant 59 jours. Les résultats obtenus sont affichés au Tableau 3.1.

| Datasets | Percentage accuracy | Percentage error |
|---|---|---|
| Real (without rules) | 55 | 45 |
| Real (with rules) | 56 | 44 |
| Synthetic (without rules) | 64 | 36 |
| Synthetic (with rules) | 69 | 31 |

**Tableau 3.1: Comparaison des résultats avec et sans les règles temporelles**

Le Tableau 3.1 montre une nette amélioration des résultats quand l'algorithme est appliqué sur les données synthétiques. Quant aux données réelles, l'amélioration était moins apparente. Cette dernière observation peut être expliquée par le fait que l'apprentissage automatique des règles temporelles nécessite un plus grand volume de données pour bien résumer les habitudes de l'agent acteur.

### 3.3.2.2.2 Évaluation de l'approche

L'approche de Jakkula et Cook [67] est une très intéressante approche qui se base sur les relations temporelles entre les actions pour faire des prédictions. Néanmoins, elle a quelques limitations. Si le calcul effectué pour détecter les erreurs est assez simple, il suffit de calculer la probabilité des relations entre la dernière action détectée, celle que nous allons juger, et celles détectées avant elle, le calcul effectué pour la prédiction est vraiment lourd surtout pour une application en temps réel. En effet, il faut calculer les probabilités des relations temporelles de toutes les actions, sans exception, avec les actions détectées. Donc, ce que nous pouvons leur reprocher, c'est qu'ils n'utilisent pas de contraintes capables de réduire le nombre d'actions

avant d'effectuer le calcul. Le fait de ne pas utiliser de contraintes provoque un autre problème, avec les mêmes actions détectées, nous aurons toujours la même action qui sera prédite, alors qu'une ou plusieurs actions peuvent appartenir à différentes activités. Le dernier point à soulever, c'est que dans cette approche nous n'avons aucune idée sur l'activité en cours, ce qui nous oblige à refaire le calcul après chaque action détectée, tandis que lorsque nous prédisons une activité et que nous ne détectons aucune erreur, le prochain calcul ne s'effectuera qu'après l'achèvement de cette activité avec toutes les actions qui la composent.

## 3.4. Conclusion

Ce chapitre avait pour but de présenter un état de l'art sur les processus de la reconnaissance d'activités. Avant de définir la reconnaissance d'activité, nous avons commencé par définir les activités que nous allons essayer de reconnaitre, celle de la vie quotidienne (AVQ). Nous avons aussi vu comment le choix des senseurs influence sur le type de l'approche utilisée. L'utilisation de caméras, par exemple, nous oriente vers des approches basées sur la vision. Le travail de Spriggs *et al.*[20] en est un bon exemple. Un travail où la segmentation temporelle est utilisée pour diviser la séquence vidéo enregistrée par la caméra en plusieurs clusters représentant chacun une action. Ces actions sont classifiées afin de nous guider à l'activité entamée lors d'une prochaine détection. Cette approche souffre tout de même de quelques inconvénients. Elle ne réduit pas le nombre d'hypothèses, le traitement est lourd et elle ne préserve pas l'intimité de la personne observée. L'autre travail que nous avons détaillé est celui de Jakkula et Cook [67] qui fait partie des approches basées sur les senseurs. Dans ce travail, les auteurs analysent l'historique de la personne observée pour créer des relations temporelles entre ses activités du genre *activité 1* vient après *activité 2*. La prochaine activité qui

sera effectuée par la personne observée sera donc inférer à partir des activités détectées et de leurs relations avec les autres activités. Les limitations de cette approche sont presque les mêmes que celle d'avant. Elle est très lourde pour une application en temps réel et le nombre d'hypothèses n'est pas réduit.

Au chapitre suivant, nous allons présenter notre nouvelle approche qui essaiera de reconnaitre les activités tout en réduisant le nombre d'hypothèses et garantir un temps de réponse raisonnable pour une application en temps réel.

# Chapitre 4

## Une approche de reconnaissance d'activité utilisant la FDT

### 4.1. Introduction

Dans le Chapitre précédent, nous avons présenté quelques approches qui ont montré l'intérêt de l'utilisation des techniques de la fouille de données temporelles dans le domaine de la reconnaissance d'activité au sein d'un habitat intelligent. Les résultats obtenus étaient très intéressants; mais, malheureusement, aucune de ces approches n'offre une solution complète et pratique pour une assistance automatisée. Le grand inconvénient de ces approches est qu'elles n'ont pas proposé une solution à la grande problématique de la reconnaissance d'activité qui est la réduction du nombre d'hypothèses. En effet, Jakkula et Cook [67] ont évité cette problématique en travaillant directement sur les actions sans construire de plans d'activités. Cette solution a montré ces limitations puisqu'avec les mêmes actions détectées, l'action prédite sera toujours la même, alors qu'une ou plusieurs actions peuvent appartenir à différentes activités d'où la possibilité de diverses actions prochaines. Cette solution est aussi très lourde pour une

application en temps réel parce que les traitements et calculs sont obligatoires après chaque action détectée.

Spriggs et al.[20], Quant à eux, ignorent totalement cette problématique et cherchent l'action détectée dans tous les plans d'activités. Si leurs résultats étaient satisfaisants pour des expériences avec un nombre limité d'activités, il est clair qu'ils ne le seront plus dans un environnement réel, comme le Laboratoire d'Intelligence Ambiante pour la Reconnaissance d'Activité (LIARA) dans lequel nous avons l'intention de tester notre approche prochainement, où le nombre d'activités est normalement beaucoup plus grand.

## 4.2. Le LIARA

Pour expliquer les différent choix des techniques et algorithmes utilisés dans notre approche, il est nécessaire de présenter le type d'environnement où elle sera validée. Le LIARA est une sorte d'habitat intelligent construit sur 100 mètres carrés à l'intérieur de l'université du Québec à Chicoutimi (UQAC). Une centaine de capteurs et effecteurs y sont installés. Parmi les capteurs, nous trouvons des capteurs infrarouges, des tapis de pression, des contacts électromagnétiques, différents capteurs de température, des capteurs de lumière et huit antennes RFID pour capter et transmettre les signaux émis par les étiquettes RFID installées sur les différents objets comme les tasses, les assiettes, etc. Quant aux effecteurs que nous pouvons contrôler, nous y trouvons un iPad Apple, plusieurs IP speackers, une télévision à écran plat haute définition, un home cinéma et plusieurs lumières et LED positionnées dans des endroits stratégiques du laboratoire. La Figure 4.1 montre des images de l'habitat intelligent sous différentes angles.

**Figure 4.1: Images du LIARA**

L'image centrale de la Figure 4.1 montre la cuisine où les activités les plus complexes seront effectuées. Ce sont les activités de préparation de repas qui se composent de plusieurs actions et utilisent différents objets comme la tasse, affichée en bas à gauche, sur laquelle nous avons installé une étiquette RFID. Sur cette image, nous pouvons constater le iPAD installé sur le réfrigérateur qui permet de contrôler les expériences faites dans l'habitat, tester les équipements ou jouer des vidéos dans le but d'assister l'agent acteur dans ces activités complexes. La télévision, en bas à droite de la Figure 4.1 peut, elle aussi, être contrôler à distance pour fournir le même type d'assistance. En haut à gauche, nous pouvons voir le serveur

Dell qui se charge du traitement des informations. Les autres images montrent la salle à manger, les toilettes et la bibliothèque.

En fin, nous devons signaler que tous ces senseurs ont été choisis surtout pour leur facilité d'installation parce que notre but est de transformer l'habitat du patient en un habitat intelligent et non pas de l'amener à vivre dans une sorte de laboratoire où les installations sont très compliquées. Nous devons signaler aussi que ce type d'habitat nous permet le type de reconnaissance que nous avons envisagé, une reconnaissance à l'insu et une reconnaissance d'activités basée sur les objets.

L'approche que nous proposons est une nouvelle approche basée sur la fouille de données temporelles. Elle propose une solution à la réduction des nombres d'hypothèses ainsi qu'à d'autres problèmes variés. Notre approche est divisée en trois étapes qui seront détaillées dans les prochaines sections, l'étape de création des plans d'activités, l'étape de segmentation temporelle des heures de débuts des activités et l'étape qui explique le processus de recherche de l'activité entamée et de prédiction de la prochaine action.

## 4.3. La création des plans d'activités

### 4.3.1. Introduction

La reconnaissance d'activité consiste à sélectionner parmi les plans d'activités celle qui explique le mieux les actions détectées. Il est donc normal que la première étape de cette approche soit la création des plans d'activités. Comme chaque personne peut effectuer la même activité d'une façon différente, il est impossible d'utiliser les mêmes plans d'activités pour

différentes personnes. Prenons par exemple l'activité *préparer café,* son plan d'activité, selon une première personne, peut être composé des actions : *prendre tasse*, *ajouter café* et *ajouter du sucre,* alors que pour une deuxième personne il peut être composé des actions : *prendre tasse, ajouter du lait, ajouter café*. La création des plans d'activités doit donc être une tâche personnalisée pour chaque occupant de l'habitat intelligent. Vu le nombre très élevé des AVQs, l'option de les créer d'une façon supervisée est vite écartée.

