Table des matières

Liste des Figures

Liste des Tableaux

Liste des Algorithmes

Liste des Graphes

Liste des Acronymes

AVQ	Activité de Vie Quotidienne
Iam	Intelligence ambiante
FD	Fouille de Données
FDT	Fouille de Données Temporelles
BD	Base de Données
IA	Intelligence Artificielle
HMM	Modèle de Markov Caché
LIARA	Laboratoire d'Intelligence Ambiante pour la Reconnaissance d'Activités
UQAC	Université du Québec À Chicoutimi
RFID	Identification par Radio Fréquence
IMU	Unité de Mesure Inertielle
GMM	Modèle de Mélange Gaussien
eClass	Classificateur Évolutif
K-NN	K plus Proches Voisins
TS	Séries Temporelles
ARIMA	AutoRegressive Integrated Moving Average
VAR	Modèle Vectoriel AutoRégressif
KPSS	Test de stationnarité par Kwiatko,wski Phillips, Schmidt, Shin
AICC	Critère Corrigé d'Information d'Akaike
FCI	Fondation Canadienne pour l'Innovation
FQRNT	Fonds de Recherche du Québec – Nature et Technologies

Chapitre 1

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 million après juste une génération. Les conséquences de cette maladie sont désastreuses sur le patient, sa famille et toute la société. Quand une personne est atteinte de cette maladie, ses capacités cognitives se détériorent progressivement jusqu'au point d'être incapable d'effectuer, par elle-même, ses activités de vie quotidienne (AVQ). Pour remédier à cette perte d'autonomie, le patient[1] doit être assisté en permanence pour le restant de sa vie [2]. Ce type d'assistance, que ça soit effectué par un membre de sa famille ou un soignant, a de nombreux désavantages dont son coût très élevé. En 2014 au Canada, 231 millions d'heures d'assistance ont été réalisées, augmentant le coût relié à cette maladie à 15 milliards de dollars. Après juste une génération, les prévisions sont de 756 millions heures et 153 milliards de dollars [1].

[1] Patient, malade ou agent acteur sont utilisés indifféremment pour désigner la personne assistée.

La relation complexe qui s'installe entre la personne aidante et le patient, qui souhaite préserver son intimité, est un autre inconvénient de cette assistance traditionnelle [2].

Dans un souci de réduire l'impact de ce fléau et dans l'absence de signes prometteurs de découverte d'un remède efficace dans le proche future [3], de récentes recherches se sont orientées vers la réduction des effets négatifs de l'assistance traditionnelle. Profitant de l'évolution des technologies de l'information, de l'électronique et surtout de l'émergence du domaine de l'intelligence ambiante (Iam) [4], ces recherches se focalisent sur la conception d'une assistance technologique où un agent artificiel, appelé aussi agent ambiant, soutiendrait l'aidant dans sa tâche. Par définition, l'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. Dans la vision de l'assistance technologique, des senseurs sont installés à l'intérieur de la maison du patient, qui sera nommée par ce fait un habitat intelligent [5]. L'agent ambiant sera conçu pour travailler d'une façon similaire à celle de la personne aidante puisqu'ils vont collaborer. Certes, l'agent artificiel utilisera des moyens différents, mais il effectuera les trois étapes de l'assistance : l'observation, la détection de comportements anormaux[2] et l'intervention pour proposer de l'aide après la détection d'erreurs. Pour la première et la troisième étape, les deux types de senseurs vont être utilisés; les capteurs et les effecteurs. L'étape d'observation sera réalisée grâce à l'analyse des mesures envoyées par les capteurs installés sur les différents objets de la maison. Par exemple: un capteur électromagnétique placé sur un tiroir signalera son ouverture ou fermeture, un tapis à capteurs de pression localisera le patient dans l'appartement, une étiquette RFID sur une tasse aidera à la

[2] Dans cette thèse, comportements anormaux et comportements erronés sont utilisés indifféremment pour dire que c'est un comportement inhabituel de l'agent acteur.

localiser, etc. Pour proposer de l'aide au patient, différents effecteurs peuvent être choisis : un message vocal sur des haut-parleurs, une vidéo sur une télévision ou des médias plus discrets comme la lumière, les émoticônes, les bips [6], etc. La deuxième étape est la phase intelligente de ce processus où les informations envoyées par les senseurs sont analysées dans le but de détecter tout comportement anormal et ainsi utiliser les effecteurs pour le corriger. Une des principales capacités, dont on veut doter l'agent ambiant, est de pouvoir indiquer la prochaine action au patient de la maladie d'Alzheimer quand il n'est plus capable de terminer son activité entamée. Pour classifier un comportement d'anormal, il faut d'abord définir tous les comportements normaux puis les comparer aux actions détectées. Selon l'approche adoptée, les comportements normaux peuvent être des signaux électriques [7], des transitions d'états de senseurs pour des périodes définies [8] ou des modèles d'activités comme proposés dans notre approche [9], etc. La détection de comportements erronés passe donc par la reconnaissance d'activités qui représente le plus grand défi de l'assistance technologique. Avant d'explorer ce dernier domaine et pour mieux réussir notre assistance technologique, il semble essentiel de définir les différents types d'erreurs susceptibles d'être commis par la personne observée. Pour cela, la prochaine section est consacrée à la maladie d'Alzheimer.

1.2 La maladie d'Alzheimer

Décrite pour la première fois par Alois Alzheimer en 1907, la maladie d'Alzheimer est une dégénérescence progressive du cortex cérébral et d'autres zones du cerveau qui entraîne la démence [10]. 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 aussi par la suite comme : l'incapacité d'effectuer des AVQs, le changement de la personnalité et du tempérament, la désintégration lente de la personnalité, la perte graduelle de la maîtrise du corps, les hallucinations, le délire [10], etc.

Selon le nombre d'années après l'atteinte de cette maladie, Gauthier [11] classifie le patient dans l'une des trois phases : Initiale, Intermédiaire ou Avancée. Il utilise la phase par la suite pour spécifier le niveau de dépendance du malade. Le Tableau 1-1définit la phase et l'autonomie selon le nombre d'années de la maladie.

Tableau 1-1 Évaluation de la dysfonction globale et cognitive associée aux trois phases de la maladie

Phases	Durée (années)	Échelle de détérioration globale [3]	Échelle MEEM [4]	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

Dans la phase initiale, 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 commence à avoir des problèmes sans qu'ils soient très fréquents. Durant cette phase de la maladie, le patient est confronté à de petits problèmes de concentration diminuant son attention sur une tâche particulière sans vraiment l'empêcher de mener une vie normale.

[3] 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).
[4] 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).

C’est durant cette phase que le patient doit être observé et ses comportements normaux déterminés. Sachant que le patient est susceptible de commettre quelques erreurs pendant cette phase, il est plus judicieux que le comportement normal corresponde à la manière la plus fréquente d’effectuer une tâche au lieu de toutes les possibilités d’effectuer la même tâche. Cette remarque est très importante, surtout que notre approche proposée pour l’assistance technologique est basée dessus. Il faut aussi préciser que chacune des deux solutions a ses propres avantages et inconvénients. Considérer le comportement fréquent comme le seul comportement normal pour une tâche facilite la création des modèles d’activités ainsi que la proposition d’aide par l’agent ambiant, mais sans un algorithme évolué de détection d’erreurs, l’agent peut classer une autre façon d’effectuer une tâche comme une erreur d’exécution. Par contre, considérer toutes les possibilités d’effectuer la même tâche comme comportements normaux complique grandement la création des modèles d’activités. De plus, il ne règle pas totalement l’inconvénient de la première solution puisqu’une erreur peut être enregistrée comme une manière particulière d’effectuer une tâche lors de la création des modèles d’activités.

Dans la phase intermédiaire, une supervision est requise puisque, 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. [12], [13] classifie ces erreurs en six catégories :

1. Erreurs d'initiation : Quand le patient n'arrive pas à amorcer une activité. Par exemple, même s'il est conscient qu'il doit prendre ses médicaments, il n'arrive

toujours pas à effectuer la première action de cette activité qui pourrait être *prendre un vers d'eau*.

2. Erreurs d'organisation : Quand le patient n'utilise pas les bons outils pour réaliser une activité. Il 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 entamée. En préparant son petit déjeuner, le patient peut par exemple oublier de griller le pain et le tartiner directement.
4. Erreurs de séquence : Quand le patient réalise de façon désordonnée les différentes étapes de l'activité.
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".

Donc, d'après le Tableau 1-1, le patient a besoin d'une assistance d’environ quatre ans, de la fin de la phase initiale jusqu’au début de la phase avancée. 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 [2], [10].

Pour remédier à tous ces problèmes, l'assistance technologique est donc proposée. Malheureusement, jusqu'à présent elle souffre toujours de plusieurs problèmes dans ses différentes étapes et surtout dans la reconnaissance d'activités. La prochaine section présente la problématique générale ainsi que quelques problèmes rencontrés durant ce processus compliqué.

1.3 Problématique générale

Après cette présentation de la maladie d'Alzheimer, la conception d'une assistance technologique doit donc considérer les différentes erreurs susceptibles d'être commises par les patients de cette maladie. Si la majorité des approches existantes [14]–[17] sont conçues pour répondre aux erreurs de réalisation, elles s'avèrent inefficaces pour traiter d'autres catégories d'erreurs comme celles d'initiation. En d'autres termes, si le patient commence une activité et n'arrive plus à effectuer les prochaines actions, ces approches peuvent l'aider en lui envoyant des messages via les effecteurs pour lui indiquer la prochaine étape à exécuter. Par contre, s'il n'arrive même pas à amorcer l'activité, elles ne peuvent pas la prédire ou la reconnaître, puisqu'aucune action n'est détectée, et donc elles ne peuvent l'assister. La prise en charge de ces contraintes doit se faire dans une

vision globale et se refléter dans toutes les étapes de l'assistance commençant par l'étape d'observation. Certes, l'étape d'observation se résume à la réception et la sauvegarde des mesures effectuées par les capteurs, mais le choix en lui-même des capteurs à intégrer dans la maison influence directement l'étape de détection de comportement anormal et sa capacité à gérer les différentes erreurs. En effet, l'utilisation de caméras, de senseurs portables ou de senseurs sur les objets définit le domaine auquel appartiendra la détection de comportements anormaux : le domaine de traitement d'image [17], celui de traitement de signaux [7], etc.

En général, quel que soit le domaine auquel appartiendra la détection de comportement anormal, cela passe par la prédiction ou la reconnaissance du comportement visé par le patient, que nous définissons par la prédiction d'activités et la reconnaissance d'activités. Puis, par sa comparaison avec les récentes actions détectées. Plusieurs techniques peuvent être utilisées pour la prédiction d'activités, dont les séries temporelles [18] que nous allons explorer dans ce travail. La reconnaissance d'activités est aussi traitée différemment selon l'approche et les techniques utilisées, mais le processus général suivi est toujours le même. Tout d'abord, toutes les AVQs que le patient a l'habitude d'effectuer, que nous appelons les modèles d'activités, sont identifiées. Ensuite, parmi ces modèles d'activités, celui qui explique le mieux les récentes actions détectées est sélectionné. Pour expliquer ce processus, considérons l'exemple suivant.

Supposons que nous possédons trois modèles d'activités composés des actions suivantes :

- *Préparer thé* : *Prendre tasse, Bouillir de l'eau, Prendre thé.*
- *Préparer café* : *Prendre tasse, Prendre café, Prendre sucre.*

- *Regarder télévision* : *S'assoir au fauteuil*, *Prendre télécommande.*

Si la récente action détectée est *Prendre tasse*, alors l'activité qui sera reconnue est soit *Préparer thé* ou *Préparer café* puisqu'elles contiennent toutes les deux *Prendre tasse*. Si par la suite *Bouillir de l'eau* est détectée, l'activité reconnue sera définitivement *Préparer thé*.

Une fois *Préparer thé* est reconnue et selon l'algorithme utilisé pour la détection de comportement anormal, si la prochaine action est différente de *Prendre thé* ou si aucune action n'est détectée après un certain temps, un comportement anormal peut être signalé et l'effecteur adéquat sera utilisé pour inciter le patient à prendre le thé.

Malheureusement, cet exemple est trop simpliste et le processus de la reconnaissance d'activités est beaucoup plus complexe, surtout que l'activité doit être reconnue instantanément pour permettre à l'agent ambiant d'intervenir au moment opportun. Cette complexité, qui est la problématique générale de notre recherche, est causée par le très grand nombre d'AVQs qu'un patient peut avoir l'habitude d'effectuer dans sa journée, que nous appelons nombre d'hypothèses. Elle rend la reconnaissance d'activités, dans la vraie vie, très difficile en temps réel. En effet, supposons que nous possédons ***m*** modèles d'activités et que chacun d'entre eux est composé au plus de ***n*** actions, le temps de recherche de l'activité sera donc dans l'ordre de ***O(m.n)***. Alors, il est évident et indispensable de minimiser ***m*** pour accélérer cette recherche. En d'autres termes, pour que la reconnaissance d'activités se fasse en temps réel, il faut réduire le nombre d'hypothèses.

La réduction du nombre d'hypothèses s'effectue généralement en se basant sur des informations spatiales ou temporelles. Si nous revenons à notre exemple et si, grâce à un

tapis à capteurs de pression, le patient est localisé dans le salon, les deux premières activités peuvent être écartées : *Préparer thé* et *Préparer café*. Dans notre recherche, nous nous basons sur les informations temporelles pour réduire le nombre d'hypothèses en considérant exclusivement, dans un premier lieu, les activités que le patient a l'habitude d'effectuer au moment de la reconnaissance. Revenons encore une fois à notre exemple et supposons que l'analyse des observations a montré qu'il a plus l'habitude d'effectuer les activités *Préparer thé* et *Préparer café* entre 08h:00 et 10h:00 et *Regarder télévision* entre 18h:00 et 21h:00, si par exemple l'heure courante est 09h:00, *Regarder télévision* peut être écartée à ce moment.

À part la réduction du nombre d'hypothèses, le système de reconnaissance d'activités doit répondre à plusieurs autres difficultés et problèmes pour qu'il soit fonctionnel et efficace. La création des modèles d'activités est une tâche assez délicate, surtout qu'elle doit se faire à partir des mesures envoyées par les capteurs d'une façon non supervisée. Le problème d'équiprobabilité, montré dans notre exemple, peut aussi se produire quand les récentes actions détectées appartiennent à plusieurs modèles d'activités, ce qui induit à une confusion lors de la spécification de l'activité reconnue. Le dernier problème que nous allons citer correspond aux erreurs d'initiation qui font en sorte qu'aucune action n'est détectée ce qui empêche la reconnaissance.

Pour mieux situer notre approche et avant de détailler les différentes techniques que nous avons développées pour répondre à ces différents problèmes, la prochaine section propose une vue générale sur les catégories de travaux existants sur la reconnaissance d'activités.

1.4 Les catégories des approches de reconnaissance d'activités

Comme nous l'avons déjà mentionné, la reconnaissance d'activités est l'étape la plus délicate dans l'assistance technologique, ce qui explique le grand nombre de recherches consacrées à ce domaine [8], [9], [15]–[17], [19]. Pour mieux comprendre la reconnaissance d'activités, citons sa définition par Schmidt et al. [20] : "Prendre en entrée une séquence d'actions exécutées par un acteur et inférer le but poursuivi en organisant la séquence d'actions en terme d'une structure de plan". L'acteur suit donc un plan d'actions afin d'atteindre un but bien déterminé. Dans ce temps, l'agent ambiant, qui doit posséder tous les plans d'actions ou ce que nous appelons les modèles d'activités, essaie de trouver le plan commencé par l'acteur en se basant sur les actions observées. Pour doter l'agent ambiant d'un système d'inférence efficace lui permettant de trouver en temps réel le modèle d'activité entamé par l'acteur, différentes techniques sont utilisées : modèles markoviens [21], réseaux bayésiens [22], fouille de données [23], etc. Toutes ces techniques peuvent être regroupées dans trois catégories : les approches logiques, les approches probabilistes et les approches basées sur l'apprentissage automatique.

1.4.1 Les approches logiques

Les approches logiques [24], [25] se basent sur une série de déductions logiques pour sélectionner, parmi les modèles d'activités, l'ensemble des activités qui peuvent expliquer un ensemble d'actions observées. Toutes les approches proposées dans cette catégorie sont dérivées du travail de Kautz [24] que nous allons détailler pour expliquer les lignes directrices de cette catégorie.

Tout d'abord, Kautz présume que l'agent ambiant possède une librairie de plans comme celle schématisée à la Figure 1-1 où les plans et les actions sont considérés indifféremment comme des événements.

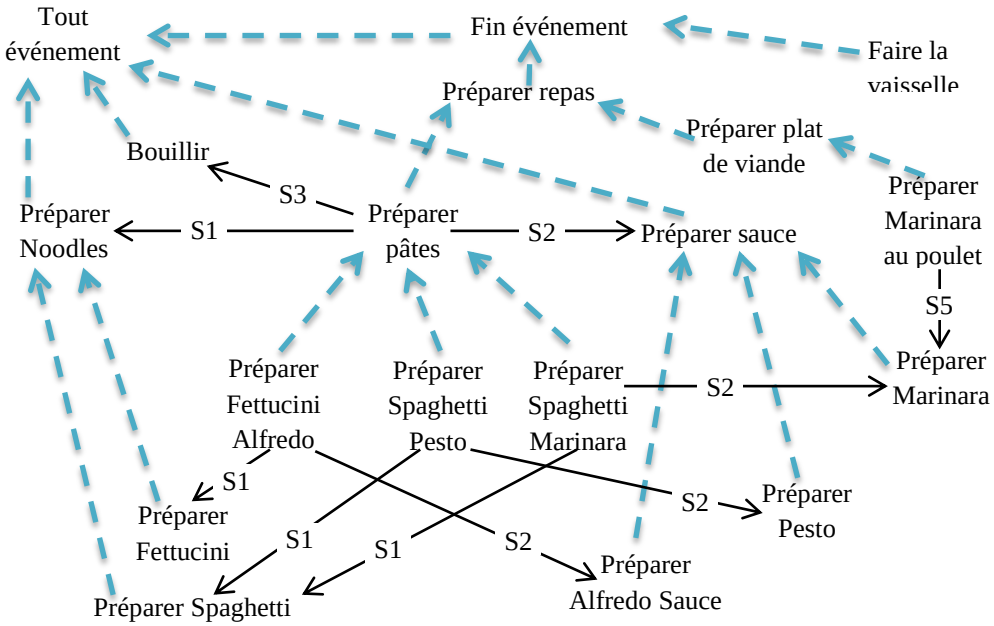


Figure 1-1. Exemple de librairie de plans par Kautz (adaptée de [24])

Chaque événement réfère à son abstraction (arc bleu discontinu) ou à une de ses décompositions (arc noir continu). Par exemple : *Préparer repas* est une abstraction (ABS) de *Préparer pâtes* et *Bouillir* est une décomposition (DEC) ou une étape de *Préparer pâtes*. Le système d'inférence est ensuite créé en utilisant la logique du premier ordre et en se basant sur les quatre hypothèses suivantes :

(EXA) $\forall x. E_0(x) \supset (E_1(x) \vee E_2(x) \vee \ldots \vee E_n(x))$

(DJA) $\forall x. \neg E_1(x) \vee \neg E_2(x)$

(CUA) $\forall x. E(x) \supset (End(x) \vee (\exists y. E_{1,0}(y) \wedge f_{1i}(y) = x) \vee \ldots \vee (\exists y. E_{m,0}(y) \wedge f_{mi}(y) = x)$

(MCA$_n$) $\forall x_1 \ldots \forall x_n. (End(x_1) \ldots End(x_n) \supset \bigvee_{i,j}(x_i = x_j)), \quad i, j \in [1, n] \wedge i \neq j$

(EXA) stipule que si une observation *x* est associée à un événement de type E_0 alors elle est associée à au moins une de ses spécialisations $E_1, E_2, \dots E_n$. Par exemple, si *Préparer pâtes* est observée alors soit *Préparer Fettucini Alfredo*, *Préparer Spaghetti Pesto* ou *Préparer Spaghetti Marinara* est valide.

(DJA) complète (EXA). Elle stipule que si une spécialisation d'une abstraction est valide alors les autres spécialisations de la même abstraction sont invalides. Par exemple, si *Préparer Fettucini Alfredo* est valide alors *Préparer Spaghetti Pesto* et *Préparer Spaghetti Marinara* sont invalides.

(CUA) assume que chaque action est soit une action de Fin (*Fin événement*) ou une étape d'un plan d'activité. *Bouillir* par exemple n'est pas un *Fin événement*, mais c'est une étape de *Préparer pâtes*.

(MCA) stipule que quand deux événements observés appartiennent à un plan, c'est ce plan qui sera considéré et non pas tous les plans qui contiennent un des deux événements.

Le système d'inférence commence par l'application de (CUA) après chaque observation. Par la suite, il utilise (ABS) récursivement pour obtenir les *Fin événement*. (DJA) et (EXA) sont ensuite exécutés pour réduire le nombre de plans. (MCA) est utilisée comme dernière étape pour fusionner les différentes observations et diminuer encore plus le nombre de plans possible.

Le grand avantage de l'approche de Kautz et des approches logiques en générale est leur efficacité 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 supposent que la librairie de plans est complète et que l'acteur 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, ce qui peut amener l'agent ambiant à ignorer le plan visé. Les approches logiques souffrent aussi du problème d'équiprobabilité qui se manifeste dans l'incapacité de reconnaître le plan visé parmi l'ensemble sélectionné par ces approches. Tous les plans de l'ensemble sélectionné ont la même probabilité de correspondre à l'activité recherchée.

1.4.2 Les approches probabilistes

Les approches probabilistes [26]–[28] sont basées sur des modèles markoviens [21] ou des réseaux bayésiens [22]. Pour utiliser ces approches, une probabilité initiale est assignée à chaque modèle d'activité. Par la suite, ces probabilités sont mises à jour en fonction des nouvelles observations.

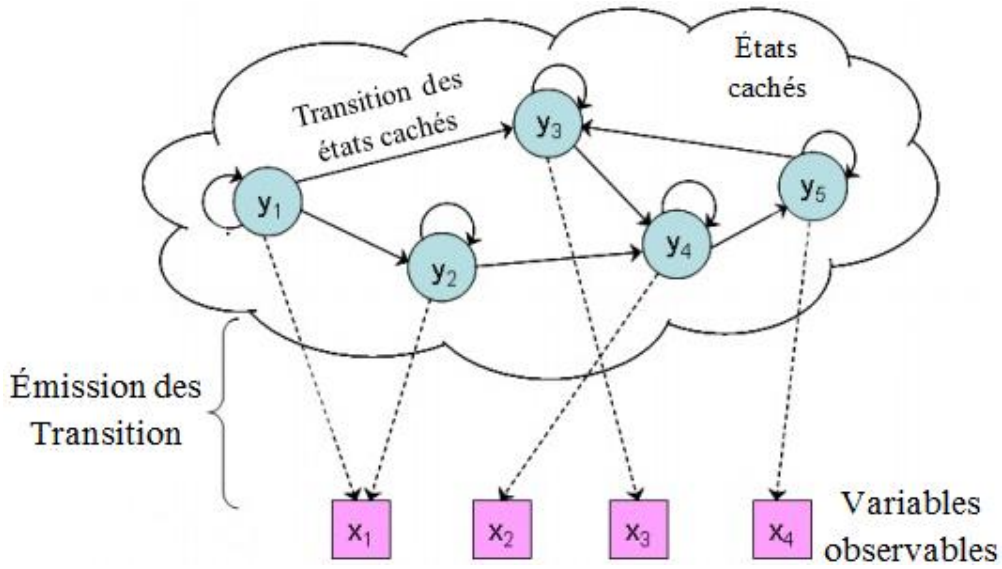


Figure 1-2. Représentation graphique d'un HMM (adaptée de [29])

La Figure 1-2, adaptée de Eunju et al. [29], montre une représentation graphique du modèle Markovien caché (HMM) qui peut être utilisé dans la reconnaissance d'activités.

Dans un HMM, nous observons des variables ($X_1,\ldots,X_n$). Chaque variable est assignée à un ou plusieurs états appelés états cachés parce qu'ils ne sont pas observables ($y_1, \ldots, y_m$). Dans notre cas de reconnaissance d'activités, les états cachés peuvent être des modèles d'activités ou des actions d'activités et les variables observables sont les senseurs. HMM repose sur deux hypothèses d'indépendance pour l'inférence :

1. L'hypothèse de transition du premier ordre de Markov :

$$P(Y_t|Y_1, Y_2, \ldots, Y_{t-1}) = P(Y_t|Y_{t-1})$$

   L'état futur ne dépend que de l'état actuel et non sur les états passés [21].

2. L'hypothèse d'indépendance conditionnelle des paramètres d'observation :

$$P(X_t|Y_t, X_1, X_2, \ldots, X_{t-1}, Y_1, Y_2, \ldots, Y_{t-1}) = P(X_t|Y_t)$$

   La variable observable à l'instant *t*, X_t, ne dépend que de l'état caché actuel Y_t.

Pour trouver la séquence d'états cachés la plus probable à partir d'une séquence de variables observée, HMM trouve la séquence d'états qui maximise la probabilité conjointe $P(X,Y)$ à partir de la probabilité de transition $P(Y_{t-1}|Y_t)$ et de la probabilité d'observation $P(X_t|Y_t)$.

$$P(X, Y) = \pi_{t=1}^{T} P(Y_t|Y_{t-1})\, P(X_t|Y_t)$$

Pour définir les probabilités et relier les variables observables aux états cachés, des connaissances expertes ou une phase d'apprentissage peuvent être utilisées. La Figure 1-3, affichée dans la page suivante, présente un exemple du HMM de l'activité *Manger*.