La méthode que nous utilisons pour créer les plans d'activités est donc, une méthode non supervisée qui se base sur l'historique des actions du patient observé. Comme les informations de départ ne sont que les données envoyées par les capteurs, une action, pour nous, sera le numéro du capteur qui a connu un changement d'état (ON/OFF), ou celui qui a signalé un grand changement de mesure. Les petits changements sont considérés comme des bruits de signaux et sont donc ignorés.

La première étape de cette création est, comme pour toute approche utilisant la fouille de données temporelles, une étape de transformation. Elle en résultera des séquences ordonnées de senseurs puisqu'elle va comparer les données envoyées par les capteurs à l'instant T avec celles de l'instant T-1, et ne sauvegardera que les numéros des capteurs qui se sont manifestés entre ces deux périodes. L'entrepôt de données qui en résoudra sera similaire au Tableau 4.1 créé d'après la base de données proposée par Kasteren et all.[10].

| Jour | Senseurs |
|---|---|
| 1 | 5 2 1 4 3 2 4 6 8 3 |
| 2 | 6 2 1 4 7 9 2 6 1 6 9 4 |
| 3 | 9 8 3 7 6 1 6 9 7 1 2 1 4 |

**Tableau 4.1: Exemple d'un entrepôt de données**

À titre d'information, dans cette même étape des séquences horodatées de senseurs seront créées où le temps où chaque senseur s'est manifesté sera aussi sauvegardé. Ces séquences horodatées seront utiles dans l'étape de segmentation dans laquelle on commence par récupérer les temps de débuts des différentes activités. Le jour 1, par exemple, sera schématisé comme suit : 5(8h50) 2(9h15) 1(9h16) 4(9h18) 3(9h50) 2(9h55) …

Comme le montre le Tableau 4.1, l'entrepôt de données ressemble beaucoup à celui utilisé dans le domaine de gestion de panier commercial (Tableau 2.5.3.1), ce qui rend la technique du « Sequential pattern mining »[22] la technique la plus adéquate pour la détection des activités. Plusieurs algorithmes ont déjà été proposés pour cette technique, mais comme nous ne devons trouver que les « closed frequent patterns » : si l'activité 2 1 4 est rapportée, ne pas rapporter les sous-séquences tel que 2 1 si elles apparaissent le même nombre de fois que la première séquence, et aussi pour des raisons de performance, nous avons choisi l'algorithme Bide proposé par Wang .all [23] qui sera expliqué dans la prochaine section.

### 4.3.2. Application de l'algorithme BIDE pour la découverte des plans d'activités

Le problème du « Sequential pattern mining » a été présenté par Agrawal et Srikant [70] qui ont généralisé; puis, développé l'algorithme GSP basé sur l'algorithme Apriori [76] expliqué au Chapitre 2. Depuis, plusieurs autres algorithmes ont vu le jour pour essayer d'améliorer les performances, SPAD[73], PrefixSpan[74], SPAM[75], etc. D'autres se sont intéressés plus à trouver les « closed frequent patterns » [71], A-Close [71], Closet [72], CHARM [69], etc. La plupart de ces derniers algorithmes maintiennent une liste des motifs fréquents pour le test de la fermeture, alors que BIDE soulage la mémoire avec sa méthode d'extension bidirectionnelle «BI-Directional Extension».

La première étape de cet algorithme consiste à énumérer toutes les séquences fréquentes en créant un arbre de séquences. L'arbre est créé en affectant ∅ à la racine, puis un nœud N à un niveau L est développé en ajoutant un seul item. Après la suppression de tous les nœuds dont la fréquence est inférieure à la fréquence minimale spécifiée au départ, l'arbre va contenir l'ensemble des séquences fréquentes. La Figure 4.2 schématise l'arbre résultant de l'application de cette étape sur le Tableau 4.2.1 avec une fréquence minimale égale à 3.

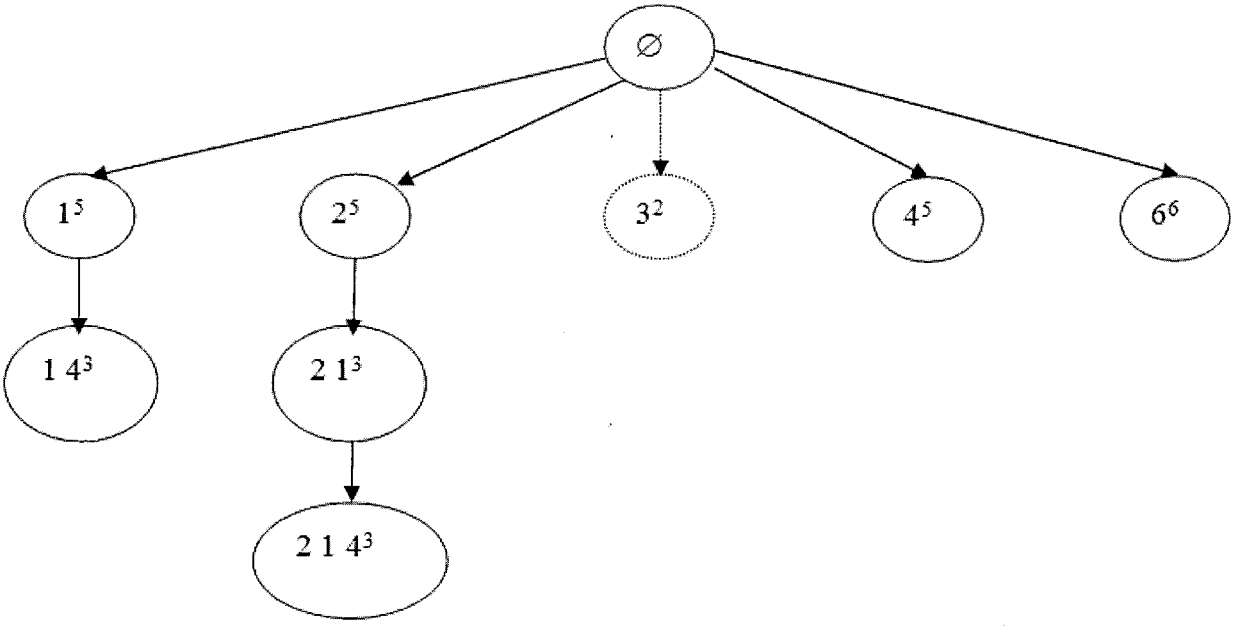



**Figure 4.2: Arbre de séquences**

Comme le montre cette Figure, le nœud 3, en pointiller, ne sera plus développé parce que sa fréquence, qui est égale à 2, est inférieure à la fréquence minimale. La liste des fréquentes séquences qui sera donc construite sera composée de : 1, 2, 4, 6, 1 4, 2 1 et 2 1 4. L'algorithme 4.1 utilisé dans BIDE pour trouver cette liste évite la projection physique de la base de donnée, qui consomme beaucoup trop d'espace mémoire et de temps de traitement, et utilise la pseudo-projection qui est plus efficiente.

**Frequent-sequence-enumeration** (*SDB, min_sup, FS*)
**Input**: an input sequence database *SDB*, a minimum support threshold *min_sup*
**Output**: the complete set of frequent sequences, *FS*
1: $FS = \emptyset$;
2: call *Frequent-sequences(SDB, Ø, min_sup, FS)*;
3: return *FS*;

**Frequent-sequences** ($S_p$_*SDB*, $S_p$, *min_sup*, *FS*)
**Input**: a projected sequence database $S_p$_*SDB*, a prefix sequence $S_p$,
and a minimum support threshold *min_sup*
**Output**: the current set of frequent sequences, *FS*
4: if $S_p$ is non-empty
5: $FS = FS \cup S_p$;
6: $LF\_S_p$ = locally frequent items ($S_p$_*SDB*, $S_p$, *min_sup*);
7: if $LF\_S_p$ is empty
8: Return;
9: for each locally frequent item *i*
10: $S_p^i = \langle S_p, i \rangle$;
11: $S_p^i$_*SDB* = pseudo projected database ($S_p^i$, $S_p$_*SDB*);
12: call *Frequent-sequences*($S_p^i$_*SDB*, $S_p^i$, *min_sup, FS*);

**Algorithme 4.1: Pour trouver la liste des fréquentes séquences**

La deuxième étape de BIDE consiste à sélectionner parmi cette liste de séquences fréquentes les séquences fermées. À la différence des autres algorithmes qui ont besoin de garder en mémoire les séquences susceptibles d'être fermée et les comparer à chaque nouvelle séquence trouvée pour voir si la nouvelle séquence appartient déjà à une des séquences sauvegardées ou si elle l'englobe, BIDE utilise la technique d'extension bidirectionnelle. Cette technique part de l'idée qui stipule que si une séquence $S = e_1, e_2, \ldots, e_n$ est non fermée alors une nouvelle séquence $S'$ doit forcément exister qui a la même fréquence que S et qui est l'extension de S avec un nouvel élément $e'$. Trois cas sont possibles, $e'$ peut être au début : $S' = e', e_1, e_2, \ldots, e_n$, au milieu : $S' = e_1, e_2, \ldots, e_k, e', e_{k+1}, \ldots, e_n$ ou bien à la fin : $S' = e_1, e_2, \ldots, e_n, e'$. Le test de fermeture

revient donc à chercher si le e' n'existe pas dans les trois cas et cela se fait suivant l'algorithme 4.2 présenté ci-dessous.

**BIDE** (*SDB*, *min_sup*, *FCS*)
**Input**: an input sequence database *SDB*, a minimum support threshold *min_sup*
**Output**: the complete set of frequent closed sequences, *FCS*
1: *FCS* = Ø ;
2: *F1*=frequent 1-sequences(*SDB*, *min_sup*);
3: for (each 1-sequence *fl* in *F1*) do
4: $SDB^{fl}$ = pseudo projected database (*SDB*);
5: for (each *fl* in *F1*) do
6: if (!*BackScan*(*fl*, $SDB^{fl}$))
7: *BEI*=backward extension check (*fl*, $SDB^{fl}$);
8: call *bide*($SDB^{fl}$, *fl*, *min_sup*, *BEI*, *FCS*);
9: return *FCS*;

**bide** ($S_p$_*SDB*, $S_p$, *min_sup*, *BEI*, *FCS*)
**Input**: a projected sequence database $S_p$_*SDB*, a prefix sequence $S_p$,
a minimum support threshold *min_sup*, and the number of backward extension items *BEI*
**Output**: the current set of frequent closed sequences, *FCS*
10: *LFI* = locally frequent items ($S_p$_*SDB*);
11: $FEI = \left|\left\{z \text{ in } LFI \mid z.sup = sup^{SDB}(S_p)\right\}\right|$;
12: if ((*BEI*+*FEI*)==0)
13: *FCS*=*FCS* ∪ {$S_p$};
14: for (each *i* in *LFI*) do
15: $S_p^i = \langle S_p, i \rangle$;
16: $SDB^{Spi}$ = pseudo projected database ($S_p$_*SDB*, $S_p^i$);
17: for (each *i* in *LFI*) do
18: if (!*BackScan*($S_p^i$, $SDB^{Spi}$))
19: *BEI*=backward extension check ($S_p^i$, $SDB^{Spi}$);
20: call *bide*($SDB^{Spi}$, $S_p^i$, *min_sup*, *BEI*, *FCS*);

**Algorithme 4.2: Algorithme de test de fermeture**

L'application de l'algorithme 4.2. sur la liste de séquence doit normalement nous donner la liste des séquences fermées qui ne se composera plus que des séquences : 1, 2, 4, 6 et 2 1 4. Il

est à noter que par exemple 1 est une sous séquence de 2 1 4, mais comme sa fréquence, 5, est différente de la deuxième séquence, 3, il doit aussi être rapporté.

### 4.3.3. Validation de la découverte des plans d'activités

Avant de parler des tests et résultats obtenus avec BIDE il faut juste préciser qu'au moment de la validation le LIARA, comme c'est un nouveau laboratoire, ne possédait pas encore une base de données construite par les informations envoyées par les senseurs pour une période assez longue pour permettre de définir les habitudes du patient observé. Il faut aussi préciser que le choix de la base de données proposée par Kasteren et al.[10] est due à la forte ressemblance des deux habitats intelligents comme le montre la Figure 4.3.


**Figure 4.3: L'habitat intelligent du travail de Kasteren et al.[10]**

La Figure 4.3 montre l'habitat intelligent où la base de données, utilisée dans les tests, a été créée après l'observation d'un occupant pendant 28 jours. Dans l'habitat différent senseurs ont été placés, sur les portes, les placards, le réfrigérateur, etc. La position de ces senseurs devra permettre la détection de sept activités :

1: ' *quitter la maison* '
2: '*utiliser les toilettes*'
3: ' *prendre une douche* '
4: '*se coucher*'
5: '*préparer le petit déjeuner*'
6: '*préparer le diner*'
7: '*prendre une boisson*'

Pour tester BIDE, nous avons utilisé une version personnalisée qui permet, à chaque exécution, de spécifier la valeur de quelques paramètres très importants. Le premier indique la fréquence minimale de l'activité selon le nombre de jour total : si cette valeur est mise à 0.9 et que le nombre de jour total est dix, l'activité doit apparaitre au moins une seule fois dans neuf des dix jours. Les autres spécifient le nombre d'erreurs toléré pour une activité et le nombre d'erreurs toléré entre deux senseurs successifs : par exemple si l'activité est 2 1 4 la séquence 2 6 1 4 peut être interprétée comme étant la même activité avec une erreur qui s'est introduite entre le premier et le deuxième senseur.

Les résultats de nos tests ont été un peu différents de ceux attendus. En effet, juste cinq parmi les sept activités ont été découvertes. La première raison de cette différence est que l'occupant avait l'habitude d'effectuer l'activité 1, *quitter la maison*, juste après l'activité 3, *prendre une douche*, ce qui a fait que notre algorithme a reconnu les deux activités comme une seule.

Cette différence n'est pas vraiment grave parce qu'elle permettra d'assister l'occupant dans ses deux activités et pourra prédire la prochaine action, mais l'utilisation d'autres informations temporels, comme la durée moyenne d'une activité ou la durée maximale entre deux senseurs successifs, peut rendre cette assistance encore plus efficace.

Quant à la quatrième activité, *se coucher*, elle n'a pas été détectée parce qu'elle est effectuée aux environs de minuit; ce qui divise généralement ses actions entre deux journées. C'est un problème, non pas juste pour Bide, mais pour tous les algorithmes similaires.

En fin de compte, l'application de Bide a permis de trouver plus que 71% des plans d'activités; mais, il aide à assister l'occupant pour plus que 85% de ses activités.

## 4.4. La segmentation temporelle

À ce stade nous avons déjà effectué deux étapes très importantes. La première consistait en une étape de transformation où les informations significatives envoyées par les senseurs sont organisées sous deux formes, des séquences ordonnées et des séquences horodatées. Si l'utilisation des séquences ordonnées était nécessaire pour l'application de l'algorithme BIDE pour trouver les différents plans d'activités, les séquences horodatées vont permettre de trouver les heures de débuts de ces plans. Cette dernière opération est très simple et se réalise en parcourant les séquences horodatées pour trouver la liste de senseurs constituant un plan d'activité, puis en lui affectant l'heure où son premier senseur s'est manifesté. De cette façon, nous allons bien trouver tous les heures de débuts de chaque plan d'activité, mais ces informations restent inutile pour avoir une idée globale sur les habitudes du patient observé qui aideront à solutionner notre problématique générale qui, rappelons-le, relève de l'obligation du

parcoure du nombre très élevé d'activités à chaque fois nous cherchons celle entamée. En effet, ce que nous cherchons c'est des informations du genre : ce patient effectue d'habitude l'activité *se coucher* le soir, ou encore plus précisément entre 21h00 et 23h00, de cette façon si l'heure courante est 11h00 du matin, cela nous aidera à ignorer cette activité lors de la recherche de l'activité entamée à cette heure-ci.

Pour cette raison, nous avons besoin de transformer toutes les heures de débuts de chaque plan d'activité en intervalles de temps déterminants les périodes de temps où l'occupant a l'habitude de commencer les activités. De cette façon, à chaque moment nous pourrons utilisés l'heure courante, l'heure du système, pour ignorer les activités qu'il n'a pas l'habitude d'effectuer à ce temps-là. Il suffira donc, de choisir l'activité dont un intervalle contient l'heure courante. C'est cette sélection qui nous permet de réduire le nombre de possibilités. En fait, au lieu de chercher l'activité entamée parmi tous les plans d'activités, nous la cherchons juste parmi les activités que l'occupant a l'habitude d'effectuer à cette heure précise de la journée. La Figure 4.4 schématise le résultat de la segmentation de deux activités(x et •) en deux intervalles chacune.

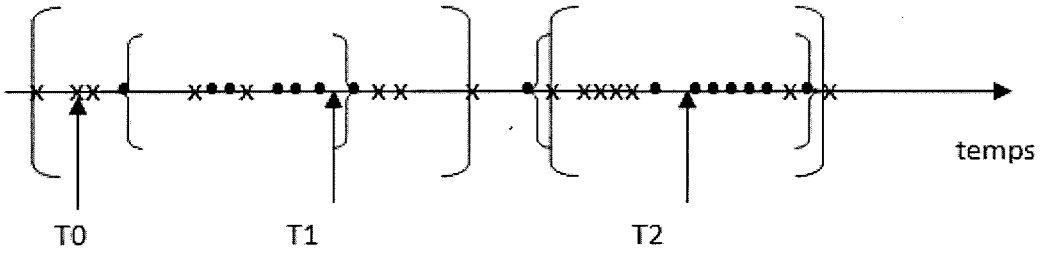



**Figure 4.4: exemple de répartition de deux activités dans le temps**

Dans cette Figure, les intervalles d'heures de débuts de la première activité (x) sont délimités par les grands crochets, alors que les petits crochets délimitent ceux de la deuxième (•).

Cette Figure explique bien le processus de réduction du nombre d'activités au moment de la recherche de celle entamée. Si, par exemple, le temps courant est T0, nous voyons bien qu'aucun intervalle de la deuxième activité ne comporte ce temps-là; ce qui revient à dire que le patient observé n'a pas l'habitude d'effectuer cette activité à ce temps-là et donc cette activité pourra être ignorée lors de la recherche sans influencer le résultat attendu.