Les approches probabilistes règlent tous les problèmes cités des approches logiques. Elles permettent de résoudre le problème d'irrationalité de l'acteur du fait qu'une action qui n'a aucun rapport avec l'objectif visé ne l'éliminerait plus de l'ensemble sélectionné, mais elle va juste modifier d'une façon non significative les différentes

probabilités calculées. Le problème d'équiprobabilité ne se pose pas aussi parce que le modèle d'activité avec la probabilité la plus élevée sera toujours choisi comme objectif visé. L'inconvénient majeur des approches probabilistes est la lourdeur des calculs et du traitement qui ralentit grandement le temps d'exécution alors que la reconnaissance d'activités doit se faire en temps réel pour permettre à l'agent ambiant d'intervenir au moment opportun. La complexité des calculs vient du problème du nombre d'hypothèses puisque pour chaque action observée, les probabilités de tous les modèles d'activités de la base de connaissance doivent être mises à jour.

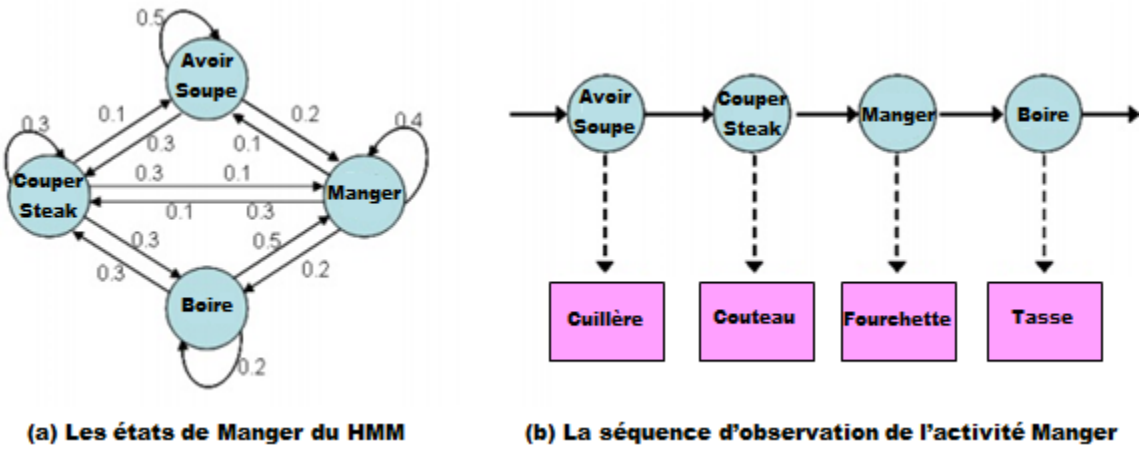


Figure 1-3. Exemple de HMM pour l'activité *Manger*.

1.4.3 Les approches basées sur l'apprentissage automatique

Après la définition de Schmidt, Patterson *et al.* [30] ont redéfini la reconnaissance d'activités par : "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 à la précédente 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 précision est très importante parce qu'elle permet aux nouvelles approches de ne plus se baser uniquement sur les actions observées pour discriminer des modèles

d'activités et n'en garder que les plus probables, mais elle leurs permet aussi d'utiliser d'autres contraintes spatiales ou temporelles. En effet, détectant que l'acteur est dans le salon peut discriminer les modèles d'activités qui se produisent en dehors du salon comme *Préparer café*, ou l'apprentissage d'une habitude temporelle comme le fait d'effectuer *Préparer café* entre 8h00 et 10h00 peut aussi la discriminer si la reconnaissance d'activités est exécutée en dehors de cet intervalle. L'apprentissage de telles habitudes et l'utilisation de ces contraintes demandent l'exploitation d'un volume élevé de données, générées par les senseurs, de nature complexe comme les données temporelles. Pour cette raison, ces approches font recours au domaine de l'apprentissage automatique [31] et surtout à la fouille de données [23]. Avant de donner un exemple d'une approche de cette catégorie, il faut préciser que le chapitre suivant est consacré aux différentes techniques de la fouille de données utilisées pour la reconnaissance d'activités.

Dans le travail de Jurek et al. [14], les modèles d'activités sont créés d'une manière non supervisée en utilisant la technique de segmentation. L'algorithme K-means [32] est utilisé plusieurs fois avec un nombre d'éléments et de clusters choisis au hasard tout en prenant en considération de ne pas produire des clusters vides. Ensuite, une matrice de support A_k est créée pour représenter le support apporté par chaque cluster, représenté par une ligne, à chaque classe, représentée par une colonne.

$$A_k[i,j] = \begin{cases} \dfrac{N_{ij} - {}^{N_i}/_{M}}{N_i - {}^{N_i}/_{M}} & Si\ N_{ij} - {}^{N_i}/_{M} \geq 0 \\ \dfrac{N_{ij} - {}^{N_i}/_{M}}{{}^{N_i}/_{M}} & Sinon \end{cases}$$

où N_i représente le nombre total d'occurrences dans le cluster i, N_{ij} fait référence au nombre d'occurrences de la classe j qui appartiennent au cluster i et M représente le nombre de classes dans la classification du problème considéré.

La recherche d'activité entamée est effectuée en se basant sur la technique de classification. La distance euclidienne entre chaque cluster et les récentes observations est calculée, puis l'activité reconnue est le cluster le plus proche aux récentes observations :

$$ExSupp(c_j) = \sum_{k=1}^{k} \begin{cases} e^{\frac{A_k[i_k,j]}{1+d(x,x^k)}} & Si \quad A_k[i_k,j] > -1 \\ 0 & Sinon \end{cases}$$

où i_k est la ligne de la matrice A_k qui représente le cluster sélectionné et d représente la distance euclidienne métrique.

1.5 Contribution

Notre travail s'inscrit dans cette nouvelle tendance à vouloir exploiter les contraintes temporelles pour proposer une solution à la problématique de la reconnaissance d'activités au sein de l'habitat intelligent afin d'offrir une assistance technologique aux patients de la maladie d'Alzheimer. Quant à notre contribution, elle ne concerne pas seulement la réduction du nombre d'hypothèses, mais elle propose aussi des solutions aux différents problèmes, déjà cités, que nous avons rencontrés durant ce processus.

L'approche commence par une transformation de la base de données créée par l'enregistrement de toutes les mesures de tous les senseurs en tout temps. Au Laboratoire

d'Intelligence Ambiante pour la Reconnaissance d'Activités (LIARA) où nous avons testé notre approche, il existe 114 senseurs qui envoient leurs mesures tous les 500 millisecondes. La base de données, ainsi créée, est très volumineuse ce qui rend son exploitation presque impossible, d'où la nécessité de la transformation. À partir de la nouvelle base de données, nous créons tous les modèles d'activités qui résument les AVQs que le patient est habitué d'effectuer. Dans un premier travail [33], nous avons utilisé BIDE [34], un algorithme du "Sequential pattern mining", pour trouver les plus longues sous séquences de senseurs qui se répètent assez fréquemment. Bien que les résultats fussent satisfaisants, les modèles d'activités créés avec BIDE ne contenaient aucune information temporelle nécessaire à la détection d'erreurs. Pour cette raison, nous avons créé un nouvel algorithme [35] appartenant au domaine de l'extraction des motifs fréquents "Frequent pattern mining" [36] qui est un sous domaine de la fouille des données. La particularité de cet algorithme est qu'il permet de trouver les sous-séquences fréquentes, tout en estimant le temps normal d'activation des senseurs. Cette dernière information est très utile parce que nous l'utilisons pour détecter les erreurs du patient et pour décider du moment opportun pour lui envoyer de l'aide [19].

Pour répondre au problème du nombre d'hypothèses, nous avons, dans la première approche, utilisé une segmentation temporelle pour créer des intervalles de temps qui résument les heures de début des modèles d'activités [37]. Ensuite, les modèles d'activités qui ne possèdent pas un intervalle qui comprend l'heure courante sont écartés de la recherche de l'activité entamée. Le seul problème avec cette solution est que les intervalles temporels ne reflètent pas la réalité de tous les jours. Par exemple, si l'acteur prend son petit déjeuner entre 8h00 et 10h00 tous les jours sauf le dimanche, cette activité

sera considérée même le dimanche puisque le même intervalle sera créé pour tous les jours de la semaine. Après une étude approfondie de ce problème, nous avons opté pour une solution qui utilise les séries temporelles pour profiter de leur propriété de saisonnalité. L'approche que nous proposons [38] utilise donc les séries temporelles pour prédire les temps de début des modèles d'activités selon le jour de la semaine. Ces temps sont utilisés par la suite pour ne considérer que les modèles d'activités avec un temps prédit assez proche de l'heure courante. Ils sont aussi utilisés pour répondre au problème des erreurs d'initiation et celui de l'équiprobabilité en programmant un réseau bayésien qui fait en sorte que le modèle d'activité avec le temps prédit le plus proche de l'heure courante soit le plus probable.

Pour que l'assistance technologique soit efficace, il faut que les messages d'aides se déclenchent au bon moment. Notre dernière contribution consistait donc à trouver une méthode permettant la détection d'erreurs dans les plus brefs délais. Nous avons proposé trois solutions et la comparaison entre elles a permis la sélection de celle qui donne les meilleurs résultats.

Doter l'agent ambiant d'une intelligence artificielle lui permettant d'assister une personne atteinte de la maladie d'Alzheimer est une tâche compliquée qui soulève plusieurs défis dans chacune de ses étapes. Nos contributions dans ses différentes étapes, qui ont été publiées dans plusieurs articles [9], [19], [35], [38], ont permis la conception d'une assistance technologique fonctionnelle. Elle nécessite certainement plusieurs améliorations comme l'exploitation des données spatiales que nous n'avons pas considérées durant cette recherche qui visait surtout l'exploitation des données temporelles.

1.6 Objectifs et méthodologie de la recherche

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

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 [10], [12], [13].

Comme nous avons créé un algorithme d'extraction de motifs fréquents, une autre étude sur les techniques de la fouille de données s'imposait et a fait l'objet de la deuxième étape. Cette étape avait aussi comme objectif de bien situer notre algorithme et de comprendre d'autres algorithmes du même domaine, ce qui nous a permis d'intégrer un de ces algorithmes au nôtre pour des résultats encore meilleurs.

Quant à la troisième étape, elle définit plus précisément la reconnaissance d'activités et réalise un état d'art sur les travaux déjà existants. 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 troisième chapitre.

La quatrième étape avait pour objectif de créer une toute nouvelle approche basée sur les contraintes temporelles. L'approche proposée est composée de trois parties : la création des modèles d'activités à partir des mesures des senseurs, la réduction du nombre d'hypothèses en s'appuyant sur les prédictions des séries temporelles et la conception

d’un système de recherche de l’activité entamée capable de spécifier à n’importe quel moment l’activité la plus probable et en fin, l’utilisation des séries temporelles bivariées pour une prédiction assez précise des temps d’activations des senseurs, ce qui permet une meilleure détection d’erreurs.

La dernière étape concerne la validation de notre approche afin de vérifier son opérationnalité. Les tests ont été réalisés avec des données réelles, enregistrées après la simulation des activités matinales d'un patient de la maladie d'Alzheimer au sein d'un habitat intelligent pour une période de vingt-huit jours.

1.7 Organisation de la thèse

Cette thèse est organisée en six chapitres qui viennent détailler chacune des étapes décrites dans la méthodologie de recherche.

Ce premier chapitre permet de bien comprendre la problématique générale et de spécifier les problèmes auxquels nous comptons proposer des solutions. Au cours de ce chapitre, nous avons présenté une étude sur la maladie d’Alzheimer pour avoir une idée sur les différentes erreurs susceptibles d’être commises par les personnes observées. Nous avons aussi présenté les catégories des approches de reconnaissance d'activités pour situer notre approche.

Le deuxième chapitre est consacré à la fouille de données, ses étapes, ses différentes techniques et son utilisation dans les différents domaines, surtout celui de la reconnaissance d’activités. Ce chapitre détaille plus l’extraction des motifs fréquents, le domaine pour lequel nous avons conçu un nouvel algorithme. Il explique aussi d’autres

algorithmes utilisés dans notre approche ou dans les approches citées dans le troisième chapitre réservé à la reconnaissance d'activités.

Le troisième Chapitre définit plus profondément la reconnaissance d'activités et présente différents travaux déjà existants portant sur cette problématique de recherche. Les travaux utilisant les contraintes temporelles sont couverts plus en profondeur afin de les comparer à notre approche.

Dans le quatrième chapitre, nous expliquons en détail l'approche proposée qui offre une solution à l'assistance technologique et surtout à la problématique de la reconnaissance d'activités. Nous commençons par l'explication de notre nouvel algorithme qui permet de créer des modèles d'activités spécifiant une estimation du temps normal entre deux actions successives. Ensuite, nous détaillons la prédiction des heures de début des modèles d'activités avec la technique ARIMA des séries temporelles ainsi que le réseau bayésien utilisé pour trouver à tous moments l'activité la plus probable d'être entamée ou celle déjà entamée. À la fin de ce chapitre, différentes solutions de prédictions des temps d'activation des senseurs sont présentées.

Quant au cinquième chapitre, il présente une vue globale sur les maisons intelligentes et sur les différents choix techniques qui doivent être pris lors de leur conception pour présenter ensuite le LIARA où nos expériences se sont déroulées. Le reste du chapitre est réservé aux tests et validations des différentes étapes de notre approche.

Nous terminons cette thèse par une conclusion générale où nous montrons les avantages et les limites de notre approche. Nous y discutons également les améliorations

à entrevoir et les travaux existants qui peuvent s'intégrer à notre approche pour un meilleur rendement.

Chapitre 2

2 Fouille de données temporelles

2.1 Introduction

Avant l'ère de l'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 découvrir des motifs, faire des extrapolations et à trouver les informations intéressantes qu'il va ensuite véhiculer via des rapports écrits, graphiques et diagrammes. Par la suite, la tâche est devenue trop compliquée pour l'expert [39] et les ordinateurs, avec des algorithmes assez sophistiqués, ont fait leur apparition offrant de nouveaux outils facilitant son travail. De nos jours, ces algorithmes pèchent à répondre aux problèmes pour lesquels ils sont créés. L'une des causes de cette réalité revient à l'information qui est maintenant répartie sur de multiples plateformes et stockée dans une grande variété de formats; texte, image, son, fichier, vidéo, etc. De plus, la quantité de données à analyser devient de plus en plus énorme. En fait, d'après Dunham [40] les données doublent chaque année et pourtant la quantité d'informations utiles dont nous disposons est à la baisse. En effet, comme l'informatique touche presque à tous les domaines, un grand volume de données est généré toutes les

secondes à partir de multiples sources à un point qu'on parle déjà de l'ère des données [41]. Pour avoir une idée sur la quantité de données disponibles ces temps-ci, nous n'avons qu'à penser au nombre de textes, images et vidéos ajoutés quotidiennement aux sites Internet, aux différentes informations échangées sur les réseaux sociaux et aux mesures effectuées par tous les senseurs dans le domaine de l'informatique ubiquitaire [42] qui vise à améliorer le quotidien de l'humain en y intégrant des outils technologiques d'une façon transparente à l'utilisateur, etc. Rappelons que c'est à ce dernier domaine qu'appartient notre recherche qui a pour objectif de prolonger la vie des patients de la maladie d'Alzheimer chez eux d'une manière autonome et sécuritaire, tout en préservant leur intimité.

Le grand volume des données et leurs différents formats ont posé plusieurs défis aux méthodes traditionnelles de sauvegarde et de traitement, ce qui a nécessité l'apparition d'un nouveau domaine appelé mégadonnées "Big data" [41], [43]. En effet, les fameux systèmes de gestion de bases de données relationnelles (SGBDR), qui ont révolutionné la sauvegarde et l'analyse des données dans les dernières décennies, sont devenus impuissants devant l'hétérogénéité et la quantité massive des données actuelles. Le domaine du mégadonnées offre aujourd'hui de nouvelles solutions comme MongoDB [44], le système noSQL (pas seulement SQL) [45], ou Apache Hadoop [46], etc. Quant aux algorithmes habituels, ils sont devenus incapables de traiter ce grand volume de données ou leur temps d'exécution est tellement lent qu'on préfère ne pas les utiliser. Pour cette raison, le domaine du mégadonnées fait souvent appel aux différentes techniques de la fouille de données que nous allons explorer au cours de ce chapitre.

2.2 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'apprentissage automatique [31] et l'intelligence artificielle (IA) [47]. La volonté de rendre les machines intelligentes et capables de prendre des décisions à la place de l'expert a permis l'émergence de ce dernier domaine.

Le développement de l'IA et du domaine des statistiques ont permis la conception de nouveaux algorithmes tellement intéressants qu'ils sont utilisés de nos jours dans la fouille de données, comme l'analyse de régression [48], les réseaux de neurones [49] et les modèles linéaires de classification [50], etc. 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 [39]. Le domaine de la récupération d'informations "information retrieval" [51] a fait sa contribution lui aussi dans cette même période par les techniques de segmentation et de mesure de similarité [52].

L'année 1971 est une année très importante dans l'histoire de la fouille de données. Durant cette année, 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 [40].

Après cette année, plusieurs autres algorithmes très importants ont vu le jour comme les algorithmes génétiques, le k-means [32] pour la segmentation et les

algorithmes des arbres de décisions, etc. Aux années 90 et pour la première fois, le terme entrepôt de données "Data Warehouse" [53] a été utilisé pour décrire une grande base de données composée d'un schéma unique, créé à partir de la consolidation des données opérationnelles et transactionnelles d'une ou plusieurs bases 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 en devenant une partie intégrante des pratiques standards du monde des affaires [40]. 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 FD est utilisée dans des domaines très variés, comme la télécommunication, la sécurité, le marketing, la finance, le marché boursier et le commerce électronique, la santé, etc. Même les gouvernements n'hésitent plus à l'utiliser dans leurs projets. En 2004, dans le rapport des activités fédérales sur la fouille de données, le bureau de comptabilité générale des États-Unis [54] a signalé qu'il y a 199 opérations de fouille de données en cours ou prévu 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.

2.3 Définition de la fouille de données

La définition la plus complète de la fouille de données, à notre avis, est celle présentée par Benoît [55] qui stipule que : la fouille de données est un processus, à plusieurs étapes, d'extraction de connaissances préalablement imprévues à partir de grandes bases de données, et d'application des résultats pour la prise de décision. En d'autres termes, les outils de la fouille de données détectent des motifs à partir des données et déduisent des associations et des règles. Les connaissances 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, les analyse et les traite pour en extraire, non pas des données déjà existantes, mais des connaissances. Enfin, elle formule ces connaissances sous un format compréhensible et les présente comme sortie à l'utilisateur.

Les différentes techniques de la fouille de données permettent de travailler sur tous les types de données existants. Si les fichiers plats, qui contiennent du texte ou des données binaires avec une structure bien déterminée, sont les entrées les plus courantes [56], d'autres entrées sont de plus en plus exploitées par ces techniques comme les bases de données avec des attributs de différents formats : texte, entier, réel, etc. ou les bases de données qui comportent des attributs spéciaux en plus de ceux cités : les bases de données multimédias avec des attributs de type sons, images ou vidéos, les bases de données spatiales avec des attributs de type cartes géographiques, des positions

régionales ou globales ou toute autre donnée géographique et les bases de données temporelles spécifiant pour chaque enregistrement le temps de sa sauvegarde dans la base de données (temps de transaction) et le temps où il s'est déroulé dans le monde réel (temps valide).

Les sorties de la fouille de données sont la représentation finale de la connaissance ou du savoir extrait des données. D'après Han [23], elles doivent être transmises à l'utilisateur final d'une manière lui permettant d'agir sur elles et de fournir une rétroaction au système. Elles peuvent être de différentes formes : des Arbres de décision, des règles de classification (Si valeurs d'attributs = X alors attribut = y), des règles impliquant des relations (Si attribut1 > attribut2 alors classe = y), des segments (clusters), etc.

2.4 Les étapes du processus de la fouille de données

Après une analyse approfondie du problème permettant la spécification des questions auxquelles il faut trouver des réponses [55], le processus de la fouille de données suit les étapes schématisées à la Figure 2-1.

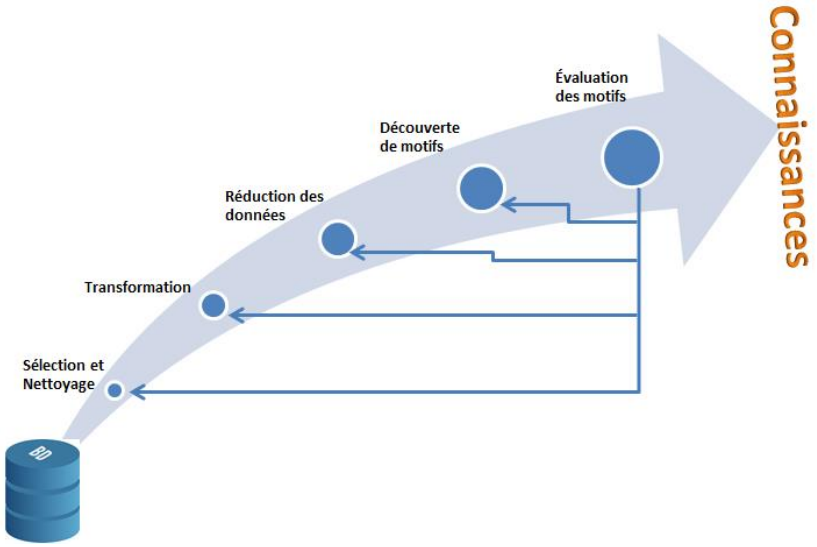


Figure 2-1. Étapes de la fouille de données.

- **La sélection et nettoyage des données** : au cours de cette étape, une solution est proposée pour les données manquantes et celles bruitées. Elles peuvent être estimées ou remplacées par une valeur, sinon tout l'enregistrement est supprimé [39], [55].
- **La transformation des données** : cette étape vise à transformer les données dans un format plus approprié à la fouille de données. Elle peut inclure [23], [57] :
    - Le lissage de données "smoothing" : par exemple, utiliser les moyennes pour remplacer les données erronées.
    - L'agrégation : par exemple, afficher les données mensuellement plutôt que quotidiennement.
    - La généralisation : par exemple, remplacer l'âge exact d'une personne par une des descriptions : petite, jeune ou vieille.
    - La normalisation : changer des valeurs pour qu'elles soient toujours dans une fourchette fixe.
    - La construction d'attributs : ajouter de nouveaux attributs à l'ensemble des données.
- **La réduction des données** : généralement, le volume de données est assez grand pour les exploiter directement. Alors, différentes techniques sont utilisées pour les réduire afin que le processus d'analyse soit gérable, rentable et plus rapide.
    - L'agrégation : la même technique utilisée dans l'étape de transformation.
    - La réduction de dimensionnalité : les attributs non pertinents ou redondants sont supprimés.

- La compression des données : les données sont codées afin de réduire leur taille.
- 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 pour trouver les relations ou les motifs intéressants et utiles [55]. Certains motifs sont plus intéressants que d'autres. Cet "intérêt" est l'une des mesures utilisées pour déterminer l'efficacité de l'algorithme [58].
- **Évaluation des motifs** : cette étape ne valide pas seulement les résultats obtenus et la capacité de la technique utilisée à répondre aux questions préalablement établies, mais elle vérifie plusieurs autres critères comme le temps d'exécution et l'espace mémoire utilisé, etc. Si les résultats ne sont pas satisfaisants, on peut revenir à n'importe quelle étape comme la réduction des données ou le changement de la technique utilisée, sinon les résultats sont présentés à l'utilisateur final d'une manière simple et facilement compréhensible.

Avant d'entamer la prochaine section, nous tenons à signaler l'importance de l'étape de réduction de données qui joue un rôle primordial dans la réussite des techniques de la fouille de données, surtout quand elles traitent des données assez complexes comme les données temporelles. Par exemple, l'algorithme que nous avons créé pour la détection des modèles d'activités, détaillé au chapitre 4, n'a pu gérer la base de données générée à partir des senseurs qu'après une transformation radicale de sa structure permettant la réduction de ses données.

2.5 La fouille de données temporelles

Les bases de données temporelles [59], une des entrées de la fouille de données citées dans la section précédente, augmentent considérablement la complexité des techniques visant à les exploiter puisqu'elles doivent tenir compte des différents types de données temporelles, des relations ou opérateurs temporels et de la granularité temporelle, [60] etc. En effet, l'enregistrement de ces données dans une base de données temporelle attache le reste des attributs à une période de temps qui peut être soit le temps valide ou le temps de transaction. Le temps valide désigne la période de temps pendant laquelle un fait est vrai par rapport au monde réel, tandis que le temps de transaction est la période de temps pendant laquelle un fait est enregistré dans la base de données. L'introduction de telles informations temporelles complique l'exploitation des bases de données puisqu'au lieu d'avoir une seule réalité, plusieurs réalités existent selon le temps comme schématisé à la Figure 2-2.

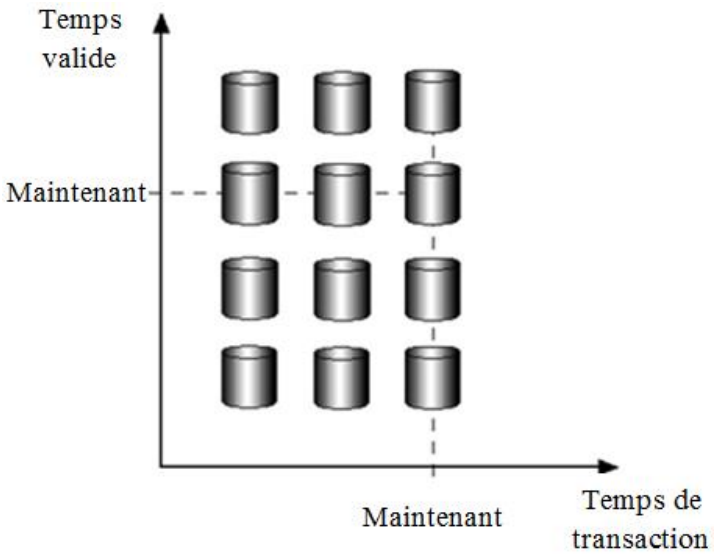


Figure 2-2. Complexité ajoutée par les données temporelles (tirée de [61]).