### 4.4.1. La création des intervalles temporels

Après avoir parcouru les séquences horodatées, nous obtiendrons pour chaque plan d'activité un ensemble d'heures de débuts. La transformation de ces heures en intervalles temporels peut directement se faire en utilisant un algorithme de la segmentation temporelle, le C-means [47] qui a la particularité d'effectuer la segmentation sans connaitre à l'avance le nombre de clusters, ou dans notre cas d'intervalles temporels, à créer, ce que nous appelons aussi le «Fuzzy clustering». Cet algorithme essaie tout simplement de créer des clusters les plus homogènes possibles, pour nous, les intervalles temporels contiendront des heures de débuts très proches les unes des autres. Par contre, c'est un algorithme qui nécessite beaucoup de temps de calcul surtout si nous le comparons à un autre algorithme comme le K-means [28] qui lui en revanche prend en entrée le nombre de cluster à créer.

Après une mûre réflexion, nous avons pu constater que le nombre de cluster à créer, pour un plan d'activité, n'est rien d'autre que le nombre maximal que cette activité apparait dans une même journée. Cette information est facile à obtenir et peut être calculée en même temps lors du parcours des séquences horodatées pour la récupération des heures de débuts des activités.

Le fait de connaitre le nombre d'intervalles sur lequel l'activité doit être divisée, nous oriente à utiliser l'algorithme du K-means où k va-t-être égale à ce nombre. C'est un algorithme qui commence par choisir au hasard k centres de clusters, par exemple les k premières heures de débuts. Ensuite, sur plusieurs itérations, il calcule pour chaque heure les distances qui la séparent des centres et l'affecte au cluster avec le centre le plus proche; puis, il recalcule le nouveau centre de chaque cluster. Ces itérations s'arrêtent quand les centres ne bougent plus.

La solution avec le K-means semblait être la plus rapide jusqu'à ce qu'on découvre que la prochaine étape, celle de la recherche d'activité, s'effectue plus rapidement si les heures de débuts de chaque intervalle étaient triées. Avec cette remarque, nous avons préféré développer un autre algorithme, plus simple et surtout plus rapide, qui donne les mêmes résultats que le K-means. La différence entre les deux algorithmes est que le k-means crée les intervalles en leur affectant les heures les plus proches de leur centre, tandis que notre algorithme, détaillé à l'Algorithme 4.3, se base sur le fait que les heures sont déjà triées et cherche, k-1 fois, les deux points successives les plus éloignées l'une de l'autre pour créer deux intervalles distincts.

**Entrée:** une table des activités probables ActivProb
**Sortie:** une liste non vide des intervalles de chaque activité dans ActivProb

```
1: Pour chaque activité Activ dans  ActivProb
2:        i=1
3:        Tant que i < Activ.NbrIntvl
4:                Pour chaque Time dans Activ.TimeBeg  et Time ∉ Table
5:                        M = Max (NextTime - Time)
6:                Ajouter IndiceM dans Table
7:                ajouter1 à i
8:        Trier_Asc (Table)
9:        Ajouter 0 à ActivProb.Intvl
10:       Pour chaque T in Table
11:               Ajouter T à ActivProb.Intvl
12:               Ajouter T+1 à ActivProb.Intvl
13:       Ajouter Activ.TimeBeg.size -1 à ActivProb.Intvl
```

**Algorithme 4.3: création des intervalles temporels**

L'algorithme 4.3 commence par parcourir le tableau ActivProb qui comporte tous les plans d'activités. Chaque plan Activ est composée du nombre d'intervalles à créer NbrIntvl, d'un tableau trié d'heures de débuts TimeBeg, et d'un tableau vide Intvl qui comportera les indices des heures de débuts et de fins de chaque intervalle dans TimeBeg. Pour chaque plan, il itère NbrIntvl-1 fois pour trouver les T indices dans TimeBeg ($t_1$, $t_2$, …, $t_{NbrIntvl-1}$) qui divisent les intervalles. Ensuite, les intervalles sont créés en commençant évidement par l'intervalle dont l'heure de début est le premier élément du tableau TimeBeg d'où l'indice 0 et l'heure de fin est celle pointée par l'indice $t_1$. Le deuxième intervalle sera du premier indice après $t_1$ qui est le $t_1$+1 jusqu'à $t_2$. Ainsi de suite jusqu'au dernier intervalle qui sera de $t_{NbrIntvl}$ jusqu'au dernier élément de TimeBeg qui a l'indice égal au nombre d'éléments de TimeBeg -1 (puisque le premier indice est 0).

Une fois les intervalles créés, une dernière étape de vérification est indispensable pour améliorer le résultat de la segmentation utilisée. En effet, le patient peut, parfois, effectué une

activité à un horaire différent de celui où il a l'habitude de l'effectuer. Comme nous prenons en compte toutes les heures de débuts des activités et que le nombre d'intervalles créés est égal au nombre maximal qu'apparait une activité dans une même journée, des intervalles peuvent être créés alors qu'ils contiennent un nombre d'activités très inférieur à la fréquence minimale. Ces intervalles ne traduisent pas une habitude du patient; mais plutôt, des exceptions qui ne doivent pas être prises en considération. L'étape de vérification servira donc, dans ce cas, à supprimer ce genre d'intervalles. Elle servira aussi dans d'autres cas à diviser en deux un intervalle donné. Prenons par exemple le cas où le patient effectue toujours une activité une seule fois par jour, mais parfois le matin et d'autres le soir. Nous pouvons penser par exemple à l'activité *prendre une douche*. Dans ce cas, un seul intervalle sera créé et débutera le matin et ne finira que le soir. Ce type d'intervalle ne reflète pas une idée précise sur l'habitude du patient puisqu'il comporte une longue période où le patient n'a pas l'habitude d'effectuer cette activité. En plus, notre principal objectif est de réduire le nombre d'activités, alors que des intervalles de ce type ne vont pas être ignorés dans les périodes où l'activité est peu probable. L'algorithme utilisé dans cette étape de vérification est détaillé ci-dessous.

```
Entrée: ActivProb.Intvl
Sortie: ActivProb.Intvl

1: Pour chaque Activ dans ActivProb
2:       Pour chaque Intvl dans Activ.Intvl
3:             Si Intvl.NbrTime < (fréquence/ Activ.NbrIntvl)
4:                   Supprimer (Intvl)
5:                   Soustraire 1 de Activ.NbrIntvl
6:             Sinon
7:                   Pour chaque Time dans Intvl
8:                         si (NextTime – Time) > MaxDist
9:                         Ajouter 1 à Activ.NbrIntvl
10:                        End = Intvl.end
11:                        Intvl.end = Time
12:                        Insérer NewIntvl
13:                        NewIntvl.beg = Time +1
14:                        NewIntvl.end = End
```

**Algorithme 4.4: L'algorithme de l'étape de vérification**

Le principe de cet algorithme est vraiment simple. Il parcourt tous les intervalles de tous les plans d'activités et test en premier lieu si l'activité n'est pas fréquente dans cet intervalle et le supprime dans ce cas. Si elle est fréquente, il teste si deux heures successives sont trop éloignés et dans ce cas il divise l'intervalle en deux.

### 4.4.2. Validation de la segmentation temporelle

À n'importe quel moment, pour trouver l'activité entamée, nous somme obligé normalement de parcourir tous les plans d'activités. C'est comme si tous les plans s'étalent sur toute la journée. L'application des deux algorithmes, le 4.3 et le 4.4, a permis, comme le montre la Figure 4.5, de préciser des plus courtes périodes où le patient a l'habitude de les effectuer.

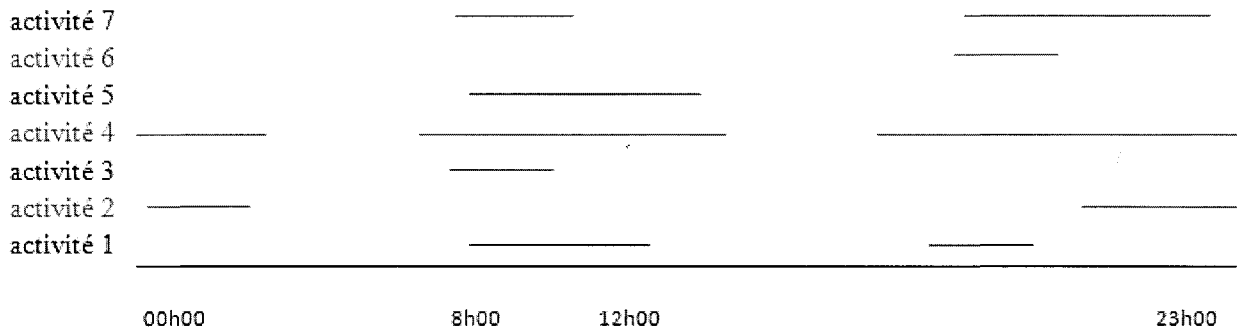



**Figure 4.5: representation des intervalles des heures de début des activités**

Dans la Figure 4.5, les traits devant les activités représentent les périodes où le patient les effectue d'habitude. Nous pouvons constater que la réduction du nombre des possibilités varie de 30 à 70% selon la période. Il est à noter, quand même, que le nombre d'activité, qu'une personne effectue normalement par jour, est nettement supérieur aux sept activités qui ont été utilisé lors de ces expérimentations. De cette constatation, nous pouvons nous attendre à ce que le pourcentage de réduction des possibilités augmenterait forcément dans un cas réel.