Les bases de données temporelles sont aujourd'hui capables de sauvegarder des données temporelles encore plus complexes comme les séquences et les séries temporelles. Une séquence peut être définie comme un ensemble ordonné d'événements souvent représenté par une série de symboles nominaux [61]. Il existe deux types de séquences, les séquences ordonnées qui font ressortir une seule information temporelle indiquant que les événements sont ordonnés dans le temps et se produisent séquentiellement et les séquences horodatées qui ajoutent plus de précision temporelle en intégrant le temps de début de chaque événement [56]. Alors qu'une série temporelle (ou chronologique) est définie comme la suite d'observations d'une famille de variables réelles notées $(X_t) \in \theta$ où l'ensemble θ est appelé espace des temps qui peut être discret $(\theta \subset Z)$ ou continu $(\theta \subset R)$ [18].

À cause de cette complexité, certaines techniques ont été modifiées et d'autres ont vu le jour pour créer un sous-domaine de la fouille de données, appelé fouille de données temporelles, capable de bien gérer les données temporelles. Pour qu'une technique de la fouille de données fasse partie de la FDT, elle doit être capable de [61]:

- Répondre à des vraies requêtes du monde réel même sur un grand volume de données temporelles;
- Répondre aux requêtes quantitatives et qualitatives sur les relations temporelles. "*Prendre médicament* est effectuée après une heure de l'activation du senseur S" est un exemple de requêtes quantitatives, alors que "*Prendre médicament* est effectué après *Prendre déjeuner*" est une requête qualitative;
- Représenter la causalité entre les événements temporels. Par exemple, l'activation du senseur S_2 est le résultat de l'activation du senseur S_1;

- Distinguer entre le temps valide et le temps de transaction
- Exprimer la persistance d'une manière parcimonieuse. En d'autres termes, quand un événement se produit, seules les parties du système affectées par cet événement doivent changer;
- Offrir un langage expressif qui permet les mises à jour.

2.6 Techniques de la fouille de données temporelles

La fouille de données temporelles a pour objectif d'extraire de la connaissance après l'analyse des données stockées dans des entrepôts 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 [55]. Selon l'objectif de l'utilisateur, les techniques de la fouille de données temporelles peuvent être catégorisées en quatre tâches générales [62] :

1. **Analyse exploratoire des données (EDA)** [63] : le but de cette tâche est d'explorer simplement les données sans avoir une idée bien précise sur ce qu'on cherche. Les techniques de cette catégorie sont interactives et visuelles.
2. **Modélisation descriptive** [62]: le but d'un modèle descriptif est de décrire le processus générant les données en décrivant, par exemple, la relation entre les variables ou en partitionnant les données en groupes (segmentation), etc.
3. **Modélisation prédictive: classification et de régression** [64]: le but ici est de construire un modèle qui permettra la prédiction de la valeur d'une variable à

partir des valeurs connues des autres variables. Dans la classification, la variable prédite est catégorique, tandis que dans la régression elle est quantitative.

4. **Découverte de motifs et de règles** [65]**:** dans cette catégorie, on cherche des règles qui associent certains attributs ou des motifs avec des caractéristiques spécifiques comme des motifs fréquents (des éléments qui reviennent souvent ensemble) des comportements anormaux ou frauduleux, etc.

Pour réaliser ces différentes tâches, la FDT possède plusieurs techniques, dont la classification, la segmentation et l'extraction des règles d'association que nous expliquons dans le reste de ce chapitre. Le choix de détailler ces trois techniques ainsi que quelques algorithmes qui leurs appartiennent a pour but de faciliter la compréhension de notre algorithme étalé au chapitre 4 et certaines approches de reconnaissances d'activités citées au chapitre 3.

2.6.1 La classification

La classification [66] est l'opération la plus utilisée de la FD et a pour objectif d'affecter une instance inconnue à une des classes déjà définies [67]. Par exemple, dans le domaine de la reconnaissance de la parole [68], la classification permet d'affecter les signaux de la parole à leurs représentations textuelles correspondantes. Tappert et al. [69] l'ont utilisé pour reconnaître des mots écrits à la main en considérant chaque image (contenant le mot) comme une séquence de colonnes de pixels. Dans le domaine de reconnaissance de gestes, les vidéos contenants des mouvements des mains ou de la tête sont classifiés, en les découpant en plusieurs trames, afin de trouver les actions ou les messages qu'ils essaient de transmettre [70].

Cette opération commence par la division des données en deux ensembles; des données d'apprentissage et des données de tests, qui seront utilisées dans les deux étapes de classification illustrées dans la Figure 2-3 [71].

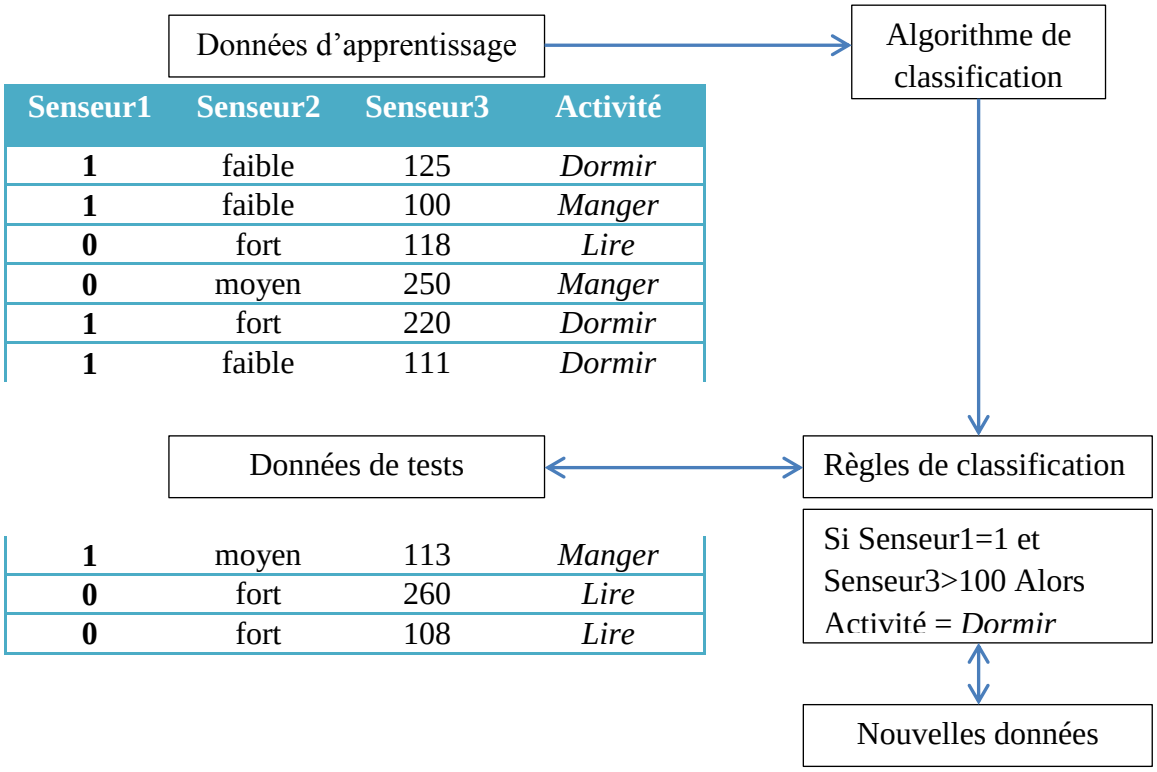


Figure 2-3. Étapes de classification (adaptée de [61]).

- **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" sélectionné à partir des enregistrements de l'entrepôt de données. Parmi les attributs de l'ensemble des données d'apprentissage, un attribut représente la classe dont la valeur est définie par les valeurs des autres attributs. Dans l'exemple présenté à la Figure 2-3, la classe est l'attribut Activité et sa valeur (*Dormir*, *Manger*, *Lire*) est définie par les valeurs des senseurs. Le classificateur créé peut être représenté par des règles de classification, comme celle de la Figure 2-3, 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 les différents attributs x_1,x_2, ...

- **L'étape de classification** : Avant d'utiliser le classificateur créé dans l'étape précédente pour trouver la classe de nouvelles données, sa précision est validée en l'essayant sur les données de tests. Il est important de ne pas tester le classificateur avec des données déjà utilisées dans l'étape d'apprentissage, sinon les résultats seront biaisés. Plusieurs techniques existent pour diviser l'entrepôt de données en données d'apprentissages et données de tests, comme la technique des deux tiers un tiers, "cross-validation", "bootstrap" [72], etc. Dans l'exemple de la Figure 2-3, c'est la technique deux tiers un tiers qui est utilisée en réservant les deux tiers des données à l'apprentissage (l'entrainement) et un tiers aux tests. Quand les résultats des tests sont satisfaisants, le classificateur est ensuite utilisé pour les nouvelles données, sinon l'algorithme de classification ou les données d'apprentissages peuvent être changés.

Le type de classification utilisé dans l'exemple de la Figure 2-3 est dit supervisé parce que la classe est connue à l'avance. Quand la classe n'est pas connue à l'avance, classification non supervisée, la technique de la segmentation, que nous détaillons dans la prochaine section, peut être utilisée pour créer des groupes qui représenteront les classes.

2.6.2 La segmentation

L'opération de segmentation "clustering" [73] permet de grouper les éléments 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 [74]. Elle est utilisée pour créer des classes ou pour compresser les données en

travaillant simplement avec les clusters créés au lieu de tous les éléments qui les composent. Tout comme la classification, la segmentation essaie par la suite d'assigner les nouvelles entrées aux clusters adéquats. La différence majeure 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 en calculant une distance métrique entre les différentes entrées [71].

La segmentation est d'un intérêt particulier puisqu'elle fournit un mécanisme capable de trouver, automatiquement, des structures, dans de grands entrepôts de données, qui seraient autrement difficiles à résumer ou à visualiser [67]. Il existe de nombreuses applications qui montrent l'importance de l'utilisation de cette technique. Par exemple, en l'appliquant sur les historiques de navigation Web, les clusters peuvent indiquer les modes de navigation des différents groupes d'utilisateurs [75]. Pour les données financières, il est intéressant, par exemple, de grouper les stocks qui présentent des tendances similaires après les mouvements de prix [76]. Un autre exemple pourrait être la segmentation des séquences biologiques, comme les protéines ou les acides nucléiques, de telle sorte à ce que les séquences du même groupe aient des propriétés fonctionnelles similaires [77]. Il existe plusieurs méthodes de segmentation qui peuvent être groupées en deux principales catégories, les méthodes hiérarchiques et les méthodes de partitionnement [73].

- **Les méthodes hiérarchiques** : Cette méthode crée une décomposition hiérarchique de l'ensemble des éléments de données. Elle peut être une méthode d'agglomération ou de division selon la forme de décomposition hiérarchique. Les méthodes d'agglomération commencent en considérant chaque objet comme un cluster. Ensuite,

elles fusionnent successivement les clusters qui sont proches l'un de l'autre jusqu'à l'obtention d'un seul cluster représentant le niveau supérieur ou jusqu'à ce qu'une condition d'arrêt soit satisfaite. À l'opposé, les méthodes de division commencent par un seul cluster qui englobe tous les objets. Ensuite, dans chaque itération, les clusters sont divisés en plus petits clusters jusqu'à ce que chaque cluster ne soit plus composé que d'un seul élément ou jusqu'à ce qu'une condition d'arrêt soit satisfaite.

- **Les méthodes de partitionnements** : ayant des entrées de *n* enregistrements, ces méthodes permettent de créer *k* partitions avec $k < n$ sachant que le nombre de clusters *k* est soit définie manuellement à l'avance ou trouvé automatiquement grâce à une fonction d'entropie. Chaque partition créée représente un cluster vérifiant les deux conditions suivantes : 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" [78] que nous utilisons dans notre algorithme de création de modèles d'activités pour compresser les données et trouver des intervalles temporels homogènes. Plus précisément, nous avons utilisé l'algorithme C-means [79] qui, à l'exception du fait qu'il permet à un élément d'appartenir à plusieurs clusters, est identique à l'algorithme le plus connu et le plus utilisé de cette catégorie qui est le k-means. L'explication du C-means aidera à la compréhension de notre algorithme, mais puisque le k-means lui ressemble beaucoup tout en étant plus facile à comprendre, c'est ce dernier que nous allons détailler en expliquant ses différentes étapes présentées à l'algorithme 2-1.

Algorithm K-means (D, K, ϵ)

Entrees: Un ensemble d'éléments D, le nombre de clustre K
Sorties: K clusters

1: t = 0
2: Initialiser K centres de clusters au hasard μ_1^t, μ_2^t, ..., μ_K^t
3: **repeter**
4: t ← t+1
5: $C_j \leftarrow \phi$ pout tous j=1,..., K
6: **Pour chaque** $x_j \in D$
7: $j^* \leftarrow argmin_i\{|x_i - mu_i^t|^2\}$ //assigner elements
8: $C_{j^*} \leftarrow C_{j^*} \cup \{x_j\}$ //au centre le plus proche
9: **Fin Pour**
10: **Pour chaque** i = 1 a K
11: $\mu_i^t \leftarrow \frac{1}{|C_i|}\Sigma_{x_j \in C_i} x_j$
12: **Fin Pour**
13: **jusqu'à** $\Sigma_{i=1}^K |\mu_1^t - \mu_1^{t-1}|^2 \leq \epsilon$
14: **retourner** Tous les clusters

Algorithme 2-1. L'algorithme k-means.

Le K-means commence par choisir *k* centres de clusters au hasard puis il répète les deux opérations suivantes jusqu'à ce que les centres des clusters ne changent plus (dans l'algorithme 2-1, cette condition d'arrêt est formulée mathématiquement dans la ligne 13 comme suit : la somme des distances entre chaque centre et sa valeur dans l'itération précédente est inférieure à la distance maximale ε.) :

- Il affecte chaque élément au cluster possédant le centre le plus proche à l'élément.
- Il calcule le nouveau centre de chaque cluster.

2.6.3 La découverte de règles d'association

La découverte de règles d'association [65] est une tâche assez spéciale de la fouille de données du fait que ses techniques ne possèdent aucune requête spécifique pour fouiller la base de données [67]. De plus, c'est une tâche propre à la fouille de données au contraire d'autres tâches comme la classification et la segmentation qui ont des origines

d’autres domaines comme la théorie de l'estimation [80] ou l’apprentissage automatique [31]. La découverte de règles d'association est utilisée dans divers domaines tels que les réseaux de télécommunications, les achats en ligne, le contrôle des stocks, le marketing et la gestion des risques, etc. Elle a pour objectif d’extraire les corrélations, les associations, les structures informelles et les motifs d'intérêt sachant qu’il n'y a aucune notion universelle pour désigner un motif d’intéressant. Cependant, une des notions très utiles dans l'exploration de données est celle des motifs fréquents.

La découverte de motifs fréquents "frequent pattern mining" [36], que nous utilisons pour créer les modèles d’activités, a été introduite pour la première fois dans le travail de Agrawal et al. [81] dans le domaine de l’analyse du panier de la ménagère "Frequent itemset mining" [82] pour répondre à des questions du genre : quels sont les articles vendus ensemble ? Les algorithmes de ce domaine considèrent un motif comme fréquent si sa fréquence, son nombre d’occurrences, dans la base de données, est supérieur à une fréquence minimale définie par l’utilisateur. Une fois un motif détecté, il peut aider à générer des règles d’associations du genre X→Y qui signifie qu’il est fort probable que l’élément Y apparaîtra après l’apparition de l’élément X. C’est exactement ce genre de règles qui nous ont poussées à chercher les motifs fréquents qui, dans notre cas, seront des séquences de senseurs S_1, S_2, S_3, ... et qui permettront de prédire le prochain senseur, la prochaine action de la personne assistée, après l’activation du dernier senseur. Pour mieux comprendre le fonctionnement de ces algorithmes et les comparer au nôtre, nous détaillerons dans la prochaine section l’algorithme Apriori sur lequel la majorité des autres algorithmes de ce domaine se sont basés. Nous allons aussi détailler

l'algorithme BIDE que nous avons utilisé dans notre première approche expliquée dans le prochain chapitre.

2.6.3.1 L'algorithme Apriori

La majorité des algorithmes du domaine d'extraction des motifs fréquents sont basés sur l'algorithme Apriori [81] qui a été créé à partir des deux propriétés suivantes :

1. Un article est fréquent si sa fréquence est supérieure ou égale à la fréquence minimale précisée par l'utilisateur. Ce dernier est donc appelé à spécifier la fréquence minimale dès le départ.
2. Si un article n'est pas fréquent, alors sa jointure avec un autre article ne le sera pas elle aussi. Cette observation est très importante parce qu'elle permet de gagner un temps considérable en ignorant toutes les combinaisons des éléments non fréquents.

Les différentes étapes de l'algorithme Apriori sont présentées dans l'algorithme 2-2, affiché dans la page suivante.

Apriori commence par créer une liste de tous les éléments fréquents L_1. Ensuite, il répète les deux étapes suivantes jusqu'à ce que la dernière liste créée soit vide :

- Créer toutes les combinaisons possibles C_k à partir des éléments de la dernière liste créée L_k;
- Créer la nouvelle liste L_k en éliminant les combinaisons non fréquentes de C_k.

Algorithm Apriori (T,ϵ)

Entrees: L'ensemble de séquences T, la fréquence minimale ϵ
Sorties: L'ensemble des sous-séquences fréquentes

1: $L_1 \leftarrow \{large1 - itemsets\}$
2: $k \leftarrow 2$
3: **Tant que** $L_{k-1} \neq \phi$
4: $C_k \leftarrow \{a \cup \{b\} | a \in L_{k-1} \wedge b \notin a\} - \{c | \{s | s \sqsubseteq c \wedge |s| = k-1\} \not\sqsubseteq L_{k-1}\}$
5: **Pour** transactions $t \in T$
6: $C_t \leftarrow \{c | c \in C_k \wedge c \sqsubseteq t\}$
7: **Pour** candidats $c \in C_t$
8: $count[c] \leftarrow count[c] + 1$
9: **Fin Pour**
10: **Fin Pour**
11: $L_k \leftarrow \{c | c \in C_k \wedge count[c] \geq \epsilon\}$
12: $k \leftarrow k + 1$
13: **Fin Tant que**
14: **retourner** $\bigcup L_k$

Algorithme 2-2. L'algorithme Apriori.

2.6.3.2 *L'algorithme BIDE*

En se basant sur les propriétés de l'algorithme Apriori, plusieurs autres algorithmes de l'extraction des motifs fréquents ont été créés pour améliorer les résultats ou pour répondre à des problèmes plus spécifiques. L'algorithme FP-growth [83], par exemple, essaie de réduire la mémoire utilisée en stockant la base de données selon un format compressé dans des structures d'arbres spéciales appelées FP-tree. CloSpan [84] essaie de trouver des séquences fréquentes qui ont la particularité d'être fermées. Une séquence est dite fermée, si elle n'est pas contenue dans n'importe quelle autre séquence. Cette propriété est très intéressante pour nous parce que nous souhaitons trouver des activités entières et non pas des parties d'activités. Le problème avec CloSpan est qu'il garde l'ensemble des séquences fermées déjà trouvées pour estimer si les nouvelles séquences fréquentes trouvées sont susceptibles d'être fermées, ce qui encombre la mémoire. Alors, nous nous sommes tournés vers BIDE [34].

BIDE est un algorithme du "Sequential pattern mining" [85] qui essaie de trouver des séquences fréquentes et fermées sans maintenir des séquences candidates dans la mémoire. Il prend comme entrées un ensemble de séquences, comme celui présenté au Tableau 2-1. Chaque séquence peut contenir plusieurs fois le même élément et l'ordre des éléments dans la séquence est partiellement respecté.

Tableau 2-1. Exemple d'entrées de BIDE.

Identificateur séquence	Séquence
1	*C A A B C*
2	*A B C B*
3	*C A B C*
4	*A B B C A*

De la première séquence, on peut constater que l'élément A apparaît deux fois dans cette séquence et que *CB* apparaît aussi même si *B* n'apparaît pas immédiatement après *C*. Par contre, *AC* n'apparaît pas dans cette séquence. Dans notre cas, il faut être capable d'indiquer, sans ambiguïté, le prochain senseur à la personne assistée. Alors, dans notre approche précédente, nous avions personnalisé BIDE pour que l'ordre soit parfaitement respecté. Dans cette approche, *CB* n'apparaît donc pas dans la première séquence puisque *B* n'apparaît pas immédiatement après *C*. Cet ordre strict est très important et nous nous sommes basés dessus pour élaborer notre nouvel algorithme de création de modèles d'activités. Il permet de réduire la complexité de la détection des motifs fréquents puisqu'il en découle une propriété qui stipule que pour qu'une séquence soit fréquente il faut que tous les couples de senseurs adjacents, qui la composent, soient fréquents. Cette

propriété permet à notre algorithme de travailler sur des séquences moins longues en coupant la séquence entre chaque couple de senseurs non fréquent.

BIDE s'effectue en deux étapes. La première étape, détaillée dans l'algorithme 2-3, essaie de trouver toutes les séquences fréquentes.

Algorithm Enumération des séquences fréquentes (SDB, min_sup, FS)

Entrees: La BD de séquences, la fréquence minimale min_sup
Sorties: L'ensemble des séquences fréquentes FS

1: FS = ϕ
2: **Appeler** Séquences_Fréquentes(SDB, ϕ, min_sup, FS)
3: **retourner** FS

Séquences_Fréquentes(S_p_SDB, S_p, min_sup, FS)
Entrees: Une BD projetéeS_p_SDB, une séquence de prfixe S_p, la fréquence minimale min_sup
Sorties: L'ensemble actuel des séquences fréquentes FS

4: **Si** S_p n'est pas vide **Alors**
5: FS = FS $\cup$ S_p
6: **Fin Si**
7: LF_S_p = les items locals fréquents(S_p_SDB, S_p, min_sup)
8: **Si** LF_S_p est vide **Alors**
9: **retourner**
10: **Fin Si**
11: **Pour chaque** item local fréquent i
12: $S_p{}^i = < S_p, i >$
13: $S_p{}^i$_SDB = pseudo BD projetée($S_p{}^i$, S_p, min_sup)
14: **Appeler** Séquences_Fréquentes($S_p{}^i$_SDB, $S_p{}^i$, min_sup, FS)
15: **Fin Pour**

Algorithme 2-3. Énumération de toutes les séquences fréquentes.

Dans cette étape, BIDE crée un arbre qui ne garde que les séquences fréquentes. L'arbre est créé en affectant ϕ 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 2-4 schématise l'arbre résultant de l'application de cette étape sur le Tableau 2-1 avec une fréquence minimale égale à 2.

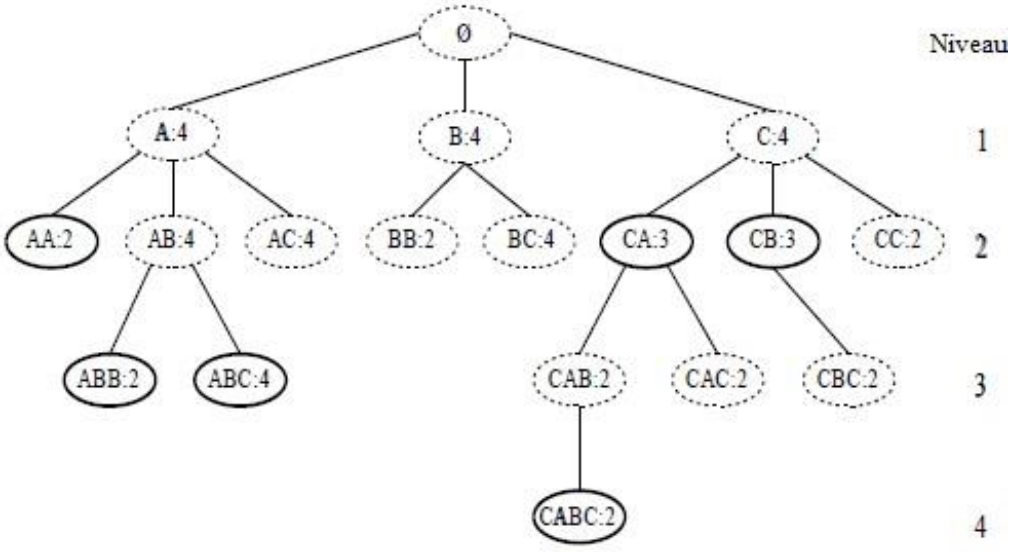


Figure 2-4. Arbre contenant les séquences fréquentes de l'exemple (tirée de [34]).

La deuxième étape de BIDE, détaillée dans l'algorithme 2-4, affiché dans la page suivante, 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 de les comparer à chaque nouvelle séquence trouvée, BIDE utilise la technique d'extension bidirectionnelle qui stipule que si une séquence $S = e_1, e_2,..., e_n$ est non fermée alors une nouvelle séquence S' doit forcément exister ayant la même fréquence que S et possédant un élément de plus e'. Trois cas sont possibles, e' peut être au début : $S' = e', e_1, e_2,..., e_n$, au milieu : $S' = e_1, e_2,..., e_k, e', e_{k+1}, e_n$ ou bien à la fin : $S' = e', e_1, e_2,..., e_n, e'$. Le test de fermeture est effectué donc en cherchant si le e' n'existe pas dans les trois cas. Si on revient à l'exemple, les séquences fréquentes et fermées sont celles encerclées par des traits pleins dans la Figure 2-4.

Algorithm BIDE (SDB, min_sup, FS)

Entrees: La BD de séquences SDB, la fréquence minimale min_sup
Sorties: L'ensemble des séquences fréquentes et fermées FCS

1: FCS = ϕ
2: F1 = Les fréquentes 1-séquences(SDB,min_sup)
3: **Pour chaque** 1-séquence f1 $\in$ F1
4: SDB^{f1} = pseudo BD projetée(SDB)
5: **Fin Pour**
6: **Pour chaque** f1 $\in$ F1
7: **Si** !BackScan(f1,SDB^{f1}) **Alors**
8: BEI = vérification d'extension backward(f1, SDB^{f1})
9: **Appeler** bide(SDB^{f1}, f1, min_sup, BEI, FCS)
10: **Fin Si**
11: **Fin Pour**
12: **retourner** FCS

bide(S_p_SDB, S_p, min_sup, BEI, FCS)
Entrees: Une BD projetéeS_p_SDB, une séquence de prfixe S_p,
la fréquence minimale min_sup, le nombre de backward BEI
Sorties: L'ensemble actuel des séquences fréquentes e tfermées FCS

13: LFI = les items locals fréquents(S_p_SDB)
14: LFI = $|\{ z \in LFI \mid z.sup^{SDB}(S_p)\}|$
15: **Si** ((BEI $\neq$ FEI)==0) **Alors**
16: FCS = FCS $\cup$ $\{S_p\}$
17: **Fin Si**
18: **Pour chaque** i $\in$ LFI
19: ${S_p}^i = < S_p, i >$
20: $SDB^{S_p i}$ = pseudo BD projetée(S_p_$SDB, {S_p}^i$)
21: **Fin Pour**
22: **Pour chaque** i $\in$ LFI
23: **Si** !BackScan(${S_p}^i$,$SDB^{S_p i}$) **Alors**
24: BEI = vérification d'extension backward(${S_p}^i$,$SDB^{S_p i}$)
25: **Appeler** bide($SDB^{S_p i}$, ${S_p}^i$, min_sup, BEI, FCS)
26: **Fin Si**
27: **Fin Pour**

Algorithme 2-4. L'algorithme BIDE.

2.7 Conclusion

Durant ce chapitre, nous avons essayé de donner une vue générale sur la fouille de données temporelles et ses techniques largement utilisées dans la reconnaissance d'activités. La FDT est définie 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. Les données temporelles, que ça soit le temps de transaction, le temps valide, les séquences ou les séries temporelles, posent de grands défis à la fouille de données qui pèche à gérer les dimensions ajoutées par les relations temporelles. Des techniques ont donc été modifiées et d'autres ont vu le jour pour créer un sous-domaine appelé fouille de données temporelles que nous utilisons dans notre nouvelle approche.

Au cours de chapitre, nous avons aussi expliqué plus en détail trois techniques de la fouille de données temporelles. Nous avons commencé par la classification qui est utilisée, dans le domaine de la reconnaissance d'activités, par plusieurs approches [14], [15], [86] dont celles citées dans le prochain chapitre. La classification vise à affecter une nouvelle donnée à la classe adéquate en créant un classificateur à partir des données d'apprentissage. Nous avons détaillé aussi la segmentation qui permet de créer des clusters dont les éléments sont le plus homogènes possible et très hétérogènes des éléments des autres clusters. Puisque nous avons utilisé le C-means, un algorithme de cette technique, nous avons préféré expliquer le mode de fonctionnement du k-means qui est l'algorithme le plus utilisé de la segmentation et qui ressemble beaucoup au C-means tout en étant plus simple à comprendre. Nous avons terminé ce chapitre par l'explication de la technique de l'extraction des modèles fréquents en détaillant Apriori, l'algorithme qui a défini les propriétés utilisées par la majorité des algorithmes de cette catégorie, et en détaillant aussi BIDE que nous avons utilisé dans notre première approche citée dans le chapitre suivant.

Chapitre 3

3 État de l'art sur la reconnaissance d'activités

3.1 Introduction

Reconnaître l'activité est un acte inné chez l'être humain. En effet, 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 du matin, 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 ambiant doit posséder des modèles d'activités et doit être capable d'en créer des nouveaux. La reconnaissance d'activités, pour l'agent ambiant, revient donc à sélectionner l'activité qui explique les récentes observations parmi ces modèles d'activités.

Le type d'activités que nous désirons reconnaître, le type de collaboration de la personne observée ainsi que les outils utilisés pour l'observation influencent directement

l'approche de reconnaissance d'activités. Pour mieux situer notre approche, nous allons donc commencer ce chapitre par l'explication de ces trois éléments, ensuite, différentes approches de reconnaissance d'activités seront détaillées.

3.2 Les activités de la vie quotidienne (AVQ)

La notion d'AVQ a été définie en 1963 par Katz *et al.* [24] 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 AVQs 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 âgées 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 AVQs 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 adéquat. Au fil des ans et avec l'évolution du domaine de la santé, d'autres tests du niveau d'autonomie ont vu le jour [12], [87]; mais, ils restent tous basés sur les AVQs qui sont, de nos jours, classés en trois différents types [88]:

- Les activités de vie quotidienne basiques (AVQB) : qui est l'ensemble des activités fondamentales et obligatoires pour combler les besoins primaires de la

personne. Elles 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 aidants, 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 est une personne autonome qui peut vivre toute seule chez elle. Les AVQI sont plus complexes et se composent de plus d'étapes 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 ses achats par Internet sur un site qui peut modifier son design et la position des liens.

Dans le cadre de notre étude, nous visons à reconnaître les AVQB et les AVQI parce que la non-réussite d'exécution des AVQA n'empêcherait pas l'acteur de vivre d'une façon autonome chez lui. Dans cette thèse, le terme AVQ est utilisé exclusivement pour généraliser les AVQB et les AVQI.

3.3 Les types de reconnaissance d'activités

Après avoir précisé les types d'activités que nous désirons reconnaître dans la section précédente, dans cette section, nous allons préciser le type de collaboration de l'agent-acteur avec l'agent observateur. D'après Geib et al. [89], il existe trois types de

reconnaissance d'activités basée sur cette collaboration : la reconnaissance communicative, la reconnaissance d'opposition et la reconnaissance à l'insu.

- La reconnaissance d'activités 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'agent-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 d'activités (LIARA) à l'Université du Québec à Chicoutimi, où nous répétions une action plusieurs fois jusqu'à ce que nous soyons sûrs que le système l'a détectée.
- La reconnaissance d'activités d'opposition : comme dans le premier type de reconnaissance, l'acteur est au courant qu'il est observé, mais cette fois-ci il n'est pas consentant à ce que l'agent observateur reconnaisse 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 stratégies militaires et jeux vidéo pour dissimuler la stratégie d'attaque par exemple.

- La reconnaissance d'activités 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.

Pour des raisons d'éthique, le patient doit être informé qu'il est observé, mais cela ne veut pas dire que la reconnaissance à l'insu est exclue de nos choix. N'oublions pas que le premier objectif de l'assistance technologique est de permettre à l'agent-acteur de vivre d'une façon normale chez lui sans lui demander d'adapter ses actions pour influencer les résultats de la reconnaissance. Donc, bien que l'agent-acteur soit avisé qu'il est observé, les capteurs utilisés doivent être assez transparents pour l'aider à se concentrer sur l'accomplissement de ses activités d'une façon tout à fait normale. Le choix des capteurs ne rend pas seulement la reconnaissance à l'insu possible, mais il spécifie aussi le type de reconnaissance d'activités qui peut être : la reconnaissance d'activité basée sur la vision "Vision-based activity recognition" [17] ou la reconnaissance d'activité basée sur les senseurs "Sensor-based activity recognition" [90].

3.4 La reconnaissance d'activités basée sur la vision

Les approches de reconnaissance d'activités basées sur la vision sont caractérisées par l'utilisation de caméras pour détecter les changements de comportement de l'agent-acteur ou de son environnement. En effet, les caméras fournissent des données, sous forme de vidéos et 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 [91] pour l'analyse des observations visuelles afin de reconnaître 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'activités basées sur la vision évitaient de reconnaître 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'activités. Le travail de Duong et al. [92] en est un bon exemple. Dans ce travail, plusieurs caméras installées aux coins de la chambre, observent l'agent-acteur effectuant ses 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 micro-onde*, etc.). Aucune reconnaissance des actions primitives n'est effectuée, mais la liste des cellules visitées est utilisée comme entrée pour la reconnaissance d'activités.

Avec le développement des techniques utilisées et l'émergence de la fouille de données, les nouvelles approches visent la détection directe des actions primitives à partir des données enregistrées par les caméras. La figure 3-1 illustre les principales étapes d'un système générique de reconnaissance des actions.

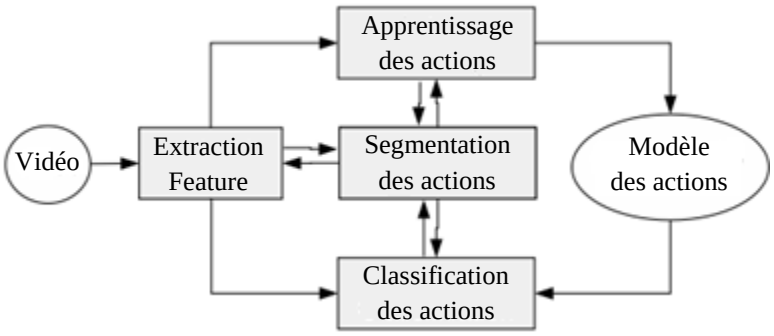


Figure 3-1. Système de reconnaissance des actions (adaptée de [92]).

Le système commence par l'extraction des caractéristiques discriminantes "Feature" qui permettent la différenciation des actions. Les deux étapes de la technique de classification, expliquée au chapitre précédent, sont appliquées sur les caractéristiques détectées. L'étape d'apprentissage permet de créer un classificateur en trouvant les caractéristiques propres à chaque action. La dernière étape consiste à utiliser ce classificateur sur les caractéristiques des nouvelles données pour trouver les actions qu'elles représentent. Le travail de Spriggs et al., détaillé dans la prochaine section, illustre bien ce processus.

3.4.1 Approche de Spriggs et al.

Dans le travail de Spriggs et al. [17], 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'activités.

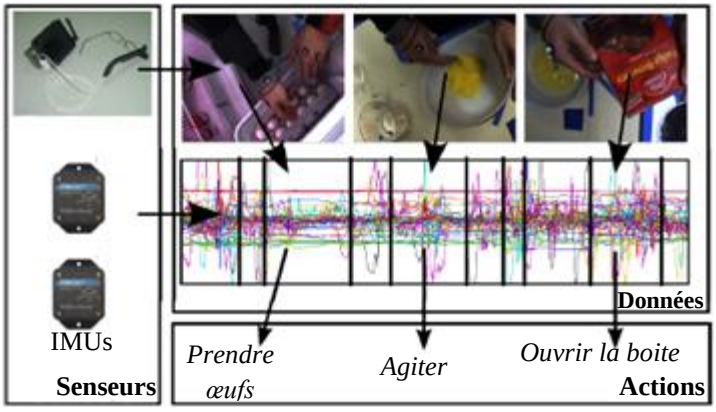


Figure 3-2. Segmentation temporelle sur vidéo et signaux (tirée de [17])

La Figure 3-2 montre les senseurs utilisés ainsi que le résultat de la segmentation temporelle sur les deux différents types de données. La caméra offre une vision à la première personne; ce qui aide à comprendre les intentions de l'agent-acteur. 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 divisé en blocs de 4x4. Ils sont ensuite transformés en un vecteur à 512 dimensions pour chaque trame vidéo. Pour réduire la dimensionnalité, une analyse en composantes principales, présentée dans le travail de Barbic et al. [93], est effectuée 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) [94]. Le GMM est une somme pondérée de M densités des composantes gaussiennes donnée par la formule suivante :

$$p(x \mid \lambda) = \sum_{i=1}^{M} w_i \, g(x \mid \mu_i, \sum_i)$$

où x est le vecteur de dimension 32, w_i, i= 1, ..., M, sont les poids du mélange, et g(x $|\mu_i, \sum_i i = 1, \ldots, M,)$, sont les densités des composantes gaussiennes. La densité de chaque composante est une fonction à 32-variables gaussiennes de la forme :

$$g(x\,|\mu_i, \sum_i) = \frac{1}{(2\pi)^{\frac{D}{2}}\,|\sum_i|^{\frac{1}{2}}}\; exp\{-\frac{1}{2}(x-\mu_i)'\,\sum_i{}^{-1}(x-\mu_i)\}$$

avec μ_i est le vecteur moyen, $\sum_i$ est la matrice de covariance et le mélange des poids satisfaisant 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 reconnaître l'activité entamée et pouvoir prédire la prochaine action. Pour la classification, les auteurs ont utilisé la méthode des k proches voisins "k -Nearest Neighbor" [95]. 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 :

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

où $p = (p_i, \ldots, p_n)$ et $q = (q_i, \ldots, q_n)$ deux points dans l'espace euclidien de dimension n.

Pour valider leur approche, les auteurs de cette étude 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 3-2 montre un exemple du résultat de cette comparaison.

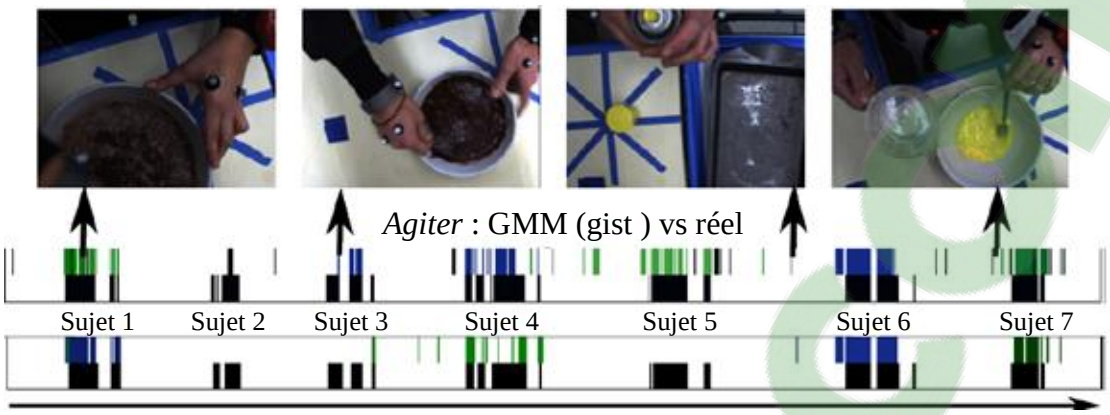


Figure 3-3. Comparaison des segments automatiques et manuels (tirée de [17])

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%.

3.4.2 Évaluation de l'approche de Spriggs et al.

Ce travail 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 de futures recherches qui doivent répondre à plusieurs questions. D'abord, quelle est la raison de ce taux de réussite pas assez élevé (70%) ? On peut se demander aussi, comment une trame va être classifiée si l'agent-acteur change les outils utilisés pour effectuer l'action ? En effet, en effectuant l'action *Agiter*, si l'agent-acteur utilise un bol de différente forme et de différentes couleurs, la trame sera très différente de celles classifiées pour la même action. Une autre question qui se pose concerne le temps de réponse de cette approche, sachant que le traitement des séquences vidéo est une lourde tâche pour des applications en temps réel. D'ailleurs, les auteurs ne l'ont pas essayée dans le cadre d'une assistance instantanée.

3.5 La reconnaissance d'activités basées sur les senseurs

L'utilisation des caméras pour la reconnaissance d'activités 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 et vidéos, 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'activités 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.

3.5.1 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 ses 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 localisation 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 [96].

Les senseurs portables peuvent être de différents types; des accéléromètres, des gyroscopes, des magnétomètres, des étiquettes RFID, etc. ou des téléphones portables qui incluent différents senseurs. Les données qu'ils émettent fournissent principalement des informations sur la position et les mouvements de l'agent-acteur. Ces informations sont

utilisées pour reconnaître des mouvements basiques tels que *marcher*, *courir*, etc. Cela n'empêche qu'elles peuvent être très utiles pour la reconnaissance de certaines AVQs comme : *se brosser les dents*, *écrire*, *utiliser un ordinateur*, etc. La Figure 3-4, prise du travail de Ravi et *al* [7], montre comment un senseur accéléromètre triaxial placé sur le corps peut aider à la reconnaissance de certaines activités.

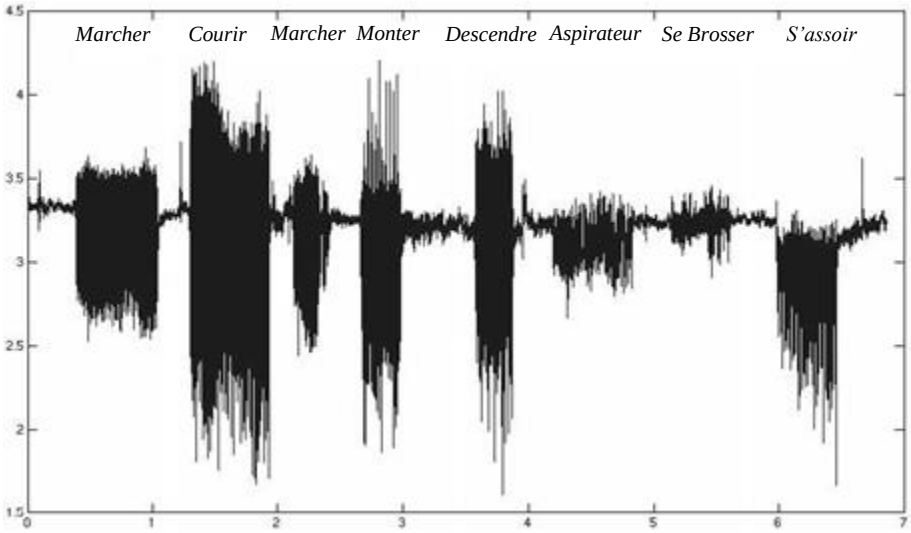


Figure 3-4. Reconnaissance d'activités par un senseur porté (tirée de [7])

Plusieurs approches de reconnaissance d'activités à partir de senseurs portables ont été proposées utilisant différents senseurs. Dans Vigilante, le travail de Lara et al. [97], une application, en temps réel, pour les téléphones portables a été proposée. Le senseur Zephyr's BioHarness BT, qui s'attache sur de la poitrine, a été utilisé pour mesurer l'accélération et les signaux physiologiques tels que la fréquence cardiaque, le rythme respiratoire, l'amplitude de l'onde respiratoire et la température de la peau, etc. Des caractéristiques sur la fréquence et le temps ont été extraites à partir des signaux d'accélération alors qu'une régression polynomiale a été appliquée sur les signaux psychologiques. Ensuite pour la classification, l'algorithme C4.5 [98] a été utilisé pour l'apprentissage et la reconnaissance d'activités avec une précision de 92,6%. Des

utilisateurs avec des caractéristiques différentes ont participé à la phase d'apprentissage et de tests assurant la flexibilité de reconnaître les activités des nouveaux utilisateurs sans avoir à refaire l'étape d'apprentissage.

Maurer et al. [99] ont aussi proposé un système de reconnaissance d'activités appelé eWatch qui utilise un accéléromètre, un capteur de lumière, un thermomètre, un microphone et un microcontrôleur placés dans un dispositif qui peut être porté comme une montre de sport. En utilisant l'algorithme C4.5, ils ont obtenu 92,5% de précision pour la reconnaissance de six activités : *s'assoir, se mettre debout, marcher, monter les escaliers, descendre les escaliers et courir*.

En général, la reconnaissance d'activités basée sur les senseurs portables souffre de deux inconvénients majeurs. Premièrement, la plupart des senseurs portables ne sont pas applicables dans la vie de tous les jours à cause des problèmes techniques causés par leur taille, la durée de vie de la batterie ou la facilité d'utilisation, en plus de l'acceptabilité ou la volonté de l'agent-acteur à les porter. Le second inconvénient réside dans la complexité de la plupart des AVQs 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'activités basée sur les senseurs portables est souvent combinée avec celle basée sur les objets que nous allons détailler dans la prochaine section.

3.5.2 Les approches basées sur les objets

Dans ces approches, au lieu d'utiliser des caméras ou des senseurs portables pour offrir à l'agent observateur une vue semblable à celle de l'agent-acteur, ce qui nécessite par la suite des analyses pour reconnaître les objets avec lesquels il interagit, les senseurs

sont posés directement sur les objets d'une façon totalement transparente à l'agent-acteur. C'est la comparaison des mesures successives des senseurs posés sur les objets qui indique leur utilisation par l'agent-acteur. Les senseurs utilisés dans ces approches peuvent être de différents types; 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 estimer l'emplacement d'objets comme une tasse ou un bol dans la maison, des capteurs de température, d'humidité ou à flotteur pour détecter si le four, la douche ou les toilettes sont utilisés, etc [100].

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 senseurs de type ad hoc [101]. Une étude approfondie doit normalement être faite avant le choix des senseurs et leur emplacement.

Plusieurs travaux ont essayé de répondre à la problématique de reconnaissance d'activités en se basant sur les mesures envoyées par les différents senseurs. Le grand volume de données enregistrées dans la base de données complique grandement cette tâche et rend l'utilisation des techniques de la fouille de données presque indispensable. Puisque l'approche que nous proposons dans cette thèse appartient à cette catégorie, nous allons détailler quelques approches intéressantes, dont notre première approche réalisée au cours de ma maitrise en informatique (Moutacalli et al. [86]).

3.5.2.1 Approche de Suryadevara et al.

Les approches de reconnaissances d'activités peuvent aussi être catégorisées selon l'étape d'apprentissage. Dans cette approche [15], l'occupant de la maison note, manuellement dans une feuille de temps, les activités avant de les commencer. Ce type d'approches est dit supervisé. Les senseurs utilisés ainsi que leurs durées sont ensuite utilisés pour composer la dernière activité notée. Le système de senseurs utilisé dans cette approche est composé de deux groupes. Les objets électriques comme le micro-onde, la télévision, la bouilloire, le grille-pain et le chauffage, etc., sont reliés à un dispositif électrique qui permet la détection des objets actifs et leur durée d'activation. Les objets non électriques comme le lit, les chaises et le canapé, etc., sont surveillés à l'aide d'un senseur de force (Flexi force) très mince, souple et non envahissant. Les signaux provenant des senseurs de force sont intégrés et reliés à travers une communication radio par l'intermédiaire des modules XBee. Les différentes étapes de cette approche sont schématisées à la Figure 3-5.

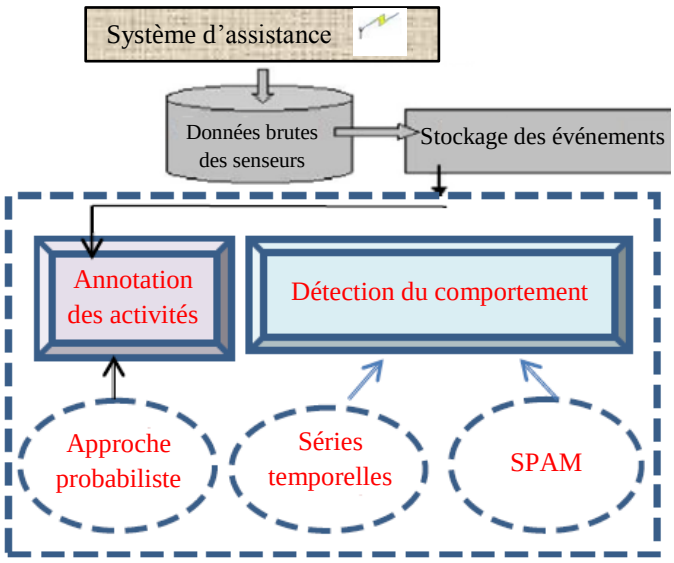


Figure 3-5. Les différentes étapes de l'approche de Suryadevara et al (tirée de [15]).

Après l'enregistrement des données envoyées par les senseurs, toutes les activités sont détectées puisque l'occupant spécifie le début et la fin de chaque activité. Ensuite, l'étape de reconnaissance d'activités se déroule selon les étapes suivantes :

- Les activités sont annotées en appliquant la méthode probabiliste conditionnelle suivante : la meilleure identification de l'activité pour le flux d'événements de senseurs est donnée par l'estimation de la probabilité conditionnelle maximale qui est la fréquence relative d'un senseur *(t)* dans le flux de senseurs appartenant à l'activité *(c)*

$$P(t|c) = \frac{N_{ct} + 1}{\sum_{t' \in V}(N_{ct'} + 1)} = \frac{N_{ct} + 1}{\sum_{t' \in V} N_{ct'} + K}$$

  où N_{ct} est le nombre d'apparitions du senseur *t* dans l'activité *c*, *V* est l'ensemble des ids des senseurs et *K*=|*V*| est le nombre d'ids unique dans *V*.
- Attribuer à chaque activité une lettre bien définie, par exemple *Préparer thé* = *A*, *Préparer café* = *B* et *Préparer Toast* = *C*.
- Selon le senseur activé, considérer les activités qui le comportent. Par exemple, si la bouilloire est activée, alors il faut considérer les activités *A* et *B*, si le réfrigérateur est activé, alors il faut considérer *A*, *B* et *C* et si le thé est activé, *A* est considérée.
- Appliquer la dernière formule pour trouver l'activité qui a la probabilité maximale avec le flux de senseurs activés et les activités considérées. Pour l'exemple, *P(A,B,C,C|TS)* = 0,0277 et l'activité reconnue sera *Préparer Toast*.