Toutefois, il faut signaler un grand problème rencontré lors de cette segmentation. Le fait de considérer une journée comme une ligne de temps fait en sorte que deux heures très proches paraissent très éloignées. Par exemple, 23h59 et 00h01 sont considérés comme éloigné de presque 24 heures et non pas de deux minutes. Si nous devons créer un intervalle il sera alors de 00h01 jusqu'à 23h59 parce que l'heure de début est toujours inférieur à l'heure de fin, alors que l'intervalle recherché devait être de 23h59 jusqu'à 00h01. La différence entre les deux est énorme et affectera grandement notre segmentation. C'est pour cela que la dernière étape de la segmentation, celle de la vérification, est très importante car elle va permettre de diviser l'intervalle qui prenait toute la journée en deux petits intervalles. Un résultat très proche de celui recherché.

## 4.5. La recherche d'activité

Un système basique de recherche d'activité consisterait à chercher, parmi tous les plans d'activités, celui ou ceux qui explique le mieux les actions observées. L'étape précédente, celle de la segmentation temporelle, améliore grandement ce système en limitant le nombre de plans à parcourir. Par contre, comme la personne observée souffre de la maladie d'Alzheimer, elle risque de commettre des erreurs d'initiations. Des erreurs auxquelles nous voulions proposer une solution efficace vu leur importance et leurs caractéristiques particulières. En effet, on peut imaginer un patient qui n'arrive pas à entamer l'activité *prendre médicament* et comment un système de recherche d'activité peut l'assister s'il ne détecte aucune action du patient. Plus précisément, ce problème est un problème d'équiprobabilité. En effet, tant qu'aucune action n'est détectée, toutes les activités que le patient a l'habitude d'effectuer au moment de cette recherche ont la même probabilité d'être entamée. Le système que nous proposons dans la prochaine section profite de l'étape de la segmentation temporelle pour réduire le nombre de possibilités et répond avec une méthode simple et efficace aux erreurs d'initiations.

### 4.5.1. Le système de recherche d'activité

Dans l'étape de la segmentation temporelle, les intervalles de tous les plans d'activité sont créés. Notre système de recherche d'activité profite de cette étape et utilise l'heure courante, l'heure système, pour réduire le nombre de possibilités. Parcourir tous les plans d'activités permet de garder un plan, si un de ses intervalles temporels comporte l'heure courante, ou de l'ignorer, si aucun des intervalles du plan ne la comporte. C'est une opération facile qui évite, à chaque détection d'une action, de la chercher dans tous les plans d'activités.

Après cette première étape, certes le nombre des possibilités est réduit, mais toutes les activités conservées ont la même probabilité d'être entamées par le patient observé. Pour trouver l'activité la plus probable, celle qui sera suggérée par le système au patient au cas où ce dernier commet une erreur d'initiation, notre système calcule, pour toutes les activités conservées, une priorité initiale. L'activité qui sera suggérée par le système sera donc celle avec la priorité initiale la plus élevée.

La priorité initiale d'une activité est calculée en se basant sur les deux constats suivants :

- Plus l'heure courante est proche de la fin d'un intervalle d'une activité, plus la priorité de cette activité est élevée. Si nous revenons à la Figure 4.4 et que nous supposons que l'heure courante est T1, alors l'activité 2 (•) est plus prioritaire que la première puisque T1 est plus proche de la fin de l'intervalle de l'activité 2, symbolisé avec le petit crochet, que celui de l'activité 1.

  La relation que nous utilisons pour ce calcul est :

$$P_{il} = 1 - ((T_f - T_c) / D)$$

  Où $T_f$ et $T_c$ sont respectivement l'heure de fin de l'intervalle et l'heure courante, et D est la durée de l'intervalle : $D = (T_f - T_d)$, sachant que $T_d$ est l'heure de fin de l'intervalle.

- Plus le pourcentage d'apparition d'une activité avant l'heure courante au sein d'un intervalle est grand, plus la priorité de cette activité est élevée. En d'autres termes, le patient a plus l'habitude d'effectuer cette activité avant l'heure courante qu'après. Toujours dans la Figure 4.4, Si $T_c$ = T2, alors l'activité 1 sera la plus prioritaire; puisque, avant T2 l'activité 1

apparait 5 fois sur un total de 7, soit un pourcentage de 71%, tandis que l'activité 2 apparait 2 fois sur un total de 5, soit un pourcentage de 40%.

La relation que nous utilisons pour ce calcul est :

$$P_{i2} = N_{ac} / N_t$$

Où $N_{ac}$ et $N_t$ sont le nombre avant l'heure courante et le nombre total de l'activité dans l'intervalle.

En fin la probabilité initiale de l'activité est :

$$P = P_{i1} + P_{i2}$$

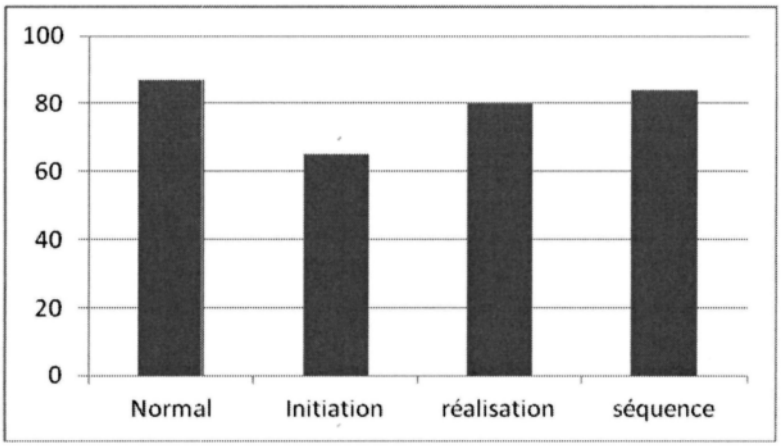
Cette priorité ainsi calculée sert donc à classifier les activités selon la plus probable à être entamée. Elle se met à jour en tenant compte des actions détectées. C'est exactement ce qui se produit dans la troisième étape de notre système de recherche d'activité qui se base sur cette priorité initiale pour favoriser les activités qui contiennent les actions observées. D'ailleurs, il est tout à fait logique de penser que le patient est en train de se faire du thé plutôt que du café si l'action *prendre thé* est observée.

La troisième étape sert donc à prendre en considération les actions du patient et consiste à mettre à jour les priorités initiales en leur ajoutant un bonus de 0.5 si les actions qui composent leur plan d'activité comportent l'action observée, et un autre bonus de 0.5 si l'action observée est dans le bon emplacement dans le plan d'activité. Par exemple, si nous possédons trois plans d'activité, le premier est composé des senseurs 2 1 4, le deuxième de 4 2 5 7 et le troisième de 3 7 2 6, et que le senseur qui vient de se manifester est le senseur 4, à la priorité initiale du premier plan s'ajoutera un bonus de 0.5, 1 au deuxième et aucun bonus ne sera ajouté à la troisième.

### 4.5.2. Validation du système de recherche d'activité

Pour que les tests d'une application basée sur la fouille de données temporelles soient signifiants, il ne faut pas que les données utilisées pour les tests aient déjà servi dans le processus d'extraction de la connaissance. En effet, il est clair que les résultats des tests seront biaisés si des données qui ont servi de base à l'opération d'extraction du savoir, servent aussi à tester ce savoir. Plusieurs techniques existent pour diviser l'entrepôt de données dès le départ en données d'entrainement et tests. Pour les tests du système de recherche d'activité, nous avons choisi la technique du 90/10. En effet, nous n'avons utilisé que 25 jours dans toutes les précédentes étapes et nous avons consacré les trois derniers jours aux tests.

Aussi, pour voir comment notre système répondra aux différentes erreurs commises par un patient souffrant de la maladie d'Alzheimer, nous avons introduits quelques erreurs dans les données utilisées pour les tests. Le Graphe 4.1 représente les résultats obtenus.