L'une des particularités de cette approche, c'est qu'elle permet l'évaluation de l'état de santé de l'occupant de la maison en comparant sa moyenne journalière d'utilisation des différents dispositifs. Pour ce propos, les auteurs ont introduit deux fonctions, β_1 qui

calcule la durée inactive des dispositifs et β_2 qui calcule l'excès d'utilisation des dispositifs. Les formules de calcul des deux fonctions sont les suivantes :

$$\beta_1 = 1 - \frac{t}{T}$$

$$\beta_2 = 1 + (1 - \frac{T_a}{T_n})$$

où t est la durée d'inactivité de tous les dispositifs, T est la durée maximale pendant laquelle aucun dispositif n'a été activé, T_a est la durée réelle de l'utilisation d'un dispositif et T_n est la durée maximale d'utilisation de ce dispositif dans des circonstances normales.

Une fois l'activité entamée est reconnue, une étape de détection d'erreurs est effectuée. La détection d'erreurs se fait selon deux critères : la durée de l'activité et le degré de ressemblance entre les senseurs détectés et ceux prédits avec l'algorithme SPAM [102].

Pour estimer si la durée actuelle d'une activité est normale, elle est comparée à la durée prédite, de cette activité, en utilisant les séries temporelles. Ce processus est effectué en trois étapes :

- Créer, pour chaque activité, une série temporelle composée de ses durées pour chaque jour, en utilisant une saisonnalité égale à 7 afin de ressortir les particularités de chaque jour de la semaine.
- Appliquer récursivement un double lissage exponentiel aux séries temporelles pour bien gérer la saisonnalité et extraire la tendance :

$$T_t = \delta(L_t - L_{t-1}) + (1 - \delta)T_{t-1}$$

$$L_t = \alpha(X_t - S_{t-S}) + (1 - \alpha)(L_{t-1} + T_{t-1})$$

$$S_t = \gamma(X_t - L_t) + (1 - \gamma)S_{t-S}$$

où T_t est la tendance, L_t est la pente locale de saisonnalité, S_t est le facteur saisonnier, X_t est l'observation en temps réel, $s = 7$ est la saisonnalité et α, γ et δ sont les paramètres du lissage trouvés en minimisant les erreurs par la méthode des moyennes carrées.

- Prédire la durée de l'activité en extrapolant la tendance saisonnière obtenue par la dernière formule et en appliquant la formule suivante :

$$F_{t+m} = L_t + T_{tm} + S_{t-s+m}$$

où *m* est la période de prédiction requise.

En ce qui concerne la détection d'erreurs basée sur les senseurs détectés, SPAM construit un arbre de séquences lexicographique propre au jour de la semaine et le temps courant. Chaque branche de l'arbre est considérée comme une séquence mère dont un nouveau senseur peut être ajouté à sa fin. Il utilise ensuite les propriétés d'Apriori, expliqué dans le chapitre précédent, afin de réduire l'espace de recherche pour la génération des ensembles des motifs fréquents. De cette façon, SPAM est capable de prédire la séquence de senseurs possible pour une journée à un temps donné. La comparaison de cette séquence avec les récents senseurs détectés permet la détection d'erreurs.

Pour valider leur approche, les auteurs ont observé une personne pendant 8 semaines en essayant de reconnaître 5 activités : *Dormir*, *Diner*, *Utiliser toilette*, *Relaxer* et *Regarder télévision*. Les résultats obtenus sont satisfaisants quoique l'exemple qu'ils ont fourni dans leur article indique que pour l'activation de la bouilloire, du réfrigérateur et le thé, c'est *Préparer Toast* qui a été reconnue alors qu'il est clair que c'est *Préparer*

thé qui devrait être reconnue. Sinon, ils n'ont proposé aucune solution aux problèmes généraux des méthodes supervisées qui pèchent à reconnaître les nouvelles activités ou celles modifiées. Le problème du nombre d'hypothèses n'a pas été abordé non plus. Si leur temps d'exécution est correct pour les cinq activités qu'ils essaient de reconnaître, il ne le sera certainement pas quand toutes les activités seront considérées, surtout que leur approche comporte trop de calcul pour une application en temps réel.

3.5.2.2 Approche de Jakkula et Cook

Le travail présenté par Jakkula et Cook [16] exploite les techniques de la fouille de données temporelles pour prédire les prochaines activités qui seront effectuées et pour détecter des anomalies. Leur approche est une approche non supervisée qui travaille directement sur les données envoyées par les senseurs. L'analyse de ces données permet l'extraction d'importantes relations entre les activités de l'agent-acteur. Les relations sont de la forme, l'activité *allumer télé* s'effectue après l'activité *s'assoir sur le divan*. Ce genre de relation permet donc de prédire l'activité *allumer télé* après la détection de l'activité *s'assoir sur le divan*. De plus, si l'activité *allumer télé* est directement détectée, nous pouvons conclure qu'il y a eu une erreur d'exécution parce que l'activité s'assoir sur le divan devait la précéder.

Cette approche est basée sur les différentes relations temporelles définies par Allen [103] et se déroule en plusieurs étapes :

- **L'étape de transformation** : Dans cette étape, les données envoyées par les senseurs, qui sont sous la forme présentée dans le a Tableau 3-1, sont transformées pour créer des intervalles temporels pour chaque activité, comme

montré dans le Tableau 3-2. Les bornes des intervalles sont déterminées par le changement d'état (ON ↔ OFF) des senseurs.

Tableau 3-1. Exemple de données envoyées par les senseurs.

Temps réel	État du senseur	id du senseur
3/3/2003 11:18:00 AM	OFF	E16
3/3/2003 11:23:00 AM	ON	G12
3/3/2003 11:24:00 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

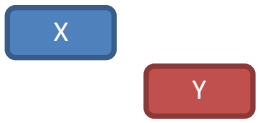
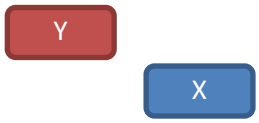
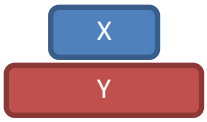
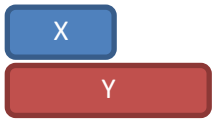
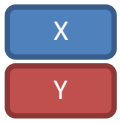
Tableau 3-2. Les différents intervalles temporels créés.

Date	id du senseur	Temps de début	Temps de fin
3/3/2003	G11	01:44:00	01:48:00
3/3/2003	G19	02:57:00	01:48:00
3/3/2003	G13	04:06:00	01:48:00
3/3/2003	G19	04:43:00	01:48:00
3/3/2003	H9	06:04:00	06:05:00
3/3/2003	P1	10:55:00	17:28:00
3/3/2003	E16	11:18:00	11:34:00
3/3/2003	G12	11:23:00	11:24:00

Sachant que le senseur Gl1 est associé à la télé, la première ligne est interprétée comme : la télé a été allumée le 03/02/2003 de 01h44 jusqu'à 01h48.

- **L'étape de recherche de relations** : durant cette étape, la comparaison des bornes des deux intervalles temporels décide du type de relation existante entre les deux activités. Le Tableau 3-3 montre quelques relations des treize relations temporelles proposées par Allen, ainsi que les formules mathématiques qui les définissent.

Tableau 3-3. Quelques relations temporelles d'Allen.

Relations temporelles	Représentation	Contraintes des intervalles
X avant Y		Début(X) < Début(Y) Et Fin(X) < Début(Y)
X après Y		Début(X) > Début(Y) Et Fin(Y) < Début(X)
X durant Y		Début(X) > Début(Y) Et Fin(X) < Fin(Y)
X commence Y		Début(X) = Début(Y) Et Fin(X) ≠ Fin(Y)
X égale Y		Début(X) = Début(Y) Et Fin(X) = Fin(Y)

- **L'étape de découverte des relations fréquentes** : à l'étape précédente, un très grand nombre de relations est généré, beaucoup trop pour les utiliser toutes d'une façon efficace pour la prédiction. De plus, les relations doivent résumer les habitudes de l'agent-acteur. Donc, seules les relations qui reviennent assez souvent doivent être sélectionnées. Les auteurs ont choisi d'implémenter leur version de l'algorithme de découverte de motifs fréquents Apriori [81], détaillé au chapitre 2, pour sélectionner les relations fréquentes. L'application de cet algorithme ne sert pas juste à sélectionner les relations les plus fréquentes, mais elle permet aussi de calculer les fréquences des relations. Ces fréquences peuvent être transformées pour désigner la probabilité qu'un membre d'une relation se produise après la détection de l'autre membre.

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

(1) $P(X|Y) =$

$$|après(Y,X)| + |durant(Y,X)| + |chevauchéPar(Y,X)| + |rencontréPar(Y,X)| + |commence(Y,X)| + |débutéPar(Y,X)| + |égale(Y,X)|/|Y|$$

Quand plusieurs activités sont détectées, le calcul de la probabilité devient un peu plus compliqué. La formule (2) montre ce calcul quand deux activités sont détectées :

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

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

L'interprétation de cette probabilité est très simple et peut facilement aider à la prédiction de la prochaine activité ou détecter une erreur d'exécution. Effectivement, si cette probabilité est proche de 1, cela veut dire que l'activité X a une forte probabilité qu'elle soit la prochaine à s'effectuer. Sinon, si cette probabilité est proche de 0 et que nous détectons que l'activité X s'est effectuée après Y, nous pouvons dire qu'il y avait erreur d'exécution de la part de l'agent-acteur.

L'approche de Jakkula et Cook est une approche très intéressante qui a donné des résultats satisfaisants. Néanmoins, elle a quelques limitations. Par exemple, 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 activités,

sans exception, avec les activités détectées. Donc, ce que nous pouvons leur reprocher, c'est qu'ils n'utilisent pas de contraintes capables de réduire le nombre d'hypothèses. Le fait de ne pas utiliser de contraintes provoque un autre problème : avec les mêmes activités détectées, nous aurons toujours la même activité qui sera prédite, alors que ce n'est évidemment pas toujours le cas.

3.5.2.3 Approche de Ordonez et al.

La phase d'apprentissage permet aussi d'organiser les approches de reconnaissance d'activités en des approches hors-ligne et des approches en ligne. Les approches hors-ligne, comme toutes les approches précédentes, divisent les données enregistrées en données d'apprentissage et données de tests. Elles créent un classificateur qui sera utilisé sur les nouvelles données enregistrées. Par contre, les approches en ligne travaillent directement sur le flux de données produit par les senseurs. L'avantage principal de ces approches, c'est que le classificateur est créé puis constamment modifié avec le flux des données, ce qui lui permet de considérer les changements d'habitudes de la personne assistée. Le travail d'Ordonez et al. [8], est un bon exemple de ces approches et donne une idée sur leur mode de fonctionnement.

La base de données utilisée dans ce travail est générée par un système de senseurs booléens installé dans une maison intelligente de trois chambres pour observer une personne effectuant sept activités : *Quitter maison*, *Utiliser toilette*, *Prendre douche*, *dormir*, *Prendre petit déjeuner*, *Diner* et *Boire*. Pour obtenir un format temporel adéquat, les données des senseurs ont été segmentées en intervalles temporels de longueur constante, comme montrés à la Figure 3-6.

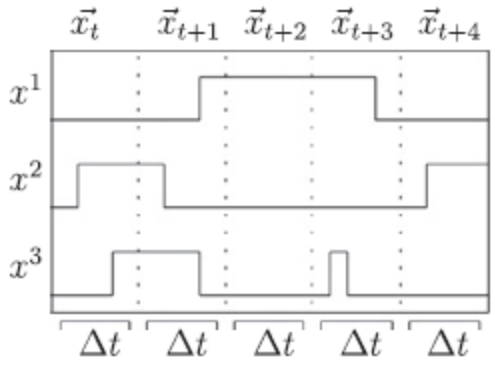


Figure 3-6. Création d'intervalles temporels de longueur constante (tirée de [8]).

La durée des intervalles temporels est Δt et le vecteur des observations est $\overrightarrow{x_t} = x_t^1, x_t^2, \dots, x_t^{N-1}, x_t^N$ où x_t^i indique si le senseur *i* a été activé au moins une fois lors de la période $t+\Delta t$, donc $x_t^i \in \{0, 1\}$. La classe correspondante à un intervalle temporel, qui se déroule au temps *t*, est notée y_t.

Pour préserver les relations temporelles entre les intervalles, l'activité de l'intervalle précédent est ajoutée dans l'identification d'un intervalle. Pour résumer, les données utilisées dans la prochaine étape de classification sont composées de plusieurs instances. Chaque instance I_t est composée de : le vecteur d'observation à l'instant *t*, l'activité correspondante à l'intervalle précédent et l'activité de l'intervalle à l'instant *t*, $I_t = (x_t^1, x_t^2, \dots, x_t^{N-1}, x_t^N, y_{t-1}, y_t)$.

Pour l'étape de classification, les auteurs ont utilisé deux classificateurs de la famille eClass "evolving classifier" [104] qui sont eClass0 et eClass1. Durant la phase d'apprentissage de ces classificateurs, un ensemble de règles floues qui décrivent les caractéristiques les plus importantes de chaque classe sont créées. Les règles créées sont constamment changées pour s'ajuster aux nouvelles données générées par les senseurs. La technique utilisée dans ces classificateurs est basée sur le partitionnement de l'espace de données en régions locales, qui peuvent se chevaucher, à travers l'estimation de la

densité récursive (RDE) et en associant les clusters (qui sont des clusters flous) à ces régions.

eClass0 possède un ordre de Takagi-Sugeno (eTS) [105] égal à zéro. Une règle floue du modèle eClass0 a la structure suivante :

$$Règle_i = \textbf{\textit{Si}}\ (X_1\ est\ P_1)\ \textbf{\textit{Et}}\ \dots\ \textbf{\textit{Et}}\ (X_n\ est\ P_n)\ \textbf{\textit{Alors}}\ Class = Class_i$$

où i est le numéro de la règle, n est le nombre de variables d'entrées (senseurs).

L'inférence dans eClass0 est produite en utilisant la règle "le gagnant prend tout" et les fonctions d'appartenance qui décrivent le degré d'association avec un prototype spécifique sont de forme gaussienne. Le potentiel, qui est une fonction Cauchy de la somme des distances entre un certain échantillon de données et tous les autres échantillons de données, est utilisé dans l'algorithme de partitionnement. Toutefois, dans ces classificateurs, le potentiel est calculé d'une façon récursive, ce qui rend l'algorithme rapide et plus efficace [106].

À la différence de eClass0, eClass1 est un classificateur non linéaire qui utilise un MIMO "Multi-Input-Multi-Output" eTS du premier ordre dans la régression sur le vecteur de fonction [107]. Ces règles évoluent dynamiquement en adaptant les paramètres du classificateur, la taille des règles et les points focaux. La structure d'une règle définie par eClass1 est de la forme suivante :

$$Règle_i = \textbf{\textit{Si}}\ (X_1\ est\ P_1)\ \textbf{\textit{Et}}\ \dots\ \textbf{\textit{Et}}\ (X_n\ est\ P_n)\ \textbf{\textit{Alors}}\ Class = (X^T\theta)$$

Dans cette dernière formule, la classe représente les sorties globales qui permettent de différencier les classes existantes en calculant la somme pondérée des sorties normalisées de chaque règle. Il est à noter que dans eClass1, le potentiel n'est pas calculé pour chaque

classe, mais c'est un potentiel global utilisé pour identifier les prototypes représentatifs de chaque classe [104].

Pour valider leur approche, les auteurs ont comparé les résultats des deux algorithmes avec ceux obtenus après l'utilisation d'algorithmes hors-ligne comme le HMM et le K-NN (K plus proches voisins). Le meilleur pourcentage de reconnaissance d'activités a été obtenu avec eClass1 64%, suivi par HMM 62% alors qu'avec eClass0 le pourcentage était de 50%. On peut constater que eClass0 n'est pas vraiment efficace dans la reconnaissance d'activités. Par contre, même si eClass1, avec sa méthode en ligne, n'a amélioré la reconnaissance que de 2%, les règles créées sont d'une grande importance et peuvent vraiment aider à répondre à quelques problèmes de la reconnaissance d'activités comme l'existence de plusieurs façons d'effectuer une activité ou le changement d'habitudes de la personne observée. En effet, à partir des trois règles, présentées ci-dessous, qui sont produites par eClass1 durant les expériences, on peut conclure qu'il existe deux façons d'effectuer l'activité *Diner* ou que l'occupant de la maison vient de changer ses habitudes en effectuant l'activité *Diner* différemment, et dans ce cas la première règle doit être retirée.

$$Règle_1 = \textbf{\textit{Si}}\ (X_1\ est\ 1)\ \textbf{\textit{Et}} (X_2\ est\ 0) \textbf{\textit{Et}}\ (X_n\ est\ 0)\ \textbf{\textit{Alors}}\ Class = \ Diner$$

$$Règle_2 = \textbf{\textit{Si}}\ (X_1\ est\ 0)\ \textbf{\textit{Et}} (X_5\ est\ 1) \textbf{\textit{Et}}\ (X_n\ est\ 1)\ \textbf{\textit{Alors}}\ Class = \ Diner$$

$$Règle_3 = \textbf{\textit{Si}}\ (X_3\ est\ 1)\ \textbf{\textit{Et}}\ (X_7\ est\ 1)\ \textbf{\textit{Alors}}\ Class = \ Dormir$$

3.5.3 Notre première approche de reconnaissance d'activités

À la lumière de ce que nous avons déjà présenté dans ce chapitre, nous avons créé une première approche de reconnaissance d'activités, lors de mon mémoire de maitrise qui a fait l'objet de trois publications [33], [37], [86], basée sur les objets. Le type de

reconnaissance de cette approche est une reconnaissance à l'insu, hors-ligne et non supervisée, qui vise à reconnaître les AVQBs et les AVQIs. Nous avons essayé, dans cette approche, de répondre à la problématique de réduction du nombre d'hypothèses qui rend la reconnaissance d'activités en temps réel une tâche très complexe. Notre approche commence par déduire le comportement normal de l'agent-acteur à partir de l'historique d'activation des senseurs en créant les modèles d'activités. Ensuite, au moment de la reconnaissance d'activités, elle sélectionne, parmi les modèles les plus probables, celle qui explique le mieux les récents senseurs activés. Cette approche est composée de quatre étapes : la réduction des données, la création des modèles d'activités, la sélection des modèles d'activités les plus probables et la recherche de l'activité entamée.

3.5.3.1 La réduction des données

Chaque senseur utilisé dans l'habitat intelligent envoie ses mesures à un délai très court (deux fois par seconde par exemple). Avec l'existence d'au moins une centaine de senseurs dans la maison, la base de données qui enregistre toutes ces mesures devient rapidement gigantesque et quasiment inutilisable. La première étape de notre approche visait donc à réduire les données sans perte d'informations pertinentes. La solution que nous avons élaborée consiste à ne plus sauvegarder tous les senseurs avec toutes leurs mesures, mais juste les senseurs activés avec leurs temps d'activation (cette étape est expliquée plus en détail dans le prochain chapitre). Un senseur est considéré comme activé quand il change d'état (entre ON et OFF) ou quand nous constatons une grande variation dans ses mesures (les petites variations sont considérées comme du bruit). La base de données ainsi créée est composée de plusieurs jours, chacun d'entre eux comporte une liste de senseurs activés avec leurs temps d'activation (*S1*(200), *S4*(220), …).

3.5.3.2 Création des modèles d'activités

À partir de la base de données créée dans l'étape précédente, une activité, qui d'habitude est composée d'une suite d'actions, est une séquence ordonnée de senseurs. Comme nous désirons reconnaître les AVQs que l'agent-acteur a l'habitude d'effectuer, ces séquences de senseurs auront la particularité d'être fréquentes. Nous considérons qu'une séquence est fréquente si son nombre d'occurrences est supérieur à une fréquence minimale définie au préalable. La création des modèles d'activités revient donc à trouver les séquences de senseurs fréquentes qui appartiennent au domaine de détection de motifs fréquents "Frequent pattern mining". Dans ce domaine, il existe plusieurs algorithmes, mais nous avons choisi d'utiliser BIDE, détaillé dans le chapitre 2, qui permet de détecter les motifs fréquents et fermés ce qui nous aide à trouver les activités sans toutes les sous-séquences qui les composent.

Bien que BIDE a permis de créer plus que 71% des modèles d'activités, il souffre d'un problème majeur du fait qu'il ne considère pas le temps entre les senseurs d'une activité ce qui empêcherait l'agent ambiant de détecter si l'agent-acteur rencontre des problèmes dans l'accomplissement de son activité.

3.5.3.3 Sélection des modèles d'activités les plus probables

Après la création de tous les modèles d'activités, les périodes où l'agent-acteur est habitué à effectuer les activités sont trouvées pour essayer de réduire le nombre d'hypothèses au moment de recherche. Nous commençons par trouver toutes les heures de début de chaque modèle d'activité à partir de la base de données. Ensuite, une segmentation temporelle est effectuée sur les heures de début de chaque modèle d'activité

afin de créer des intervalles temporels qui résument les périodes où l'agent-acteur est habitué à effectuer chaque activité. La Figure 14 schématise le résultat de la segmentation de deux activités(x et •) en deux intervalles chacune.

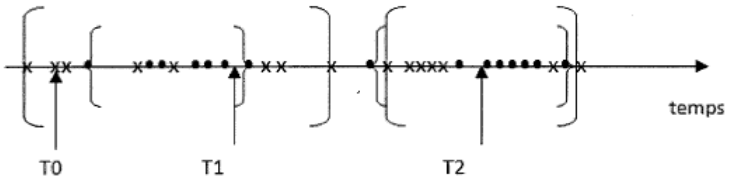


Figure 3-7. Exemple de répartition de deux activités dans le temps

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

Pour la segmentation temporelle nous avions pensé à l'algorithme k-means surtout que le nombre de clusters à créer *k*, qui doit être spécifié à l'avance, est facile à calculer puisqu'il est égal au nombre maximal que l'activité est effectuée dans une même journée. Après avoir constaté que les heures de débuts des activités sont déjà triées, nous avons décidé de créer notre propre algorithme qui profite de cette spécificité. À la différence du k-means qui choisit les centres des clusters au hasard puis affecte les éléments au centre le plus proche et recalcule les centres avant d'itérer jusqu'à ce que les centres ne changent plus, notre algorithme, détaillé à l'algorithme 3-1, cherche *k-1* fois les deux éléments adjacents les plus éloignés entre eux pour diviser l'intervalle qui les contient.

Algorithm Création des intervalles temporels

Entrees: L'ensemble des activités probables ActivProb
Sorties: Une liste non vide des intervalles de chaque activité dans ActivProb

```
1:  Pour chaque activité Activ ∈ ActivProb
2:      i ← 1
3:      Tant que i < Activ.NbrIntvl
4:          Pour chaque Temps t ∈ Activ.TempsDeb Et Temps ∉ Table
5:              M ← Max(TempsSuivant - Temps)
6:          Fin Pour
7:          Ajouter IndiceM dans Table
8:          Ajouter 1 à i
9:      Fin Tant que
10:     Trier_Asc(Table)
11:     Ajouter 0 à ActivProb.Intvl
12:     Pour chaque T ∈ Table
13:         Ajouter T à ActivProb.Intvl
14:         Ajouter T+1 à ActivProb.Intvl
15:     Fin Pour
16:     Ajouter Activ.TemsDeb.size-1 à ActivProb.Intvl
17: Fin Pour
18: retourner ActivProb
```

Algorithme 3-1. Création des intervalles temporels

Il est à noter que tout intervalle créé dont le nombre d'éléments est inférieur à la fréquence minimale est ignoré puisqu'il ne résume pas une habitude de l'agent-acteur, mais plutôt des exceptions ou des erreurs d'exécution.

Une fois les intervalles temporels créés, la réduction du nombre d'hypothèses se fait en ignorant, au moment de la recherche *T*, les modèles d'activités qui ne possèdent pas un intervalle temporel contenant *T*. Si on revient à la Figure 3-7, à *T0*, la deuxième activité (•) peut être ignorée alors qu'à *T1* et *T2*, les deux activités doivent être considérées.

3.5.3.4 Recherche de l'activité entamée

À ce stade, l'ensemble des activités que l'agent-acteur a l'habitude d'effectuer au moment de la recherche est sélectionné. Il reste à trouver, parmi cet ensemble, l'activité entamée ou celle qui doit l'être dans le cas où le patient rencontre des problèmes

d'initiation. Un réseau bayésien est utilisé pour ce fait. Les probabilités initiales de ce système sont calculées en se basant sur ces deux règles :

- Plus l'heure courante est proche de la fin d'un intervalle d'une activité, plus la probabilité de cette activité est élevée. Si nous revenons à la Figure 14 et que nous supposons que l'heure courante est *T1*, alors l'activité 2 (•) est plus probable que la première puisque *T1* est plus proche de la fin de l'intervalle de l'activité 2, symbolisé par le petit crochet, que celui de l'activité 1. La relation que nous utilisons pour ce calcul est :

$$P_{i1} = 1 - \frac{T_f - T_c}{D}$$

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 début de l'intervalle.

- Plus le pourcentage d'apparition d'une activité avant l'heure courante au sein d'un intervalle est grand, plus la probabilité 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 14, si $T_c = T_2$, alors l'activité 1 sera la plus probable; puisque, avant T_2 l'activité 1 apparaît 5 fois sur un total de 7, soit un pourcentage de 71%, tandis que l'activité 2 apparaît 2 fois sur un total de 5, soit un pourcentage de 40%. La relation que nous utilisons pour ce calcul est :

$$P_{i2} = \frac{N_{ac}}{N_t}$$

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

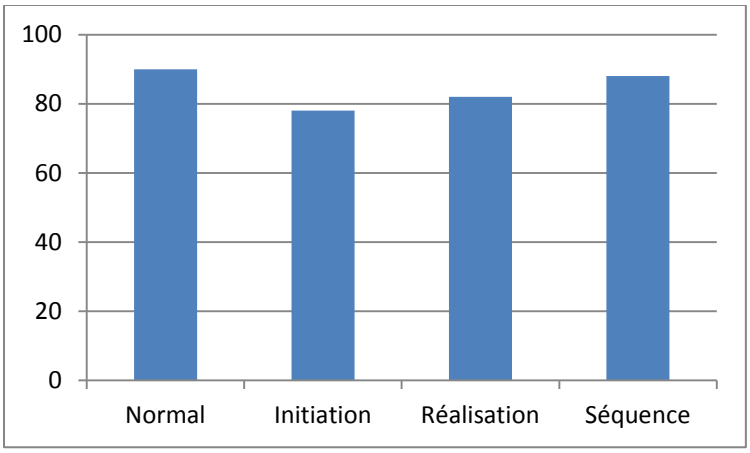
En fin, la probabilité initiale, qui est utilisée pour prédire l'activité qui doit être entamée, est calculée :

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

Les probabilités du réseau bayésien se mettent à jour pour prendre en considération les actions observées de l'agent-acteur. Chaque fois qu'un senseur est activé, la probabilité des activités, qui contiennent ce senseur, est augmentée tandis que celle des activités qui ne le comportent pas est diminuée. Enfin, l'activité avec la probabilité la plus élevée est l'activité reconnue.

Pour valider notre approche, nous avons choisi la technique du 90/10 en n'utilisant que 25 jours dans les précédentes étapes et en consacrant les données des trois derniers jours aux tests. Nous avons aussi introduit des erreurs dans les données pour simuler les erreurs susceptibles d'être commises par le patient et les résultats sont représentés dans le Graphe1:

- Normal : Les résultats ont été enregistrés après avoir donné au système les mêmes données 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 données introduites au système comportent 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.



Graphe 3-1. Résultats de tests du système

Les résultats obtenus montrent l'efficacité du système à reconnaître et prédire les activités. Même si nous imposons un ordre strict aux senseurs qui composent l'activité, le pourcentage de reconnaissance des activités avec des erreurs de séquence est élevé puisque la position du senseur n'est pas considérée lors de la mise à jour de la probabilité. Le problème majeur dont souffre ce système est la représentation des intervalles temporels des habitudes de l'agent-acteur qui est toujours la même pour les différents jours de la semaine, sachant que ces habitudes peuvent changer selon le jour de la semaine.

3.6 Conclusion

Ce chapitre avait pour objectif de présenter un état de l'art sur la reconnaissance d'activités. Nous avons commencé par définir les activités que nous allons essayer de reconnaître (AVQB et AVQI) et le type de collaboration de l'agent acteur avec l'agent observateur qui nous a orienté vers une reconnaissance à l'insu. Puis, nous avons montré comment le choix des senseurs influence 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. [17] en est un bon exemple 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 reconnaître l'activité entamée. 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. Nous avons par la suite présenté d'autres exemples expliquant les autres types de reconnaissance d'activités. Les travaux de Lara et al. [97] et Maurer et al. [99] ont été choisis pour représenter les approches de reconnaissance d'activités basées sur les senseurs portables. Nous avons vu comment les signaux générés par ces senseurs peuvent être segmentés et classifiés pour reconnaître les activités. Pour les approches supervisées basées sur les objets, nous avons détaillé le travail de Suryadevara et al. [15] où la personne observée notait dans une feuille les activités entamées pour faciliter la classification. Un autre travail que nous avons présenté est celui de Jakkula et Cook [16] qui fait partie des approches basées sur les objets, mais d'une manière non supervisée. 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 inférée à partir des activités détectées et leurs relations avec les autres activités. Parmi les limitations de cette approche, les calculs effectués sont assez lourds pour une application en temps réel et le nombre d'hypothèses n'est pas réduit. Le dernier type d'approches expliqué concernait les approches de reconnaissances d'activités en ligne et nous avons pris comme exemple le travail de Ordonez et al. [8]. Dans ce travail, l'apprentissage se fait

directement sur le flux de données généré par les senseurs pour considérer toute modification dans les habitudes de la personne assistée.

Avant de conclure ce chapitre, nous avons expliqué notre première approche qui commence par la création des modèles d'activités composés de séquences fréquentes de senseurs activés. Les heures de débuts de ces modèles sont résumées par la suite dans des intervalles temporels qui représentent les périodes où l'agent-acteur est habitué à effectuer ses activités. Au moment de la reconnaissance, seuls les modèles d'activité qui possèdent un intervalle temporel contenant l'heure courante sont considérés dans la recherche et celle avec la probabilité la plus élevée est prédite ou reconnue comme l'activité entamée. Deux problèmes majeurs dont souffre notre approche ont inspiré notre nouvelle approche qui sera présentée au chapitre suivant : le temps entre deux senseurs n'est pas considéré dans l'algorithme de création des modèles d'activités ce qui empêche l'agent ambiant de détecter les erreurs du patient et les intervalles temporels qui résument les habitudes de l'agent-acteur sont créés pour tous les jours et non pas pour chaque jour de la semaine, surtout que nos habitudes pour la fin de semaine, par exemple, sont très différentes de celles des autres jours de la semaine.

Chapitre 4

4 Prédiction et reconnaissance d'activités

4.1 Introduction

La vieillesse des populations est un des plus grands défis auquel les sociétés modernes sont confrontées. À un certain âge, la personne perd son autonomie et requiert un aidant pour subvenir à ses besoins et effectuer ses activités de vie quotidienne. Le nombre élevé des personnes âgées, auxquelles s'ajoutent les patients de la maladie d'Alzheimer qui ont besoin aussi d'assistance, pèse lourd sur le système de santé en ressources financières et personnelles. De plus, l'assistance traditionnelle, proposée jusqu'à présent, souffre de plusieurs inconvénients comme la perte d'intimité de la personne assistée, l'installation d'une relation complexe entre elle et l'assistant et la précipitation de la dépendance totale de la personne assistée qui s'habitue à recevoir de l'aide de l'assistant [2]. Le développement électronique et l'émergence du domaine de l'intelligence ambiante ont rendu possible la conception d'une assistance technologique où un agent ambiant vient épauler la personne assistante dans son travail. L'agent ambiant est doté des mêmes facultés que celles de la personne assistante qui leur permet de fonctionner d'une façon similaire : observer, apprendre les habitudes de l'agent acteur,

prédire et déduire l'activité entamée, détecter les erreurs commises par l'agent-acteur et décider du moment opportun pour lui proposer de l'aide.

Dans le chapitre précédent, nous avons présenté quelques travaux proposant différentes solutions à la reconnaissance d'activités qui est l'étape la plus délicate dans le processus de l'assistance technologique. Nous avons aussi situé notre approche en définissant les types d'AVQs que nous désirons reconnaître; les AVQBs et les AVQIs et en précisant le type de reconnaissance en une reconnaissance à l'insu basée sur les objets. La Figure 4-1 schématise la procédure d'observation choisie pour notre approche.

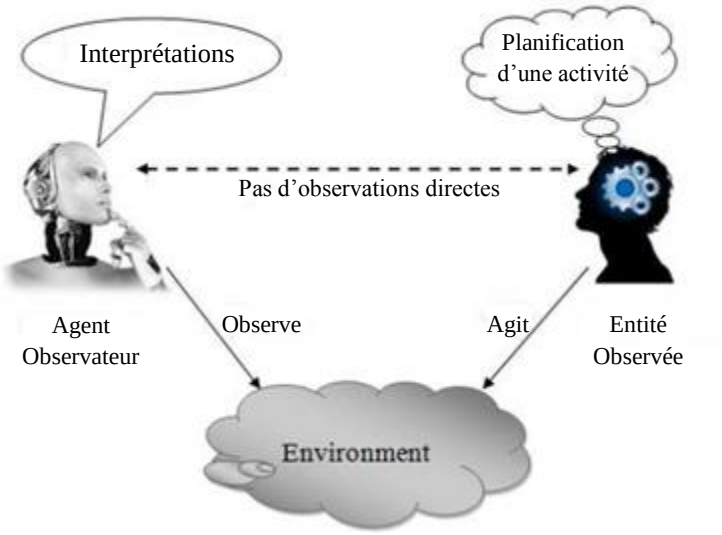


Figure 4-1. La procédure d'observation

Les senseurs que nous utilisons ne sont placés que sur les différents objets de la maison d'une façon transparente à l'agent-acteur. Aucune caméra ou autre senseur n'observe directement l'agent-acteur. Par contre, ses actions sont déduites lors de son interaction avec l'environnement. Par exemple, si l'état du senseur placé sur le réfrigérateur est à OFF, quand son état change à ON, l'action *Ouvrir réfrigérateur* est déduite. Nous avons choisi ce type de reconnaissance parce qu'il préserve mieux l'intimité de l'occupant de la maison.

Notre nouvelle approche, qui a été publiée dans le journal Springer AIHC [9], propose des solutions aux différentes étapes de l'assistance technologique tout en répondant aux différents problèmes rencontrés lors de notre première approche détaillée au chapitre précédent. La Figure 4-2 présente les différentes étapes de cette nouvelle approche.

Figure 4-2. Les étapes de l'assistance technologique

Comme le montre la Figure 4-2, notre approche est divisée en trois grandes étapes. La première est une étape de préparation des données qui transforme les données enregistrées à partir des senseurs pour qu'elles deviennent utilisables par les prochaines étapes. La deuxième étape est celle de la prédiction et reconnaissance d'activités qui est divisé à son tour en trois étapes : l'apprentissage des comportements normaux de l'agent-acteur que nous traduisons par la création des modèles d'activités, la réduction du nombre

d'hypothèses qui est basée sur la prédiction des temps de débuts des modèles d'activités et la recherche de l'activité entamée. La dernière permet à l'agent ambiant de détecter les erreurs susceptibles d'être commises par l'agent-acteur et choisir le moment opportun pour lui offrir de l'aide. Toutes ces étapes seront détaillées dans la suite de ce chapitre.

4.2 Préparation des données

Notre approche basée sur les objets consiste à installer, d'une façon transparente à l'agent-acteur, des senseurs sur les différents objets de la maison intelligente afin de mesurer différentes propriétés de ces objets. Un tag RFID, posé sur une tasse par exemple, permet de mesurer la force du signal par rapport aux différentes antennes RFID installées dans la maison, ce qui aide à estimer la position de la tasse. Puisque l'assistance doit se dérouler en temps réel, tout changement de position de la tasse doit être détecté instantanément. Pour cette raison, les senseurs doivent effectuer et envoyer leurs mesures à une fréquence très élevée. Dans notre cas, les mesures sont effectuées deux fois par seconde. Le grand nombre des senseurs, installés sur les différents objets de la maison, ainsi que la fréquence élevée des mesures rendent la base de données, qui enregistre toutes ces mesures, très volumineuse et pratiquement inutilisable. Le Tableau 4-1 donne un exemple de cette base de données.

Tableau 4-1. Enregistrement de toutes les mesures des senseurs

Date	Heure	Senseur 1	Senseur …	Senseur N
01/04/2015	08:50:20.000	ON	…	1245
01/04/2015	08:50:20.500	OFF	…	1245

Dans l'objectif de réduire les données, nous avons décidé de ne garder que les informations les plus pertinentes. En se basant sur le type d'aide qui sera proposé à l'agent-acteur, qui l'incitera par exemple à utiliser la tasse avec un moyen quelconque sans lui indiquer sa position, nous avons constaté que l'information la plus pertinente n'est pas la mesure en tant que telle, mais la détection de l'utilisation de l'objet ainsi que le moment de son utilisation. Pour détecter si l'agent-acteur a utilisé un objet, il suffit de comparer deux mesures successives du senseur installé sur cet objet. Si la mesure n'a connue aucun changement, on déduit que l'agent acteur ne l'a pas utilisé. Par contre, si un senseur booléen a changé d'état (ON à OFF ou OFF à ON) ou un senseur numérique a connu un grand changement dans sa valeur mesurée, le senseur est considéré comme activé et traduit une action de l'agent acteur. Il faut noter que les petits changements des valeurs numériques sont considérés comme du bruit et ne sont pas considérés comme des actions. La base de données que nous avons décidé de créer ne comporte donc plus toutes les mesures des senseurs, mais seulement les noms des senseurs activés ainsi que leurs temps d'activation. Le Tableau 4-2 donne un exemple de cette base de données.

Tableau 4-2. Senseurs activés

Date	Senseurs activés
01/04/2015	Senseur7(Heure 1), Senseur3(Heure 2), Senseur5(Heure 3), …
02/04/2015	Senseur2(Heure 1), Senseur7(Heure 3), …

Cette dernière base de données est beaucoup moins volumineuse que la première, ce qui permet de mieux l'exploiter. C'est cette base de données qui sera utilisée pour le reste des étapes de notre approche et peut être référée comme le journal d'historique des senseurs.

4.3 Création des modèles d'activités

Lors de toute assistance, les erreurs sont détectées en comparant le comportement actuel de l'agent-acteur à ses différents comportements normaux. L'assistant doit donc posséder à l'avance la liste de tous ses comportements normaux. Comme nous nous intéressons à l'assister durant la réalisation de ses activités quotidiennes, les comportements normaux sont donc ses différents AVQs.

Par définition, une activité est une séquence ordonnée d'actions. Dans notre cas, une action de l'agent-acteur est détectée par l'activation d'un senseur, ce qui redéfinit l'activité en une séquence ordonnée de senseurs activés. La différence entre les collections de senseurs installés dans chaque maison ainsi que la façon particulière avec laquelle chaque personne effectue ses activités, rendent impossible l'utilisation d'une liste de modèles d'activités unifiée pour tous les cas. De plus, le nombre élevé des AVQs complique la création de ces modèles d'une façon supervisée. Les modèles d'activités doivent donc être créés à partir du journal d'historique des senseurs propre à chaque personne d'une façon non supervisée en se basant sur le fait que les AVQs reviennent assez souvent et seront forcément fréquentes. La création des modèles d'activités s'effectue donc, dans notre approche, en trouvant les séquences ordonnées et fréquentes de senseurs activés à partir du journal d'historique des senseurs.

Au cours de notre première approche, résumée dans le chapitre précédent, nous avons utilisé BIDE pour trouver les modèles d'activités. Le problème majeur de cet algorithme est qu'il ne considère pas le temps entre les senseurs adjacents de l'activité, ce qui n'empêche pas seulement la détection des erreurs de l'agent-acteur, mais ignore aussi une information importante capable de différencier les activités. Par exemple, si *bouillir*

de l'eau a été détecté pour une durée de deux minutes, cette action peut être attribuée à l'activité *préparer thé*. Par contre si elle a été détectée pour une dizaine de minutes, elle sera plutôt attribuée à l'activité *préparer pâtes*. Afin de répondre à ce problème, nous avons développé un nouvel algorithme de détection de séquences ordonnées et fréquentes, qui a été présenté à la conférence SSCI en 2014 [35] et que nous détaillons dans cette section.

4.3.1 Définition du problème

À partir d'un ensemble de séquences d'événements *S*, comme celui présenté au Tableau 4-2, l'algorithme permet de trouver les sous-séquences fréquentes, homogènes et fermées. Soit *Es* l'ensemble des senseurs, un événement est une paire (*A*,*t*) où $A \in Es$ est un senseur et *t* un entier (en secondes) représentant l'heure d'activation du senseur *A* (Pour utiliser des entiers, l'heure d'activation est calculée en étant égale à la différence en secondes de l'heure d'activation du senseur et 6h00 am). La Figure 4-3 présente un exemple d'un ensemble composé de deux séquences d'événements s_1 et s_2:

s_1 = < *(A, 01), (B, 04), (C, 06), (D, 28), (A, 36), (A, 41), (B, 56), (E, 58), (A, 59)* >

s_2 = < *(E, 08), (A, 11), (B, 30), (E, 32), (A, 42), (B, 46), (C, 49)* >

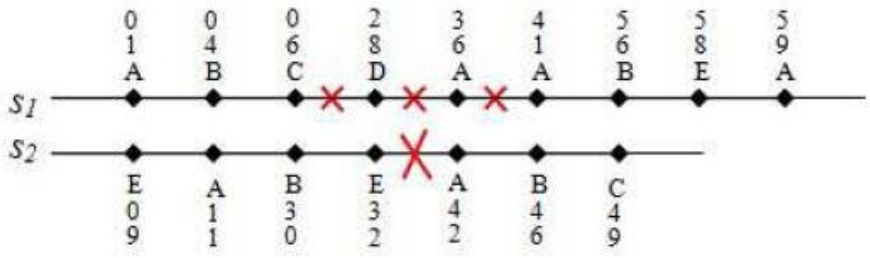


Figure 4-3. Un exemple d'un ensemble composé de deux séquences d'événements

Définition 1 (**apparence**) : une sous-séquence

$s' = < (A'_1, t'_1), \ldots, (A'_i, t'_i), (A'_{i+1}, t'_{i+1}), \ldots, (A'_k, t'_k) >$ apparaît dans une séquence

d'événements $s =< (A_1, t_1), \dots, (A_i, t_i), (A_{i+1}, t_{i+1}), \dots, (A_n, t_n) >$ avec $n > k$ s'il existe au moins une sous-séquence

$s'' =< (A''_1, t''_1), \dots, (A''_i, t''_i), (A''_{i+1}, t''_{i+1}), \dots, (A''_k, t''_k) >$ de s tel que : $A'_1 = A''_1$, …, $A'_i = A''_i$, $A'_{i+1} = A''_{i+1}$,… et $A'_k = A''_k$ et $t_i \leq t_{i+1}$.

Dans cette définition d'apparition, nous n'avons spécifié aucune condition sur le temps (t) parce qu'une activité peut être performée à n'importe quel moment de la journée et difficilement avec les mêmes délais entre les senseurs. Par contre, ces délais seront conditionnés avec la notion d'homogénéité que nous avons introduite et que nous expliquerons plus loin.

Définition 2 (**sous-séquence fréquente**) : une sous-séquence d'événements est fréquente si son nombre d'apparitions dans *S* est supérieur ou égal à une fréquence minimale prédéfinie.

Il faut noter qu'une sous-séquence peut apparaître plusieurs fois dans la même séquence comme une activité peut se produire plusieurs fois dans la même journée. À partir de la Figure 4-3, si la fréquence minimale est égale à deux, la sous-séquence $s'_1 =< (A, t_1), (B, t_2), (C, t_3) >$ est fréquente parce que sa fréquence est égale à la fréquence minimale.

Définition 3 (**sous-séquence fréquente et fermée**) : cette définition est composée de deux parties :

- Les sous-séquences fréquentes les plus longues (contiennent le plus de senseurs) sont fermées.

- Une sous-séquence fréquente est dite fermée si elle reste fréquente après la réduction de sa fréquence par son nombre d'apparitions dans toute autre sous-séquence plus longue, fréquente et fermée.

La notion de fermeture est utilisée pour ne pas considérer chaque partie d'une activité comme une activité à part entière sauf quand c'est le cas. Par exemple, *préparer café* est une activité, mais elle peut aussi être une partie de l'activité *préparer petit déjeuner*. Si on revient à notre exemple, la sous-séquence $s_1'' =< (A,t_1),(B,t_2),(E,t_3) >$ est fréquente et fermée puisqu'elle apparaît deux fois dans *S* et n'apparaît dans aucune autre sous-séquence fréquente et fermée. Par contre, $< (A,t_1),(B,t_2) >$ n'est pas une sous-séquence fréquente et fermée parce que sa fréquence après la notion de fermeture est égale à zéro; elle apparaît quatre fois dans *S*, mais apparaît deux fois dans s_1' et deux fois dans s_1''.

Définition 4 (**intervalle homogène**) : pour deux événements e et e', on considère l'ensemble $E_{e,e'}$ des paires $< (e,t),\ (e',t') >$ telles que t et t' sont deux moments successifs dans une séquence. On peut définir l'ensemble des différences $\Delta E_{e,e'}$ comme $\{\, t' - t : < (e,t),\ (e',t') > \in E_{e,e'}\}$. L'intervalle $I_{e,e'} = [t_{min}, t_{max}]$ dont les bornes sont respectivement : $t_{min} = \min \Delta E_{e,e'}$ et $t_{max} = \max \Delta E_{e,e'}$ est un intervalle homogène si et seulement si la différence $t_{max} - t_{min}$ est faible.

L'utilisation des intervalles homogènes garantit que leur représentation ne sera pas affectée par des erreurs ou des exceptions. Si par exemple la durée habituelle entre deux adjacents senseurs est entre 5 et 15 secondes, et une fois à cause d'une erreur, elle était de 200, la représentation de l'intervalle homogène sera toujours [5-15] au lieu de [5-200]. Notre algorithme utilise le "Fuzzy C-means" pour transformer un ensemble des

différences temporelles de deux senseurs adjacents en un ou plusieurs intervalles homogènes.

Définition 5 (**couple d'événements**) : nous désignons par un couple d'événements deux adjacents événements avec un intervalle homogène fréquent.

Deux différents couples de senseurs peuvent avoir les mêmes senseurs, mais différents intervalles homogènes fréquents. Dans la Figure 4-3, nous pouvons détecter deux différents couples de senseurs avec les mêmes senseurs *A* et *B*. Le premier a un intervalle homogène fréquent [3-4] et le deuxième [15-19].

Définition 6 (**sous-séquence homogène**) :

une sous-séquence $< (e_1, t_1), \dots, (e_i, t_i), (e_j, t_j), \dots, (e_n, t_n) >$ est homogène si chaque différence temporelle entre deux événements adjacents e_i et e_j appartient à l'intervalle d'un couple d'événements composé des mêmes événements : $t_j - t_i \in I_{e_i,e_j}[t_{min}, t_{max}]$.

Pour démontrer l'importance de la notion de l'homogénéité, si nous revenons à la Figure 4-3, si le senseur *B* est activé après trois secondes de l'activation du senseur *A*, l'agent ambiant peut prédire que le senseur *C* va être activé dans deux à trois secondes. Par contre, si le senseur *B* est activé après dix-sept secondes de l'activation du senseur *A*, l'agent ambiant va plutôt prédire que c'est le senseur *E* qui va être activé après deux secondes.

Définition 7 (**activité**) : une séquence d'événements

$s =< (A_1, t_1), (A_2, t_2), \dots, (A_n, t_n) >$ est considérée comme une activité si et seulement si s est une sous-séquence fréquente, homogène et fermée dans *S*.

Définition 8 (**modèle d'activité**) : un modèle d'activité est défini par

$am =< Am_1(I_1), Am_2(I_2), \dots, Am_{k-1}(I_{k-1}), Am_k >$ tel que :

- $Am_i \in Es$ pour tous i =1, …, k;
- $I_j = [t_d, t_f]$ est l'intervalle homogène fréquent du couple de senseurs Am_j et Am_{j+1} pour tous j=1, …, k-1.

Une activité s correspond à un modèle d'activité am si :

- $n = k$;
- $A_i = Am_i$ pour tous i=1, …, n;
- $(t_{i+1} - t_i) \in I_i$ pour tous i =1, …, n-1.

4.3.2 Découverte des modèles d'activités

La majorité des algorithmes de l'extraction des motifs fréquents sont basés sur l'algorithme Apriori. L'idée générale de cet algorithme est de trouver les éléments fréquents de taille inférieure et de les combiner pour trouver les plus longs. À l'opposé de cette idée, notre algorithme coupe les séquences entre les senseurs adjacents qui ne constituent pas des couples de senseurs, tel qu'ils sont définis à la section précédente, pour n'en garder que de plus petites séquences fréquentes. Si on revient à notre exemple, la Figure 4-3 montre, avec les X rouges, l'endroit des premières coupures. La première étape de notre algorithme est une étape de transformation de la base de données temporelle à une base de données non temporelle sans perte d'informations pertinentes. Après l'enregistrement de tous les couples de senseurs trouvés dans une table, la substitution de chaque couple de senseurs avec son indice dans la table élimine les informations temporelles de la base de données. La deuxième étape est l'étape de détection des séquences fréquentes et homogènes. Dans cette étape, des coupures et substitutions similaires à ceux de la première étape sont effectuées sauf que cette fois on

utilise les indices adjacents au lieu des couples de senseurs puisque les informations temporelles n'existent plus. La dernière étape de cet algorithme sélectionne parmi les séquences fréquentes et homogènes trouvées celles qui sont fermées pour constituer les modèles d'activités. Dans la suite de cette section, nous allons expliquer les algorithmes utilisés pour chaque étape.

4.3.2.1 D'une BD temporelle à une BD non temporelle

L'existence de données temporelles au sein d'une base de données augmente considérablement la complexité de son exploitation parce qu'on doit tenir compte des différents types de ces données, des relations ou opérateurs temporels et de la granularité temporelle, [60] etc. Heureusement, la nature de notre problème nous permet de résumer ces informations et au lieu de garder tous les temps entre deux senseurs adjacents pour tous les jours d'observation, on peut garder un ou plusieurs intervalles temporels qui résument ces durées. L'algorithme T2NTDB, présenté à l'algorithme 4-1, détaille la méthode que nous avons utilisée pour ce propos.