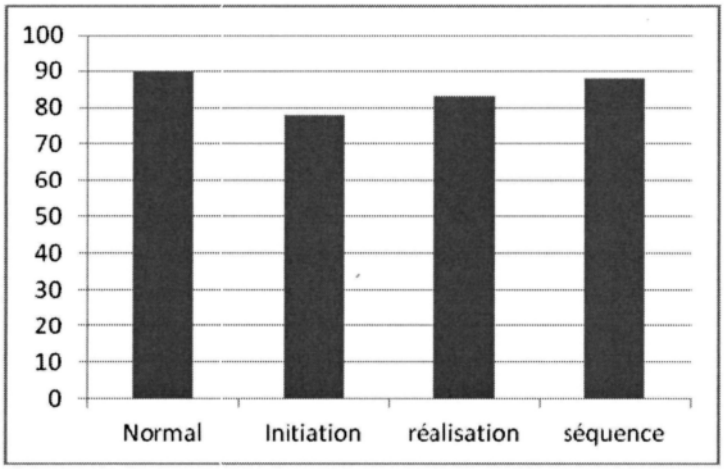

**Graphe 4.1: Les résultats des tests de notre système**

Le graphe 4.1 montre le pourcentage des activités qui ont été bien détectées par notre système selon les actions observées. Les différentes colonnes du Graphe représentent les différentes erreurs introduites :

- Normal : Les résultats ont été enregistrés après avoir donné au système les même actions des trois derniers jours d'observation du patient.
- Initiation : Aucune action n'est introduite au système. Les résultats ont été enregistrés en se basant juste sur les probabilités initiales.
- Réalisation : Les actions introduites au système comporte des actions ajoutées ou supprimées pour simuler des erreurs de réalisation
- Exécution : Les actions introduites au système ne sont pas dans le bon ordre afin de simuler des erreurs d'exécution.

En général, les résultats étaient très satisfaisants. Pour la première colonne, les activités qui n'ont pas été détectées sont celles qui ont été commencées dans une période où l'occupant n'avait pas l'habitude de les commencer. Nous sommes certains que le nombre de ces activités diminuerait encore plus si nous observons le patient pour une période plus longue. Pour la troisième et la quatrième colonne, notre système a prouvé qu'il répond bien à ce genre d'erreurs. La diminution remarquée par apport à la première colonne s'explique par la ressemblance de certaines activités dans les périodes de commencement ainsi que dans les actions qui les composent. En ce qui concerne la deuxième colonne, le relativement faible pourcentage de reconnaissance d'activité est dû principalement aux périodes où la segmentation n'a pu diminuer le nombre de possibilités qu'à 30%. Les résultats de cette deuxième colonne se sont particulièrement améliorés quand nous supprimons l'intervalle d'une activité qui vient de se

dérouler pour qu'elle ne soit pas considérer encore une fois dans la même période. Le Graphe 4.2 présente les nouveaux résultats ainsi obtenus.




**Graphe 4.2: Les résultats finaux des tests du système**

## 4.6. Conclusion

Ce chapitre avait pour but de présenter une approche pour la reconnaissance d'activité utilisant la fouille de données temporelles. Notre approche se compose de trois étapes majeures qui ont été expliquées, implémentées et validées chacune à part. La première est l'étape de création des plans d'activités où l'historique des actions de la personne observée est analysé pour trouver des motifs qui se répètent dans le temps et qui constitueront nos plans. La technique du « Sequential pattern mining » a été utilisée dans cette étape. Elle a été testée avec les données publiées par Kasteren *et al.*[10] et a montré des résultats satisfaisantes.

La deuxième étape commençait par attribuer aux plans d'activités, créés dans l'étape précédente, leurs heures de débuts. Ensuite, elle utilise la technique de la segmentation temporelle pour transformer ces heures en intervalles temporels. Au lieu d'utiliser l'algorithme

k-means pour cette segmentation, nous avons développé un algorithme plus rapide et plus adapté à notre cas d'étude. Les résultats obtenus après les tests montrent la répartition des intervalles dans le temps et comment cette segmentation permet une nette réduction du nombre d'hypothèses au moment de la recherche.

Quant à la troisième et dernière étape, Elle commence par ignorer tous les plans d'activités qui ne possèdent pas un intervalle qui comporte l'heure courante. Ensuite, elle compare les actions détectées à ceux des plans restants pour trouver l'activité la plus probable que le patient a entamée ou celle qui devrait entamer.

# Chapitre 5

## Conclusion générale

Le domaine de la reconnaissance d'activités est un domaine très vaste et a fait l'objet de plusieurs recherches. Différentes approches ont était proposées; mais ce qui se dégage de ce travail est que les techniques de la fouille de données temporelles sont très bien adaptées aux types de problèmes rencontrés dans ce domaine. La fouille de données temporelles permet d'abord de travailler facilement sur des entrepôts de données contenant un très grand nombre d'informations et différents types de données. Elle nous a également permis de créer, d'une façon non supervisée, des plans d'activités propres à la personne observée, sans parler de la segmentation temporelle et de ses autres opérations qui offrent des solutions intelligentes à des problèmes très compliqués. Le deuxième point qui se dégage de ce travail est l'importance de l'aspect temporel dans le processus de la reconnaissance d'activités. La plupart des recherches se basent essentiellement sur les actions détectées pour reconnaitre les activités; mais, il est à noter

que celles qui exploitent l'aspect temporel, spatial ou autres contraintes, affichent de meilleurs résultats. Nous l'avons bien constaté durant ce travail quand le nombre d'hypothèses a été nettement réduit en utilisant le temps de début des activités.

Notre approche qui exploite l'aspect temporel et se base sur les techniques de la fouille de données temporelles a bien répondu aux objectifs fixés même si elle souffre de quelques limitations. D'une vue générale, Notre approche reste parfaitement applicable dans un habitât intelligent, par contre des améliorations s'imposent pour garder un bon pourcentage de réussite des activités reconnues.

## 5.1. Réalisation des objectifs fixés

L'objectif ultime de notre travail est d'élaborer une approche qui se base sur la fouille de données temporelles et qui permet la reconnaissance instantanée de l'activité afin d'assister, en cas de besoin, la personne observée souffrante de la maladie d'Alzheimer dans ses activités de vie quotidiennes. Pour réaliser cet objectif, nous nous sommes imposé d'autres objectifs intermédiaires cités à l'introduction.

Premièrement, nous avons bien étudié la maladie d'Alzheimer en mettant l'accent sur les différentes erreurs susceptibles d'être commises par la personne observée [4, 5, 6, 7]. Notre but était de comprendre cette maladie et d'anticiper ou préparer des solutions à ces erreurs. C'est de cette façon que nous avons pu traiter les erreurs d'initiations qui, à notre connaissance, n'ont jamais été traité auparavant par aucune autre recherche et qui se traduisent par l'incapacité du patient à amorcer les activités.

Le deuxième chapitre a été entièrement dédié à notre second objectif qui proposait une étude détaillé sur la fouille de données temporelles et ses techniques que nous avions l'intention d'utiliser dans notre approche. Cette investigation nous a permis de mieux se rendre compte de la puissance de ces techniques et d'en choisir deux parfaitement adaptée aux problèmes rencontrés [12, 13, 14]. Le « sequential pattern mining » pour créer les plans d'activités et la segmentation temporelle pour diviser en intervalles temporelles les heures de débuts des activités.

Avant d'élaborer notre approche, notre troisième objectif consistait à faire une investigation sur les approches existantes afin de profiter de leurs avantages et éviter leurs faiblesses. Le troisième chapitre a présenté différents travaux et s'est focalisé sur ceux utilisant, comme nous, la fouille de données temporelles [18, 20]. Cette investigation nous a permis d'évaluer différentes pistes et de choisir parmi elles celles qui semblaient les plus pertinentes.

À la lumière des trois précédentes étapes, nous avons enfin pu réaliser notre objectif principal, celui de la création d'une approche qui utilise la fouille de données temporelles pour reconnaitre les activités d'un patient de la maladie d'Alzheimer dans le but de l'assister en cas de besoin. Cette approche permet bien de répondre à notre problématique de recherche et réduit considérablement le nombre d'hypothèses au moment de la recherche de l'activité entamée par le patient. Elle répond aussi aux erreurs d'initiations qui sont des erreurs spéciales parce qu'il faut reconnaitre l'activité qui sera entamée sans avoir détecté aucune action. Le chapitre quatre détail cette approche comme il détail les tests, avec des données réelles, effectuées pour sa validation. Des tests qui ont permis de montrer l'efficacité de cette approche comme ils ont montré ces limitations.

Sommairement, nous avons réalisé tous les objectifs fixés à l'introduction. L'approche créée répond aux différents problèmes rencontrés; mais, des améliorations s'imposent pour prendre en considération d'autres problèmes et garder un bon taux des activités reconnues.

## 5.2. Apport de ce mémoire

Notre contribution dans le domaine de la reconnaissance des activités consiste en l'élaboration d'une toute nouvelle approche qui utilise la fouille de données temporelles. Cette approche propose une solution pour notre problématique de recherche et permet ainsi de réduire considérablement le nombre d'hypothèses. Elle permet d'abord de créer d'une façon non supervisée les plans d'activités propres à chaque personne observée. Par une analyse de l'historique des informations envoyées par les senseurs, des informations qui représentent les actions effectuées par la personne observée, les plans sont créés en découvrant les plus longues suites des actions qui se répètent dans le temps.

L'utilisation de la segmentation temporelle pour transformer l'ensemble des heures de débuts de chaque activité en un ou plusieurs intervalles temporels est la phase la plus importante dans cette approche parce que ces intervalles permettent, si aucun d'eux ne contient l'heure courante d'ignorer au moment de la recherche leur activité. C'est ainsi que le nombre d'hypothèses est réduit. De plus, nous n'avons pas utilisé l'un des algorithmes existants de la segmentation temporelle; mais, nous avons développé un nouvel algorithme plus rapide et adapté à notre cas d'étude.