Algorithm T2NTDB: Transformer une BD temporelle en une non temporelle

Entrees: Un ensemble de séquences temporelles S, Fréquence minimale F
Sorties: Un ensemble de séquences non temporelles S', un tableau des couples de senseurs T'

```
1: Pour chaque séquence s ∈ S
2:     Pour chaque senseurs adjacents c ∈ s
3:         Si c ∈ T Alors            //T: comme Tableau 4-2
4:             c.fréquence++
5:         Sinon
6:             Ajouter c à T
7:             c.fréquence ← 1
8:         Fin Si
9:     Fin Pour
10: Fin Pour
11: Pour chaque c ∈ T
12:     Si c.fréquence < F Alors
13:         Supprimer c
14:     Fin Si
15: Fin Pour
16: Appeler l'algorithme CHI          //pour créer T' (Tableau 4-4)
17: Pour chaque sequence s ∈ S
18:     Créer une nouvelle séquence vide s'
19:     Pour chaque senseurs adjacents c ∈ s
20:         Si c ∈ T' Alors
21:             Ajouter son indice (k) dans T' à s'
22:         Sinon
23:             Si s' est non vide Alors
24:                 Ajouter s' à S'
25:                 Vider s'
26:             Fin Si
27:         Fin Si
28:     Fin Pour
29: Fin Pour
30: reoturner S' Et T'
```

Algorithme 4-1. Transformation de la base de données

L'algorithme prend en entrée la base de données temporelle *S*, Tableau 4-2, et la fréquence minimale *F* dont le choix est discuté dans le chapitre de validation, et produit comme sortie une nouvelle base de données non temporelle *S'* et un tableau *T'* qui sauvegarde les intervalles temporels de chaque couple de senseurs. T2NTDB commence par trouver tous les différents senseurs adjacents, leurs fréquences ainsi que toutes les durées qui les séparent puis n'en garde que les fréquents. Le Tableau 4-3 montre le résultat de l'application de cette partie de l'algorithme sur notre exemple.

Tableau 4-3. Les senseurs adjacents fréquents

Premier senseur	Deuxième senseur	Fréquence	Durées entre les deux senseurs
A	*B*	4	3, 15, 19, 4
B	*C*	2	2,3
B	*E*	2	2,2
E	*A*	3	1, 2, 10

Par la suite, T2NTDB fait appel à l'algorithme CHI pour créer les couples de senseurs en transformant les durées en intervalles homogènes. Dans l'algorithme CHI, nous avons choisi d'utiliser le "Fuzzy C-means" pour la création des intervalles parce qu'il permet à un élément d'appartenir à plus qu'un intervalle, ce qui augmente les chances des clusters créés pour qu'ils soient fréquents. Le problème avec le C-means est qu'il nécessite la spécification du nombre de clusters à créer à l'avance. Pour décider du nombre optimal des clusters à créer, nous exécutons le C-means avec différentes valeurs, puis le choix est précisé par une fonction critère C_N que nous avons définie. Comme expliqué à la section définition du problème, les éléments d'un intervalle temporel homogène doivent être assez proches de la médiane. Pour cette raison, C_N calcule la déviation moyenne des éléments de la médiane d'un cluster et quand plusieurs clusters existent, le résultat de cette fonction est la somme des déviations moyennes de tous les clusters :

$$C_N = \sum_{i=1}^{N} \frac{\sum_{j=1}^{Nbr\ Éléments} |x_{ij} - Md_{ij}|}{Nbr\ Éléments}$$

Pour détailler plus la création des couples de senseurs, l'algorithme CHI est présenté à l'algorithme 4-2.

Algorithm CHI: Créer les couples de senseurs

Entrees: Un tableau de senseurs adjacents avec leurs différences temporelles T, Fréquence minimale F

Sorties: Un tableau des couples de senseurs fréquents T'

1: $N \leftarrow 1$
2: **Pour chaque** $c \in T$
3: Calculer la valeure de la fonction du critère C_1 (pour $N = 1$ cluster)
4: $N++$
5: Appliquer FCM sur les différences temporelles en utilisant N clusters
6: Calculer la nouvelle valeure de la fonction du critère C_N
7: **Si** $C_{N-1} - C_N > \varepsilon$ **Alors** //ε est définie expérimentalement
8: **Aller à** 4
9: **Sinon**
10: **Pour chaque** Clucter cl de la division $N-1$
11: **Si** le nombre des éléments des clusters $n \geq F$ **Alors**
12: Ajouter c à T' (n comme fréquence, cl comme cluster)
13: **Fin Si**
14: **Fin Pour**
15: **Fin Si**
16: **Fin Pour**
17: **retourner** T'

Algorithme 4-2. Création des couples de senseurs

CHI commence par considérer que toutes les durées de deux senseurs adjacents constituent un seul intervalle homogène et calcule C_N pour N=1. Il considère ensuite le cas où il existe deux intervalles et fait appel au Fuzzy C-means avec *N*=2, puis calcule C_2. Si C_2 améliore la valeur de C_1, il considère *N*=3 et ainsi de suite jusqu'à ce que la dernière valeur de C_N n'améliore plus celle de C_{N-1} d'une façon significative. Une fois l'incrémentation de *N* est arrêtée, *N*-1 sera choisi comme valeur optimale et le C-means créera N-1 intervalle temporel homogène, ce qui signifie la création de *N*-1 couples de senseurs pour les mêmes senseurs adjacents.

Si on revient à notre exemple et on considère les deux senseurs adjacents (*A*, *B*) avec leurs durées 3, 4, 15 et 19, la déviation moyenne de ce cluster C_1 est égale à 6.75. Pour *N*=2, le premier cluster créé par FCM est $I_{A,B}[3-4]$ avec C_{2_1}égale à 0.5 et le deuxième est $I_{A,B}[15-19]$ avec C_{2_2}égale à 2. La valeur résultante de C_2 est donc 2.5 ($C_{2_1} + C_{2_2}$) qui est largement inférieur à C_1 ce qui veut dire que *N*=2 améliore nettement *N*=1. En

calculant C_3, nous trouvons qu'il n'améliore pas C_2 d'une façon significative ce qui veut dire que la valeur optimale du nombre de clusters à créer est deux. Il faut signaler que si C_{i+1} améliore C_i d'une façon non significative, le nombre optimal de clusters qui sera choisi est *i* parce que nous souhaitons avoir des clusters fréquents. Le Tableau 4-4 montre le résultat de l'application du CHI sur le Tableau 4-3.

Tableau 4-4. Création des intervalles temporels homogènes.

Premier senseur	Deuxième senseur	Fréquence	Durées entre les deux senseurs
A	*B*	2	[3, 4]
A	*B*	2	[15, 19]
B	*C*	2	[1,3]
B	*E*	2	[2,2]
E	*A*	2	[1, 2]

La dernière partie du T2NTDB consiste à parcourir *S* en testant chaque deux senseurs adjacents de chaque séquence. Quand les senseurs adjacents sont un couple de senseurs, ils existent dans le tableau créé par CHI, ils sont remplacés par leur indice dans le tableau. Par contre, si les senseurs adjacents ne sont pas un couple de senseurs, T2NTDB coupe entre les deux et crée ainsi deux différentes séquences et le parcours de *S* continue de la deuxième séquence créée. À la fin de cette partie, les séquences composées de moins de deux éléments sont éliminées et la nouvelle base de données ainsi obtenue n'est composée que d'indices et ne contient aucune information temporelle.

Pour notre exemple, l'ensemble *S'* créé sera composé de quatre séquences non temporelles :

< 0, 2 >
< 1, 3, 4 >
< 4, 1, 3 >
< 0, 2 >

4.3.2.2 Détection des séquences fréquentes et homogènes

La deuxième étape de notre algorithme ressemble à la première à la différence de l'absence d'informations temporelles. La dernière base de données créée est parcourue pour créer cette fois-ci juste un tableau de fréquences des senseurs adjacents et non pas des couples de senseurs. Un deuxième parcours est effectué pour effectuer les substitutions et les coupures selon si les deux senseurs adjacents sont fréquents ou non. Après l'élimination des séquences de moins de deux éléments, cette étape est répétée jusqu'à l'élimination de toutes les séquences. Les six premières lignes de l'algorithme 4-3 détaillent cette étape.

Algorithm Créatoin des modèles d'activitiés

Entrees: Un ensemble de séquences non temporelles S', Fréquence minimale F
Sorties: Un ensemble des modèles d'activitiès AM

1: **Tant que** S' n'est pas vide
2: Créer un nouveau tableau des couples de senseurs fréquents T' à partir de S'
3: Utiliser T' pour créer, par substitution, un nouvel ensemble qui sera le nouveau S'
4: Enregistrer T' dans un tableau de tableaux TT
5: **Fin Tant que**
6: Ajouter toutes les sous-séquences de TT à AM
7: **Pour chaque** sous-séquence s d'un tableau $TT[i]$, i de $TT.size() - 2$ à 0
8: **Pour chaque** sous-séquence s' d'un tableau $TT[j]$, j de $i + 1$ à $TT.size() - 1$
9: **Si** s apparait dans s' **Alors**
10: $s.fréquence \leftarrow s.fréquence - s'.fréquence$
11: **Fin Si**
12: **Fin Pour**
13: **Si** $s.fréquence > F$ **Alors**
14: Ajouter s à AM
15: **Fin Si**
16: **Fin Pour**
17: **retourner** AM

Algorithme 4-3. Création des modèles d'activités

Comme le montre cet algorithme, tous les tableaux créés sont sauvegardés dans un tableau de tableaux *TT* sachant que chaque tableau contient les indices du tableau qui le précède sauf le premier qui contient les couples de senseurs. De cette manière, les senseurs du dernier tableau peuvent être retrouvés d'une façon récursive en suivant les indices jusqu'au premier tableau. En programmant cette partie, nous avons trouvé que la méthode récursive prend beaucoup de temps et que la méthode itérative est plus rapide. Pour cette raison, nous avons ajouté une colonne dans les tableaux que nous avons appelée sous-séquence pour comporte la liste des senseurs. Le Tableau 4-5 donne le dernier tableau créé pour notre exemple.

Tableau 4-5. Tableau contenant les sous-séquences.

Premier senseur	Deuxième senseur	Fréquence	Sous-séquence
0	2	2	A, B[3,4], C[1,3]
1	3	2	A, B[15,19], E[2,2]

4.3.2.3 Détection des séquences fréquentes, homogènes et fermées

À ce stade de notre algorithme, toutes les séquences fréquentes et homogènes sont détectées (la colonne Sous-séquence de tous les tableaux). Il reste juste à trouver parmi elles les séquences fermées. La notion de fermeture est utilisée pour ne pas considérer toute partie d'un modèle d'activité comme un modèle d'activité à part entière sauf quand c'est le cas. La séquence fréquente et homogène doit donc rester fréquente après la soustraction, de sa fréquence, du nombre de ses apparitions dans toute autre séquence fermée. Comme détaillées aux dix dernières lignes de l'algorithme 4-3, les sous-

séquences du dernier tableau sont automatiquement considérées comme fermées, puisqu'elles ne peuvent apparaître dans de plus longues séquences. On peut voir aussi comment les fréquences des sous-séquences des autres tableaux sont modifiées en testant leur appartenance aux sous-séquences des tableaux qui les succèdent.

L'algorithme que nous proposons pour la création des modèles d'activités ne trouve pas seulement les senseurs activés qui composent une activité, mais il trouve aussi un intervalle de temps entre chaque deux senseurs adjacents, ce qui permet de différencier quelques activités et d'aider l'agent ambiant à détecter les erreurs commises par l'occupant de la maison. Les modèles d'activités créés sont parfaitement ordonnés, ce qui permet à l'agent ambiant de proposer, sans ambiguïté, la prochaine action à l'agent acteur en cas d'erreur. Nous sommes conscients qu'il est impossible que l'agent acteur effectue ses activités toujours de la même façon, mais nous savons aussi que toute personne à son propre mode de vie et ses habitudes qui feront en sorte que, la plupart du temps, elle va effectuer ses activités d'une façon similaire. Pour cette raison, le choix de la fréquence minimale joue un rôle crucial dans cet algorithme. Ce choix sera étudié lors de la section de validation.

4.4 Prédiction des temps de début des activités

Après la création des modèles d'activités, la reconnaissance d'activité est effectuée en trouvant parmi ces modèles, l'activité qui explique le mieux les récents senseurs activés. Le nombre élevé des modèles d'activités, que nous appelons le nombre d'hypothèses, rend cette recherche difficile surtout qu'elle doit se faire en temps réel. Dans notre première approche, nous nous sommes basés sur la segmentation temporelle

pour spécifier la période où chaque activité est commencée et la réduction du nombre d'hypothèses est effectuée en comparant l'heure courante à ces périodes. Le problème de cette solution est qu'elle ne considère pas la spécificité du jour de la semaine : elle crée une période pour une activité pour tous les jours de la semaine sachant que, par exemple, dans les fins de semaines, l'agent acteur peut être habitué à retarder certaines activités comme *Prendre petit déjeuner*.

Dans cette approche, après de longues recherches, nous avons décidé d'utiliser les séries temporelles pour bénéficier de la propriété de saisonnalité qui permettra de garder la spécificité de chaque jour de la semaine. L'idée générale de cette solution consiste à utiliser les techniques de prédiction des séries temporelles, avec une saisonnalité égale à 7, pour prédire le temps de début de chaque activité. Cette prédiction peut se faire le soir, par exemple, pour prédire les temps de débuts des activités de la journée suivante. Ensuite, lors de la recherche de l'activité entamée, la réduction du nombre d'hypothèses se fait en ne considérant que les activités avec un temps prédit assez proche de l'heure courante.

La prédiction des temps de débuts des activités est utilisée aussi pour répondre au problème d'équiprobabilité. En effet, quand les récents senseurs activés appartiennent à plusieurs modèles d'activités, ces derniers ont la même probabilité d'être effectués. Dans ce cas, l'activité avec le temps prédit le plus proche de l'heure courante peut être considérée comme les plus probables. Elle répond aussi aux erreurs d'initiations qui empêchent l'agent acteur d'amorcer ses activités et permet la prédiction de l'activité qui doit être entamée même si aucun senseur n'est activé.

Une série temporelle (voir l'Annexe B pour plus de détails) est une séquence d'observations prises séquentiellement dans le temps [18] dénotée $(X_t)_{t\in\theta}$ où θ est l'espace de temps. Dans notre cas, chaque modèle d'activité est une série temporelle, les jours d'observations constituent l'espace de temps et les heures de débuts des activités sont les observations. Il faut noter que si une activité est effectuée d'habitude *n* fois dans la même journée, elle est considérée comme *n* différentes activités et chacune d'entre elles est représentée par une série temporelle. En utilisant les séquences fréquentes, homogènes et fermées trouvées lors de la section précédente, un seul parcours de la base de données permet de créer les séries temporelles schématisées dans le Tableau 4-6 et la Figure 4-4.

Tableau 4-6. Les séries temporelles créées

Activité	Jour 1	…	Jour N
Se réveiller	1230	…	0
…	…	…	…
Prendre petit déjeuner	6767	…	6760

Dans le Tableau 3-5, les entiers représentent les temps des débuts des activités (la différence en seconde entre l'heure de début de l'activité est 6h00 du matin), tandis que les 0 indiquent que l'activité n'a pas été effectuée pendant cette journée.

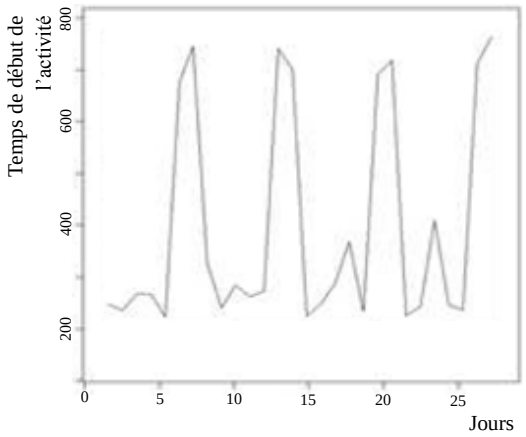


Figure 4-4. Série temporelle représentant une activité

Les techniques des séries temporelles se basent sur la notion d'autocorrélation pour prédire les prochaines valeurs. Cela veut dire que les valeurs successives des séries dépendent les unes des autres sinon les valeurs sont aléatoires et la prédiction des prochaines valeurs est impossible. Nos séries temporelles ne sont pas aléatoires puisque nous avons tous un mode de vie et des habitudes qui font en sorte que nos activités sont effectuées presque en même temps que le jour précédent ou le même jour de la semaine. De plus, les prédictions seront plus précises pour les activités que nous nous intéressons à reconnaître comme *prendre médicament* dont le temps de début est rigoureusement respecté.

En tenant compte de plusieurs paramètres comme la stationnarité, l'autocorrélation et la covariance, nous avons opté pour la technique "autoregressive integrated moving average" (ARIMA) [108] qui est une équation linéaire de temps discret avec bruit de la forme :

$$(1 - \sum_{k=1}^{p} \alpha_k L^k)(1 - L)^d X_t = (1 + \sum_{k=1}^{q} \beta_k L^k)\varepsilon_t$$

où p est l'ordre du modèle autorégressif, q est l'ordre du modèle moyenne mobile, d est l'ordre de différentiation, α_k et β_k sont des paramètres du modèle, ε_t est le bruit blanc et L est l'opérateur de retard : $LX_t = X_{t-1}$

La prédiction des prochaines valeurs avec ARIMA(p, d, q) est effectuée en trouvant les trois paramètres p, d et q qui donnent les meilleurs résultats. Le premier paramètre à déterminer est l'ordre de différentiation d : d est égal à 0 si X_t est stationnaire, sinon d est incrémenté jusqu'à ce que $(1 - L)^d X_t$ devient stationnaire. Pour décider de la stationnarité de la série temporelle, nous utilisons le teste KPSS [109]. Dans

un modèle KPSS, la série temporelle est représentée comme la somme de trois composantes :

$$X_t = \xi_t + r_t + \varepsilon_t$$

$$\text{Avec } r_t = r_{t-1} + u_t$$

où t est une tendance déterministe, r_t est un processus aléatoire, ε_t sont des termes d'erreurs stationnaires et u_t sont des termes d'erreurs avec une variation constante σ_u^2. KPSS utilise ensuite les règles suivantes pour décider de la stationnarité de X_t :

- Si $\xi = 0$ alors X_t est stationnaire aux environs de r_0;
- Si $\xi \neq 0$ alors X_t est stationnaire aux environs d'une tendance linéaire;
- Si $\sigma_u^2 > 0$ alors X_t n'est pas stationnaire.

Une fois d est déterminé et la stationnarité est confirmée, ARIMA cherche les paramètres p et q qui donnent les meilleurs résultats. Pour trouver ces paramètres, nous choisissons ceux qui minimisent le "Akaike's information corrected criterion" (AICC) [110]. La formule pour calculer l'AICC est la suivante :

$$AICC = -2Ln(Likelihood(\hat{\phi}, \hat{\theta}, \hat{\sigma}^2)) + 2n\frac{(p+q+1)}{n-(p+q)-2}$$

où ϕ est une classe de paramètres autorégressifs, $\hat{\theta}$ est une classe de paramètres du composant moyenne mobile, $\hat{\sigma}^2$ est la variance du bruit blanc, n est le nombre d'observations, p est l'ordre du composant autorégressif et q est l'ordre du composant moyenne mobile. (Voir l'Annexe B pour plus de détails)

Après la détermination des trois paramètres d, p et q, ARIMA peut prédire les prochains temps de débuts des activités. Cette prédiction, comme déjà expliquée, est basée sur les anciennes valeurs des jours précédents. S'il y a une exception ou des imprévus le jour même de la recherche de l'activité entamée, nos prédictions ne

prendront pas ces changements en considération. Supposons, par exemple, que l'agent acteur s'est réveillé exceptionnellement une journée en retard d'une heure, les prédictions de cette journée, comme le temps de début de *Prendre petit déjeuner*, ne changeront pas et seront certainement très loin des temps réels. Pour remédier à ce problème, nous avons programmé une deuxième phase de prédiction effectuée quand le temps prédit pour une activité *A* est assez loin du temps réel où elle s'est produite. Nous utilisons le même modèle ARIMA pour la prédiction, sauf que cette fois-ci les séries temporelles sont créées juste pour les activités qui se déroulent juste après l'activité *A*. Les observations d'une de ces séries temporelles $(Y_t)_{t\in\theta}$ sont la différence temporelle entre le temps de début de cette activité et celui de *A* pour les différents jours d'observations. La valeur qui sera prédite pour une activité *i* est donc le délai entre les activités *A* et *i*. Le nouveau temps de début X_i' qui sera prédit pour l'activité *i* est calculé comme suit :

$$X_i' = X_A + Y_i$$

où X_A est le temps réel où l'activité *A* s'est effectuée et Y_i est la prédiction du délai entre les deux activités *A* et *i*.

4.5 La recherche de l'activité entamée

Nous avons divisé le processus de reconnaissance d'activités en trois étapes. La première étape est celle d'apprentissage des comportements normaux de l'agent acteur qui se traduit par la création des modèles d'activités. La deuxième a pour objectif la réduction du nombre d'hypothèses. Elle prédit les temps de débuts des différentes activités qui seront utilisés pour en sélectionner juste celles qui sont assez proches de l'heure de la recherche de l'activité entamée. Pour ces deux premières étapes, nous

n'avions pas de contraintes sur le temps d'exécution des deux algorithmes utilisés puisque les deux étapes ne sont pas en temps réel; la première est censée être exécutée une fois tous les deux mois pour prendre en considération les changements d'habitudes de l'agent-acteur, alors que la deuxième est censée être exécutée une fois toutes les nuits quand l'occupant de la maison est endormi pour prédire les temps de débuts des activités du lendemain. La troisième étape est celle de recherche de l'activité entamée. Elle est plus délicate que les deux autres parce qu'elle doit s'exécuter en temps réel. Cette étape doit aussi répondre au problème de l'équiprobabilité et aux erreurs d'initiations. Pour répondre à tous ces problèmes, nous avons choisi d'utiliser un réseau bayésien après la réduction du nombre d'hypothèses. Les fréquences des activités pour le même jour de la semaine (confidence journalière) ainsi que les durées entre leurs temps de débuts prédits et l'heure de la recherche sont utilisées pour décider des probabilités initiales des activités sélectionnées. Ce sont ces probabilités qui sont utilisées pour prédire l'activité la plus probable et la proposer comme réponse aux erreurs d'initiations pour le problème de l'équiprobabilité. L'algorithme 4-4 détaille l'approche utilisée.

Algorithm Réseau Bayésien pour la prédiction des activités

Entrees: L'ensemble des modèles d'activités A
Sorties: Les activités sélectionnées avec leurs probabilités

1: **Pour chaque** activité $a \in A$
2: $a.PST \leftarrow$ temps de début prédit pour a
3: **Si** ($|a.PST -$temps actuel $Ct| < \epsilon$) **Alors**
4: Ajouter a à SA
5: Probabilité $a.P = (confidencejournalière)_{normalisee}$
6: **Fin Si**
7: **Fin Pour**
8: **Pour chaque** $a \in SA$
9: $a.P = Pa * (\frac{1}{|a.PST - Ct|})_{normalisee}$
10: **Fin Pour**
11: **Si** (une activité $a' \in SA$ est détectée) **Alors**
12: Enlever a' de SA
13: **Si** ($|a'.PST - a'.DST| > \epsilon'$) **Alors**
14: **Pour chaque** $a \in SA$
15: créer Y (série temporelle des différences des temps de début)
16: Predire PY (le temps entre a et a')
17: Nouveau $a.PST = a'.DST \pm PY$
18: **Fin Pour**
19: **Pour chaque** $a \in SA$
20: $a.P = Pa * (\frac{1}{|a.PST - Ct|})_{normalisee}$
21: **Fin Pour**
22: **Fin Si**
23: **Fin Si**
24: Attendre(S secondes)
25: **Aller à** 11
26: **retourner** AM

Algorithme 4-4. Réseau bayésien pour la prédiction des activités.

Les sept premières lignes de cet algorithme sélectionnent parmi tous les modèles d'activités ceux dont le temps prédit est assez proche de l'heure courante et réduisent par ce fait le nombre d'hypothèses. En même temps, une probabilité initiale est calculée pour chaque activité sélectionnée en se basant sur la fréquence d'occurrence de cette activité dans cette journée. Le reste de l'algorithme détaille le cas où le temps prédit est loin du temps détecté de la même activité. Les probabilités, dans ce cas, sont recalculées en se basant cette fois-ci sur les prédictions des séries temporelles composées des différences temporelles entre les temps de débuts de ces activités et le temps de début de l'activité détectée.

La recherche de l'activité entamée est effectuée en mettant à jour les probabilités initiales du réseau bayésien selon si l'activité contient ou non le récent senseur activé. Le fait que l'activité comporte le senseur activé augmente sa probabilité d'être celle entamée, tandis que son absence diminue sa probabilité sans l'annuler pour prendre en compte les différentes erreurs susceptibles d'être commises par l'agent-acteur.

4.6 Prédiction des temps d'activation des senseurs

La majorité des approches proposées pour la recherche technologique se focalisent sur la problématique de la reconnaissance d'activités. Si leurs testes et validations prouvent leur efficacité dans la reconnaissance de l'activité entamée, ils ne disent rien sur la capacité de ces approches à gérer le problème de détection des erreurs et celui du choix du moment idéal pour proposer de l'aide, qui est le but ultime de l'assistance technologique. Par exemple, notre dernière approche arrive à reconnaître l'activité entamée, mais elle est incapable de choisir le moment opportun pour proposer de l'aide puisque les modèles d'activités créés ne contiennent aucune information temporelle sur le délai habituel entre deux adjacents senseurs.

Dans notre article présenté à la conférence PETRA [19], nous avons étudié ce point très important. Il faut noter que la nature des modèles d'activités, créés par notre approche, permet déjà de répondre à ce problème en utilisant la borne supérieure de l'intervalle temporel entre deux adjacents senseurs. En effet, si le deuxième senseur s_2 ne s'est pas activé après t_{max} de l'activation de s_1, t_{max} étant la borne supérieure de l'intervalle temporel entre s_1 et s_2, une erreur est signalée et un effecteur incitera l'agent acteur à utiliser s_2. Malheureusement, en utilisant la borne supérieure, nous avons

remarqué qu'une activité comme *préparer café* durera environ 5 heures si l'agent acteur a besoin d'assistance pour chaque action qui compose cette activité.