Nous avons également proposé une solution aux problèmes d'initiations que les patients de la maladie d'Alzheimer sont susceptible de commettre. Nous tenons à signaler, qu'à notre connaissance, aucune autre recherche n'avait proposé une solution à ces erreurs.

## 5.3. Limitations et développements futurs

Certes notre approche répond aux différents objectifs fixés à l'avance et permet de réduire le nombre de possibilités; mais, elle souffre tout de même de quelques limitations que nous avons bien pu remarquer après la phase de tests. Lors de la création des plans d'activités nous avons vu comment les activités qui se produisent aux environs de minuit sont presque impossible à détecter car leurs actions sont partagées entre deux journées. Nous avons vu aussi que les intervalles temporels de ces mêmes activités prennent souvent toute la journée avant d'être améliorer par un autre algorithme de vérification. Nous tenons à noter que ces deux inconvénients découlent du fait que nous représentons une journée par une ligne de temps qui commence à 00h00 et se termine à 23h59. Pour un prochain travail, nous n'avons pas encore une solution exacte mais nous pensons que la solution doit représenter une journée par un cercle et le calcul des distances entre deux points se ferait à l'aide d'un modulo 24 par exemple.

Une deuxième limitation de cette approche consiste à son incapacité de définir le moment exacte pour proposer de l'aide; puisque, les plans d'activités se composent d'une liste ordonnée d'actions mais ne comporte aucune information sur la durée de l'action ou de la durée maximale entre deux actions. Si le patient n'arrive plus à terminer une activité, notre approche va-t-être capable de trouver la prochaine action qui devrai être effectuée; mais, ne la proposerai jamais au patient. Pour résoudre ce problème, il faudra que les plans d'activités contiennent les actions,

leurs durées ainsi que la durée maximale entre deux actions successives. Ces informations peuvent aussi aider à préciser si une action détectée est erronée, continue l'activité entamée ou en commence une autre.

La limitation qui semble la plus grave est celle causée par un changement dans les habitudes de la personne observée. Comme notre approche se base sur ses habitudes pour créer les intervalles des heures de débuts, il est clair qu'elle perd tout son intérêt si le patient change d'habitude et décide d'effectuer une activité à une heure qui n'appartient à aucun de ses intervalles. Dans ce cas, même si l'utilisation de notre approche n'accélérerai pas le temps de la recherche, mais elle ne le ralentira pas non plus. La recherche se fera d'abord parmi les activités que le patient a l'habitude d'effectuer à cette heure-là, puis parmi le reste des activités. Sinon, ce travail peut encore être amélioré, servir de base théorique d'autres travaux ou d'être intégré à une autre approche pour une amélioration des résultats.

## 5.4. Bilan personnel sur ce travail de recherche

Comme conclusion finale, j'aimerai bien parler de cette belle expérience d'initiation à la recherche. Certes mon investissement dans ce travail fastidieux m'a couté beaucoup d'efforts et de temps; mais, l'expérience et les connaissances acquises en valaient bien la peine. La participation à ce projet m'a permis de me familiariser au domaine de la reconnaissance d'activités ainsi qu'au domaine émergeant de la fouille de données. Elle m'a permis également d'assister à la naissance du LIARA et faire partie de sa formidable équipe. Elle m'a permis aussi de développer mes habiletés communicationnelles et de contribuer modestement dans mon domaine de recherche en présentant mon travail au 79$^{ème}$ congrès de l'ACFAS à Sherbrooke, et

à la conférence internationale SMART 2012 à Stuttgart en Allemagne. Comme cette première expérience s'est très bien déroulée, elle m'a encouragé à aller de l'avant et à entamer des études doctorales en lien avec mon domaine d'expertise.

# Bibliographie


[1] : Rising Tide: The impact of dementia on Canadian Society.
Alzheimer Society Canada

[2] : Weber , W., Rabaey , J. M., and Aarts, E. Ambient Intelligence. Springer, 2005

[3] : BOUCHARD, B. Un modèle de reconnaissance de plans pour les personnes atteintes de la maladie d'Alzheimer basé sur la théorie des treillis et sur un modèle d'action en logique de description. PhD thesis, Université de Sherbrooke, 2006

[4] : Norris, S. La Maladie d'Alzheimer. Direction de la recherche parlementaire, 2002.

[5] : Guathier, S. Advances in thepharmacotherapy of Alzheimer's disease. 2002, pp.617.

[6] : Baum, C., Edwards, D. F. Cognitive performance in senile dementia of the Alzheimer's type : The kitchen task assessment. American Journal of Occupational Therapy. May 1993 vol. 47 no. 5 431-436.

[7] : Baum, C., Edwards, D. F., And Morrow-Howell, N. Identification and measurement of productive behaviors in senile dementia of the Alzheimer type. The Gerontologist (1993) 33 (3):403-408.

[8] : Kautz, H. A formal theory of plan recognition. In PhD thesis, University of Rochester, 1987.

[9] : Wobke, W. Two Logical Theories of Plan Recognition. Journal of Logic Computation, Vol. 12 (3), (2002), 371-412

[10] : Kasteren, T., Noulas, A., Englebienne, G., and Ben, K. Accurate Activity Recognition in a Home Setting. In UbiComp, (2008) 21-24.

[11] : Chen, Y., Lu, C., Hsu, K., Fu, L., Yeh, Y., and Kuo, L. Preference Model Assisted Activity Recognition Learning in a Smart Home Environment. In IEEE/RSJ International Conference on Intelligent Robots and Systems October, (2009) 4657-4662.

[12] : Benoît, Gerald. Data Mining [Chapter 6, pps 265-310). In Cronin, B. (Ed.). 2002.

[13] : Chittaro, L., Carlo Combi. Giampaolo Trapasso. Data Mining on Temporal Data: a Visual Approch and its Clinical Application to Hemodialysis. Journal of Visual languages and Computing, vol. 14, no. 6, December 2003, pp. 591-620.

[14] : Antunes, C. M., and Oliveira, A. L. Temporal Data Mining: an overview. KDD Workshop on Temporal Data Mining, 2001.

[15] : Schmidt, C. F. , Sridharan, N. S., And Goodson, J. L. The plan recognition problem : An intersection of psychology and artificial intelligence. Artificial Intelligence. (1978) 45-83.

[16] : Inomata, T., Naya, F., Kuwahara, N., Hattori, F., and Kogure, K. Activity Recognition from Interactions with Objects using Dynamic Bayesian Network. Inomata, 2009.

[17] : Ghazvininejad, M., Rabiee, H. R., Pourdamghani, N., and Khanipour, P. HMM Based Semi-Supervised Learning for Activity Recognition. ACM, 2011.

[18] : Jakkula, V. R., and Cook, D. J. Learning Temporal Relations in Smart Home Data.

[19] : James Allen. Maintaining knowledge about temporal intervals. Communications of the ACM 26(11).832-843, 1983.

[20] : Spriggs, E. H., De la Torre, F., and Heber, M. Temporal Segmentation and Activity Classification from First-person Sensing. Computer Vision and Pattern Recognition Workshops, 2009. CVPR Workshops 2009. IEEE Computer Society Conference.

[21] : Harvey, S., Parvathi, C., and Subramaniam, M. Summarizing Developer Work History Using Time Series Segmentation. Proceedings of the 2008 international working conference on Mining software repositories, Pages 137-140.

[22] : Mabroukeh, N. R., and Ezeife, C. I. A taxonomy of sequential pattern mining algorithms. ACM Computing Surveys (CSUR), Volume 43 Issue 1, November 2010. Article No. 3

[23] : Wang, J., and Han, J. BIDE: Efficient mining of frequent closed sequences. In: Proceeding of the 2004 international conference on data engineering (ICDE'04), Boston, MA, pp 79–90.

[24] : Patterson, D. J., Liao, L., Fox, D., and Kautz, H. Inferring High-LevelBehavior from Low-Level Sensors. In Proceeding of UBICOMP 2003 : The 5th International Conference on Ubiquitous Computing, Anind Dey, Albrecht Schmidt, et Joseph F. McCarthy, editors, volume LNCS 2864, 73n89. Springer-Verlag, 2003.

[25] : Fayyad, U.M, Piatetsky-Shapiro, G., and Smyth, P. From data mining to knowledge discovery in databases. AI Magazine, 17(3), pp. 37-54. 1996.

[26] : Dunham, M.H. Data mining introductory and advanced topics. Upper Saddle River, NJ: Pearson Education, Inc. 2003.

[27] : Buchanan, B.G. History of Artificial Intelligence. AI Magazine 26(4): 53-60 (2005). 60, Electronic Edition.

[28] : Xiong, H., Wu, J., and Chen, J. K-means Clustering versus Validation Measures: A Data Distribution Perspective. Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions. April 2009.

[29] : Two Crows. About Data Mining [Third Edition]. Data Mining '99: Technology Report, Two Crows Corporation, 1999.

[30] : Data Mining: Federal Efforts Cover a Wide Range of Uses (GAO-04-548). General Accounting Office. 2004, May 4.

[31] : Schneier, B. Why data mining won't stop terror. Wired.com, 9 Mars 2006.

[32] : Frawley, W.J., Piatetsky-Shapiro, G., and Matheus, C.J. Knowledge Discovery in Databases: An Overview. In G. Piatetsky-Shapiro and W. J. Frawley, editors, Knowledge Discovery in Databases. AAAI / MIT Press, 1991.

[33] : Witten, I. H., Eibe, F., and Hall, M. A. Data Mining Practical Machine Learning Tools and Techniques. Third Edition (The Morgan Kaufmann Series in Data Management Systems).

[34] : Han, J., and Kamber, M. Data mining: concepts and techniques (Morgan-Kaufman Series of Data Management Systems).(2001). San Diego: Academic Press.

[35] : Valenzuela, C. L., and Jones, A. J. Evolutionary divide and conquer (i): A novel genetic approach to the tsp. Evolutionary Computation 1(4):313-333. 1993.

[36] : Weiss, S.M., Indurkha, N., Zhang, T., and Damerau, F.J. Text mining: Predictive methods for analyzing unstructured information. Springer; 1 edition (Oct 25 2004).

[37] : Freitas, A.A. On rule interestingness measures. Knowledge-Based Systems 12 (1999) 309–315.

[38] : Galushka, M., Patterson, D., and Rooney, N. Temporal Data Mining for Smart Homes. In: Augusto JC, Nugent CD (eds) Designing smart homes. The role of artificial intelligence. Springer, Berlin, pp 85–108.

[39] : Agrawal R., K.-I.Lin, Sawhney H., and K.Shim K. Fast similarity search in the presence of noise, scaling, and translation in time-series databases. VLDB '95 Proceedings of the 21th International Conference on Very Large Data Bases Pages 490 – 501.

[40] : Illa, J., Alonso, J., and Marre, S. Nearest-Neighbors for time series. Kluwer Academic Publisher, Applied Intelligence, USA, (20), (2004) 21-35.

[41] : Keogh, E., and Smyth, P. A probabilistic approach to fast pattern matching in time series databases. In Proceedings of the 3rd International Conference of Knowledge Discovery and Data Mining, pp. 20-24, 1997.

[42] : Rafiei, D., and Mendelzon, A. Efficient Retrieval of Similar Time Sequences Using DFT. In Proc. FODO Conference, Kobe (1998), 249-257.

[43] : Mörchen, F. Time series feature extraction for data mining using DWT and DFT. Technical Report 3, 2003.

[44] : Holmes, J. H., Durbin, D. R., and Winston, F. K. A New Bootstrapping Method to Improve Classification Performance in Learning Classifier Systems. PPSN VI Proceedings of the 6th International Conference on Parallel Problem Solving from Nature (2000) 745 – 754.

[45] : Hand, D., Mannila, H., and Smyth, P. Principles of Data Mining. ABradford Book, The MIT Press, 2001.

[46] : MacQueen, J. Some methods for classification and analysis of multivariate observations. In: Proc. of the fifth Berkeley Symposium on Mathematical Statistics and Probability, Vol. 1University of California Press (1967) , p. 281-297.

[47] : Bezdek, J. C., Ehrlich, R., and Full, W. The fuzzy C-means clustering algorithm. Computers & Geosciences Vol. 10, No. 2-3, pp. 191-203, 1984.

[48] : Zhao, Q., and Bhowmick, S. S. Association Rule Mining:A Survey. Technical Report, CAIS, Nanyang Technological University, Singapore , 2003, No. 2003116.

[49] : Goethals, B. Survey on Frequent Pattern Mining. Technical Report: Helsinki Institute for Information Technology, 2003.

[50] : Roy, P. C., Bouchard, B., Bouzouane, A., and Giroux, S. Challenging issues of ambient activity recognition for cognitive assistance. Handbook of Research on Ambient Intelligence and Smart Environments: Trends and Perspectives, Mastrogiovanni, F. and Chong, N. (Eds.), IGI global, pp.1-32, (2010).

[51] : Roy, P. Un modèle hybride de reconnaissance de plans pour les patients Alzheimer :Dilemme entrecroisé/erroné. Thèse de maîtrise. 2007. pps 19-20.

[52] : Russel, S., and Norvig, P. Artificial Intelligence : A Modern Approach. (second ed.)Prentice-Hall, Englewood Cliffs, NJ (2003).

[53] : Dubois, D., and Prade, H. "Possibility theory as a basis for qualitative decision theory. In Proc. of the 14th Inter. Joint Conf. on Artificial Intelligence (IJCAI'95), Montréal, Canada, August 1995.

[54] : Katz, S., Ford, A. B., Moskowitz, R. W., Jackson, B. A., and Jaffe, M. W. Studies of illness in the aged. The index of ALS, a standard measure of biological and psychosocial function. JAMA 1963;185:914-9.

[55] : Schwartz, M. F., Segal, M., Veramonti, T., Ferraro, M., and Buxbaum, L. J. The Naturalistic Action Test: A standardised assessment for everyday action impairment. Neuropsychological Rehabilitation, 12, (2002), 311–339.

[56] : Rogers, W. A., Meyer, B., Walker, N., and Fisk, A. D. Functional limitations to daily living tasks in the aged: A focus group analysis. Human Factors, Mars 1998, Vol. 40 Issue 1, pps 111-125.

[57] : Geib, C. W., and Goldman, R. P. Plan recognition in intrusion detection systems. In the Proceedings of the DARPA Information Survivability Conference and Exposition (DISCEX), 2001.

[58] : Chen, L., and Khalilb, I. Activity Recognition: Approaches, Practices and Trends. Atlantis Ambient and Pervasive Intelligence Volume 4, 2011, pps 1-31

[59] : Morris, T. Computer Vision and Image Processing. Palgrave Macmillan, 2003.

[60] : Duong, T., Bui, H. H., Phung, D. Q., and Venkatesh, S. Activity recognition and abnormality detection with the switching hidden semi-markov model. CVPR '05 Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05) - Volume 1 - Volume 01 Pages 838 - 845

[61] : Keogh, E., Chakrabarti, K., Pazzani, M., and Mehrotra, S. (2000). Dimensionality Reduction for Fast Similarity Search in Large Time Series Databases. Knowledge and Information Systems 3(3): 263-286.

[62] : Barbic, J., Safonova, A., Pan, J. Y., Faloutsos, C., Hodgins, J. K., and Pollard, N. S. Segmenting motion capture data into distinct behaviors. GI '04 Proceedings of Graphics Interface, 2004, Pages 185-194.

[63] : Olguìn, D. O., and Pentland, A. S. Human activity recognition : Accuracy across common locations for weareable sensors. Proceedings of the 10th International Symposium on Wearable Computers (Student Colloquium), 2006, pps. 11-13.

[64] : Ravi, N., Dandekar, N., Mysore, P., and Littman, M. L. Activity recognition from accelerometer data. In AAAI, 2005, pps1541–1546.

[65] : Kasteren , T. V. Activity Recognition for Health Monitoring Elderly using Temporal Probabilistic Models. These, 2011.

[66] : Al-Omari, S. A., and Sumari, P. An overview of Mobile Ad Hoc Networks For The Existing Protocols and Applications". The International Journal on Applications of Graph Theory in Wireless Ad hoc Networks and Sensor Networks (GRAPH-HOC),Vol.1, No.1, March 2010.

[67] : Jakkula, V. R., and Cook, D. J. Mining Sensor Data in Smart Environment for Temporal Activity Prediction. Poster session at the ACM SIGKDD, San Jose, CA, August 2007.

[68] : Allen, J. F., and Ferguson, G. Actions and events in interval temporal logic. In Spatial and Temporal Reasoning. O. Stock, ed., Kluwer, Dordrecht, Netherlands, 1997, 205-245.

[69] : Zaki, M. J., and Hsiao, C. CHARM: An efficient algorithm for closed association rule mining. RPI Technical Report 99-10,1999.

[70] : Agrawal, R., and Srikant, R. Mining sequential patterns. In ICDE, 1995, pps 3-14.

[71] : Pasquier, N., Bastide, Y., Taouil, R., and Lakhal, L. Discovering frequent closed itemsets for association rules. In 7th Intl. Conf. on Database Theory, January 1999.

[72] : Pei, J., Han, J., and Mao, R. CLOSET: An efficient algorithm for mining frequent closed itemsets. Proc. of ACM SIGMOD DMKD Workshop, Dallas, TX, May, 2000.

[73] : Zaki, M. J. SPADE: An Efficient Algorithm for Mining Frequent Sequences. In Machine Learning Journal, 42(1/2): 31-60, 2001.

[74] : Pei, J., Han, J., Mortazavi-Asi, B., and Pinto, H. PrefixSpan : Mining Sequential Patterns Efficiently by Prefix-Projected Pattern Growth. In Proc. 2001 Int. Conf. on Data Engineering , 2001.

[75] : Ayres, J., Flannick, J., Gehrke, J., and Yiu, T. Sequential PAttern Mining Using A Bitmap Representation. In Proc. 2002 Int. Conf. Knowledge Discovery and Data Mining, 2002.

[76] : Agrawal, R., and Srikant, R. Fast algorithms for mining association rules in large databases. Proceedings of the 20th International Conference on Very Large Data Bases, VLDB, pages 487-499, Santiago, Chile, September 1994.