La première solution à laquelle nous avons pensé a été inspirée de la prédiction des temps de début des activités et vise à prédire, cette fois-ci, le temps d'activation du prochain senseur. Quand l'activité entamée est reconnue et après la détection du dernier senseur activé, le prochain senseur à s'activer s_p est connu. Il suffit donc de créer une série temporelle $(X_t)_{t\in\theta}$ où θ est l'ensemble des jours d'observation et X_i est le temps d'activation du senseur s_p de l'activité reconnue dans la journée i. Le même modèle ARIMA est ensuite utilisé pour la prédiction des prochaines valeurs. Comme il sera montré à la section de validation, cette solution diminue le temps de déroulement des activités avec assistance de presque la moitié.

Les bons résultats obtenus avec ARIMA nous ont encouragés à aller plus loin dans notre étude pour améliorer encore plus nos prédictions. Nous avons, encore une fois, utilisé la propriété de nos modèles d'activités qui stipule que le prochain senseur s_2 s'active toujours après l'activation du senseur s_1, ce qui peut montrer que les prochaines valeurs de s_2 ne sont pas seulement influencées par ses anciennes valeurs, mais aussi par les anciennes valeurs de s_1. Cette remarque peut être formalisée en utilisant une série temporelle bivariée $Y_t = (y_{1_t}, y_{2_t})$ où y_{1_t} est la série temporelle représentant les temps d'activations du senseur s_1 de l'activité reconnue dans les différents jours d'observation et y_{2_t} celle de s_2. Y_t est ensuite modélisée en utilisant un modèle vectoriel autorégressif (VAR) qui est une extension naturelle du modèle autorégressif univarié au modèle dynamique des séries temporelles multivariées [108]. La forme basique d'un modèle vectoriel autorégressif d'ordre p VAR(P) [18] est donné par :

$$Y_t = c + \pi_1 Y_{t-1} + \pi_2 Y_{t-2} + \cdots + \pi_p Y_{t-p} + \varepsilon_t$$

où π_i est une matrice de coefficients (nn) et ε_t est un vecteur de bruit blanc.

Pour un modèle VAR(P) bivarié, l'équation a la forme :

$$y_{1_t} = c_1 + \pi^1_{1_1} y_{1_{t-1}} + \pi^1_{1_2} y_{2_{t-1}} + \pi^2_{1_1} y_{1_{t-2}} + \pi^2_{1_2} y_{2_{t-2}} + \varepsilon_{1_t}$$

$$y_{2_t} = c_2 + \pi^1_{2_1} y_{1_{t-1}} + \pi^1_{2_2} y_{2_{t-1}} + \pi^2_{2_1} y_{1_{t-1}} + \pi^2_{2_2} y_{2_{t-1}} + \varepsilon_{2_t}$$

Le processus de prédiction avec VAR commence par la sélection de l'ordre de retard p. La sélection se fait en essayant différents ordres $p = 0, \ldots, p_{max}$ et en choisissant la valeur de p qui minimise les trois critères suivants :

$$Akaike\ (AIC) : AIC(p) = \ln\left|\tilde{\sum}(p)\right| + \frac{2}{T} pn^2$$

$$\text{Schwarz} - \text{Bayesian (BIC)} : \text{BIC(p)} = \ln\left|\tilde{\sum}(p)\right| + \frac{LnT}{T} pn^2$$

$$Hannan - Quinn\ (HQ) : HQ(p) = \ln\left|\tilde{\sum}(p)\right| + \frac{2nLnLnT}{T} pn^2$$

où $\tilde{\sum}(p)$ est la matrice de covariance résiduelle sans degrés de liberté de correction d'un modèle VAR(p) et T la taille de l'échantillon.

Après la sélection de p, les coefficients du processus VAR(p) peuvent être estimés d'une façon efficiente en appliquant les moindres carrés séparément à chacune des dernières équations.

Finalement, ces coefficients sont utilisés avec les deux dernières valeurs de la série temporelle pour prédire les valeurs futures. Il faut noter que pour prédire des valeurs à un horizon *h*, les prédictions sont calculées successivement de *t+1* jusqu'à *h*.

4.7 Conclusion

Durant ce chapitre, nous avons présenté une solution complète à l'assistance technologique en abordant ses différentes étapes, surtout l'étape de reconnaissance d'activité qui présente le plus grand défi de ce processus. Notre approche est une approche basée sur les objets qui travaille directement sur les données générées par les senseurs d'une manière non supervisée. Elle est divisée en cinq étapes.

Dans une maison intelligente, une centaine de senseurs y sont installés. Chaque senseur d'entre eux envoie, comme dans le cas du LIARA, deux mesures par seconde. Pour une phase d'apprentissage efficace, il faut observer l'occupant de la maison pendant un mois ou plus. Le volume de données généré par les senseurs est donc très grand. Même les techniques de la fouille de données, connues pour leur capacité à gérer une grande quantité de données, sont incapables d'exploiter d'une manière efficace ces données. Alors, la première étape de notre approche vise la réduction des données en ne considérant que les senseurs activés au lieu de tous les senseurs.

La nouvelle base de données, obtenue de la première étape, est composée d'une séquence de senseurs activés par jour. La deuxième étape consiste à trouver parmi ces séquences, les sous-séquences fréquentes et fermées qui représentent les modèles d'activités. Pour créer ces modèles, nous avons créé un nouvel algorithme qui, itérativement, coupe entre les couples de senseurs adjacents non fréquents et remplace chaque couple fréquent par un indice.

La reconnaissance d'activités consiste à trouver, parmi les modèles d'activités créés à l'étape précédente, celui qui explique le mieux les récents senseurs activés. Chercher parmi tous les modèles d'activités créés prendrait beaucoup de temps pour une

application en temps réel. Alors, la troisième étape essaie de réduire le nombre d'hypothèses en cherchant juste parmi les modèles les plus probables. Pour déterminer les probabilités des modèles, une série temporelle est créée pour chaque modèle comportant l'historique de ces temps de début. Ensuite, son temps de début du lendemain est prédit. Les modèles les plus probables sont donc ceux avec un temps prédit proche du moment de la reconnaissance.

La quatrième étape, qui fait toujours partie de l'étape de la reconnaissance d'activités, a pour objectif de trouver l'activité entamée parmi les modèles d'activités les plus probables. La solution adoptée dans cette étape consiste à créer un réseau bayésien dont les probabilités initiales sont calculées selon la différence temporelle entre le temps prédit pour le modèle et l'heure courante. Ces probabilités sont ensuite mises à jour dépendamment si le modèle comporte ou non les récents senseurs activés.

Le but ultime de notre approche est d'assister l'agent acteur au besoin. Alors, une fois l'activité reconnue, les erreurs doivent être détectées pour proposer de l'aide au moment opportun. Dans la cinquième étape, le temps d'activation du prochain senseur de l'activité entamée est prédit en utilisant les séries temporelles bivariées. Quand le temps réel dépasse le temps prédit sans que le senseur soit activé, on considère qu'il y a erreur et un message d'aide doit être envoyé via l'effecteur adéquat.

Chapitre 5

5 Validation

Dans le chapitre précédent, nous avons proposé une approche qui propose des solutions aux différentes étapes de l'assistance technologique; l'observation et la préparation des données, l'apprentissage des comportements normaux et la reconnaissance d'activités ainsi que la détection d'erreurs et le choix du moment opportun pour proposer de l'aide. Dans ce chapitre, nous décrivons et analysons les tests effectués afin de valider notre approche. Pour comprendre et bien évaluer ces tests, il est nécessaire d'avoir une idée bien précise sur la maison intelligente où les données ont été collectées, qui est le Laboratoire d'Intelligence Ambiante pour la Reconnaissance d'Activités (LIARA). Avant de détailler les infrastructures expérimentales du LIARA, nous commençons par une étude plus approfondie sur la conception des maisons intelligentes afin de justifier les choix technologiques au LIARA.

5.1 Conception d'une maison intelligente

Une maison était considérée comme intelligente, quand elle comportait des outils technologiques qui permettaient d'automatiser des tâches aussi simples que soient-elles. Par exemple, une maison où un système de contrôle de température est installé, qui

adapte automatiquement la température des différentes chambres de la maison selon la position de l'agent acteur dans la maison ou selon l'apprentissage des périodes que passe l'agent acteur dans la maison, était considérée comme maison intelligente. De nos jours, les maisons intelligentes ont évolué et elles sont utilisées pour atteindre des objectifs plus généraux comme la réduction de consommation d'énergie, l'automatisation des tâches ménagères ou l'amélioration du confort dans la maison, etc [5]. Dans notre cas d'étude, l'objectif de la maison intelligente est d'offrir une assistance technologique à l'occupant de la maison pour lui permettre de vivre plus longtemps, d'une façon autonome, chez lui. Elle permet ainsi de soulager les ressources humaines et financières du système de santé et d'éviter la relation souvent complexe qui s'installe entre la personne aidante et l'occupant de la maison. La maison intelligente peut aussi établir et envoyer des rapports aux médecins pour évaluer l'état de santé de la personne assistée.

La maison intelligente comporte trois éléments principaux :

1. Des senseurs pour effectuer différentes mesures dans la maison;
2. Un ou plusieurs ordinateurs capables de collecter et d'analyser ces mesures;
3. Des effecteurs pour proposer de l'aide à la personne assistée.

La conception d'une telle maison ne s'effectue pas seulement en choisissant pour chaque élément ceux qui servent le mieux l'objectif général, mais plusieurs autres critères doivent être considérés comme l'architecture et le type de communication entre les senseurs et l'ordinateur principal, le type de la base de données qui sera créée, le langage qui sera utilisé pour l'exploiter, etc. Dans la suite de cette section, nous allons expliquer les choix importants qui ont été faits lors de la conception du LIARA et des différents critères qui ont été considérés. Il faut préciser que les critères utilisés ont été inspirés

d'une étude approfondie des différentes maisons intelligentes déjà existantes comme MavHome [111], eHome [112], DOMUS [113], "gator tech smart house" [114], IATSL [115], "Institute for Infocomm Research" [116] et "House_n project" [117].

5.1.1 Les senseurs

Figure 5-1. Ensemble de senseurs

Comme l'indique la Figure 5-1, il existe sur le marché une grande variété de types de senseurs qui permettent d'effectuer différentes mesures à l'intérieur d'une maison. Le choix parmi ces senseurs se fait primordialement selon l'objectif général pour lequel la maison intelligente est conçue et selon d'autres critères très importants dont :

- **Le coût** : l'un des objectifs de l'assistance technologique dans une maison intelligente est de réduire les coûts engendrés par l'assistance traditionnelle. Il est donc impératif que cette solution soit abordable afin de profiter à un grand large public de différentes classes sociales. Il faut donc choisir des senseurs pas chers pour minimiser le coût global de la maison intelligente. Il faut aussi préciser qu'il faut tenir compte des quantités de senseurs requises lors de la comparaison des coûts. Par exemple, si le but visé par l'installation d'une

caméra est d'estimer la position de l'agent-acteur dans la maison, le prix d'une seule caméra doit être comparé au prix de quatre ou cinq détecteurs de mouvements.

- **La robustesse** : le choix des senseurs ne doit pas se faire uniquement sur le prix, sinon on peut finir par avoir un système de senseurs douteux où des senseurs échoueront parfois à effectuer ou à envoyer leurs mesures. La robustesse est le critère qui garantit que chaque senseur est capable d'effectuer et d'envoyer ces mesures à la fréquence demandée. Si on prend les Tags RFID [118] comme exemple, il faut savoir qu'il en existe deux types; les Tags passifs et les Tags actifs. Les Tags actifs sont plus précis que les Tags passifs, mais ils sont moins robustes puisqu'ils comportent des batteries qui les empêchent de fonctionner une fois déchargées.
- **La précision :** lors de l'analyse des mesures envoyées par les senseurs, la précision de ces mesures influence grandement les résultats obtenus. En plus de la précision des mesures entre les différentes marques des senseurs, d'autres paramètres permettent d'augmenter la précision de ces mesures. Par exemple, pour suivre la position d'un objet dans la maison on peut y installer un Tag RFID. L'analyse de la force du signal entre le Tag RFID et une antenne RFID permet d'estimer la distance entre les deux sans donner aucune information sur l'emplacement de l'objet par rapport à l'antenne. L'utilisation d'un deuxième et troisième senseur donne, grâce à la technique de triangulation [118], plus de précision sur l'emplacement de l'objet par rapport aux trois antennes, ce qui permet de localiser sa position dans l'appartement.

- **L'invasion :** avant de choisir un senseur, il faut aussi penser aux conséquences de son utilisation sur la vie quotidienne de l'agent acteur. Un senseur non invasif effectue ses mesures sans que l'agent acteur se rende compte de son existence ni de modifier ses habitudes pour s'adapter à son utilisation. Rappelons que le type d'assistance que nous avons choisi est une assistance à l'insu où les senseurs travaillent d'une façon transparente à l'agent-acteur. Les senseurs portables sont des senseurs intéressants qui permettent la détection de quelques actions basiques de l'agent acteur, mais la nécessité de les porter tous les jours les rend parmi les senseurs les plus invasifs.
- **L'installation :** quand on parle de l'installation des senseurs, on parle de l'installation primaire et des calibrages qui s'en suivent. L'installation de certains types de senseurs peut être une opération très délicate. Si, par exemple, l'installation des Tags RFID sur les différents objets est assez simple, le choix du nombre et de l'emplacement des antennes doit se faire selon une étude très approfondie. La facilité d'installation est un critère très important dans notre cas d'assistance technologique parce que l'objectif est de permettre à la personne assistée de vivre plus longtemps chez lui, ce qui veut dire que sa maison, qui n'était pas conçue depuis le départ comme une maison intelligente, est transformée pour y installer les différents senseurs.
- **La complexité des données :** la complexité des données est un critère qu'on ne peut ignorer. La majorité des applications des maisons intelligentes sont des applications en temps réel. Cela veut dire que les données envoyées par les senseurs doivent être analysées et interprétées le plus rapidement possible pour

permettre une réaction assez rapide comme l'assistance au moment opportun. Les données complexes sont donc à éviter, si nous avons le choix, parce qu'au lieu d'être des informations qui enrichissent l'analyse, ils ajoutent une complexité supplémentaire qui peut retarder ou bloquer la réponse de la maison intelligente.

Le Tableau 5-1 résume les types de senseurs les plus importants et leurs caractéristiques par rapport aux critères mentionnés.

Tableau 5-1. Caractéristiques des senseurs

	Coût	Robustesse	Précision	Invasion	Installation	Complexité
Accéléromètres	A	A	B	A-B	B	B-C
Capteurs de forces	B	A	B	A	B-C	B-C
Senseurs ultrasoniques	B-D	B	B-D	A	C-E	C-D
Senseurs de température	A	A	B	A	C	A
capteurs de débit	B	B	B	A	B-C	B
Senseurs de lumière	B	B	B	A	C	A
Tapis de pression	C	B	A	B-C	A	A
Détecteurs de mouvements	A	A	C	B	A	A
Contacts électromagnétiques	A-B	A	A	A	B	A
RFID	B-C	B	B-D	B	C-E	B-C
Analyseurs d'énergie	B-C	A	B-C	A	A	C
Microphones	B-D	A	B	C	B-C	D
Caméras Vidéo	C-E	B-C	B	E	B-D	E

5.1.2 L'architecture du système de senseurs

Après la sélection des senseurs à utiliser, selon les critères expliqués dans la section précédente, il faut décider du type de l'architecture du système de senseurs. Il

existe deux types d'architectures; l'architecture centralisée où les senseurs sont passifs et ne font qu'envoyer leurs mesures à un serveur afin d'y être stockées puis analysées et l'architecture décentralisée où les senseurs sont actifs et communiquent entre eux pour prendre quelques décisions et collaborer sur des services. Certes, l'architecture décentralisée a beaucoup d'avantages comme le soulagement du trafic réseau et la rapidité de réaction, mais elle souffre en contrepartie de quelques inconvénients comme la gestion de la complexité et l'adaptation après l'ajout ou la disparition d'un élément suite à la décharge d'une batterie par exemple. Plusieurs recherches explorent cette architecture [119], mais au LIARA nous nous concentrons d'abord sur la création d'une solution centralisée fonctionnelle tout en envisageant d'y intégrer des modules décentralisés plus tard.

5.1.3 Les effecteurs

Le choix des senseurs adéquats pour la collecte des données qui seront analysées pour la reconnaissance de l'activité entamée est une étape très importante dans le processus de l'assistance technologique, mais cette assistance ne peut réussir sans des effecteurs capables de transmettre les messages d'aides à l'agent acteur. Si nous avons défini les critères utilisés pour le choix des senseurs, il est plus délicat de définir des critères pour le choix des effecteurs. En effet, le meilleur effecteur est celui qui permet à l'agent acteur de bien saisir l'action qu'on veut lui proposer. Il dépend donc du profil de l'agent-acteur et du type de l'action proposée. Par exemple, si la personne assistée souffre d'une aphasie de Wernicke (difficultés importantes de comprendre ce qui est dit et ce qui est écrit), il est inutile d'utiliser des effecteurs audio. Nous pensons que l'analyse des profils pour le bon choix des effecteurs est un peu loin du travail proposé dans cette

thèse. Pour cette raison, nous n'allons pas détailler ce point et nous référons ceux qui veulent plus d'information sue ce sujet aux travaux réalisés par des membres de notre équipe au LIARA [6], [120].

5.1.4 Le logiciel

Le but ultime des maisons intelligentes est d'assister son occupant. Le choix de tous les éléments de la maison intelligente doit donc se faire pour aboutir à une assistance réussie et efficace. La réussite de l'assistance technologique ne se mesure pas seulement par le taux des erreurs détectées et des actions prédites, mais aussi par le temps entre la détection d'une erreur et l'envoi du message d'aide. En effet, un message d'aide qui n'est pas reçu au bon moment fera plus de mal que du bien à la personne assistée. Par exemple, si la personne assistée commet une erreur et après un certain temps elle continue son activité. Ensuite, après trois actions de l'activité elle reçoit le message d'aide lié à l'erreur commise. Ce message n'aura plus aucune utilité et mettra le doute et la confusion dans l'esprit de la personne assistée. Pour que l'assistance technologique soit efficace, elle doit être instantanée du point de vue de l'utilisateur, ce qui veut dire que l'application programmée pour la réception des mesures des senseurs, pour les analyser afin d'inférer l'activité entamée et pour la détection d'erreurs et l'envoi du message d'aide, en cas de besoin, doit s'exécuter en temps réel.

La première contrainte à l'exécution de cette application en temps réel est la complexité des données envoyées par les senseurs. Ces données sont hétérogènes et peuvent nécessiter des précalculs pour devenir utilisables. Par exemple, un senseur électromagnétique placé sur une porte est un senseur booléen qui peut envoyer un 0 ou 1 (ON, OFF) qui montre si la porte est ouverte ou fermée. Alors que pour un Tag RFID, qui

est un senseur numérique, les données reçues sont au nombre des antennes RFID existantes dans la maison et indiquent la force du signal entre le Tag et chaque antenne. Ces forces de signaux peuvent être transformées en une position dans la maison grâce à des calculs assez complexes [121].

Pour obtenir des données homogènes et pour ne pas avoir à inclure des calculs de bas niveau dans les applications, une couche logicielle est programmée pour la transformation des données brutes reçues à des données utilisables par les applications qu'on appelle de haut niveau.

Le fait que les approches proposées pour l'assistance technologique ne travaillent pas directement sur les données brutes envoyées par les senseurs, mais sur des données homogènes transformées par une couche logicielle ne suffit pas pour garantir une exécution en temps réel. Il faut que ces approches soient conçues en respectant cette contrainte. Dans notre cas par exemple, nous avons divisé notre approche en trois étapes principales : la création des modèles d'activités qui n'est pas une application en temps réel et peut être exécutée tous les deux mois pour considérer les nouvelles tendances de l'occupant de la maison, la prédiction des temps de début des différentes activités qui n'est pas aussi une application en temps réel et peut être exécutée toutes les nuits pour prédire les temps de début des activités du lendemain et la recherche de l'activité entamée qui est une application en temps réel que nous avons programmée en utilisant un réseau bayésien assez rapide dans ces calculs et dans l'obtention des probabilités finales.

5.2 Notre maison intelligente LIARA

Grâce à un appui de la Fondation Canadienne pour l'Innovation (FCI) et en se basant sur l'étude de conception des maisons intelligentes détaillée dans la section précédente, Le Laboratoire d'Intelligence Ambiante pour la Reconnaissance d'Activités LIARA a pu être construit en intégrant les récentes avancées technologiques. Situé au troisième étage du pavillon principal de l'Université du Québec À Chicoutimi (UQAC), le LIARA est une maison de 100 mètres carrés dotée d'une centaine de senseurs et d'effecteurs. Il est conçu selon une stratégie qui vise à ne pas observer directement l'occupant de la maison, mais plutôt observer les changements dans son environnement pour préserver son intimité. Pour cette raison, les caméras ne figurent pas parmi les différents types de senseurs utilisés comme les contacts électromagnétiques, les tapis de pression, les capteurs infrarouges, les capteurs de lumière, différents capteurs de température, un analyseur de puissance intelligent, les capteurs à ultrasons et plusieurs Tags avec huit antennes RFID. Au LIARA, différents types d'effecteurs sont installés comme des Haut-parleurs IP, des lumières et des diodes électroluminescentes (DEL), un cinéma maison, une télévision à écran plat HD et un Ipad Apple. Pour avoir une idée plus précise sur le LIARA, la Figure 5-2, affichée dans la page suivante, présente différentes parties de cette maison intelligente. L'image principale de cette Figure montre la cuisine où trois antennes RFID captent les signaux envoyés par les Tags RFID installés sur les différents objets comme celui installé sur la tasse de l'image en bas à gauche. Toutes les mesures des senseurs sont envoyées à l'ordinateur central, affiché en haut à droite de la Figure, qui est un Dell serveur lames qui commande le traitement de ces informations. Le Ipad installé sur le réfrigérateur a plusieurs utilisations : tester les

équipements, contrôler les expériences menées au LIARA en permettant de les sauvegarder, les rejouer, etc. il peut-être aussi utilisé comme effecteur en jouant une vidéo d'aide quand la personne assistée est dans la cuisine. Les autres images de la Figure 4-2 montrent la salle à manger, une petite bibliothèque en avant de la salle de bain et une télévision HD contrôlée par un système de contrôle multimédia (AMX) qui contrôle aussi le lecteur de DVD et les Haut-parleurs IP.

Figure 5-2. Le LIARA

5.2.1 L'architecture du système de senseurs au LIARA

L'architecture du système de senseurs au LIARA est une architecture centralisée où tous les senseurs sont passifs et ne font qu'envoyer toutes leurs mesures à l'ordinateur central. Lors de sa conception, l'équipe du LIARA a opté pour une solution assez robuste pour supporter son utilisation quotidienne et intensive. Le choix du matériel était une étape très délicate parce qu'il devait répondre à trois critères très importants : une qualité

industrielle pour des mesures précises et une durée de vie assez longue qui minimisera la maintenance, un coût aussi faible que possible et une installation assez facile pour que la transformation de la maison de la personne à assister en une maison intelligente semblable au LIARA soit abordable et facile. De plus, pour éviter que le système de senseurs tombe en panne, les senseurs sont divisés en plusieurs groupes, comme schématisés dans la Figure 5-3, ce qui permet à certains groupes de fonctionner tandis que les autres sont bloqués à cause d'un ou plusieurs problèmes.

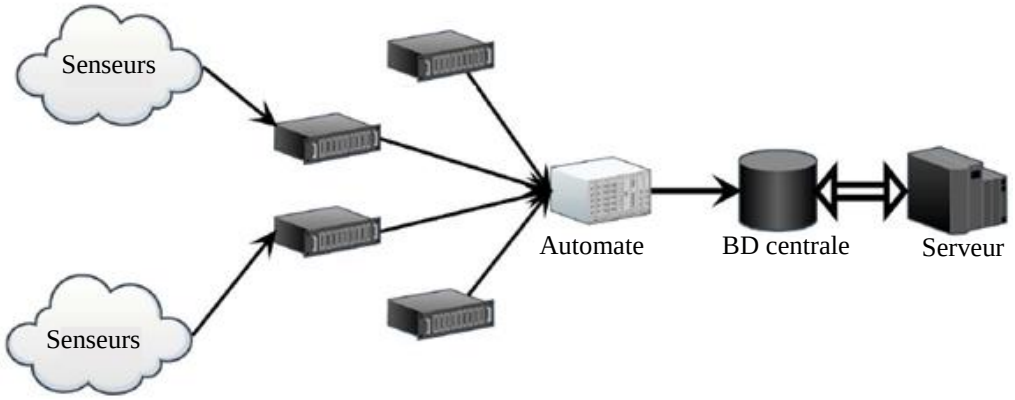


Figure 5-3. L'architecture du système de senseurs

Chaque groupe est relié à un automate APAX-5570 qui a pour rôle de collecter les mesures des différents groupes, en temps réel, et de les envoyer à une base de données sur le serveur. L'utilisation de l'automate a pour avantage de pallier les problèmes reliés aux incompatibilités de communication entre les différentes normes adoptées par les fabricants des senseurs.

5.2.2 Le logiciel au LIARA

Comme expliquée dans la section précédente, une couche logicielle est programmée pour la transformation des données brutes reçues de l'automate à des

données utilisables par les applications de haut niveau. La Figure 5-4 montre la conception de cette couche logicielle au sein du LIARA.

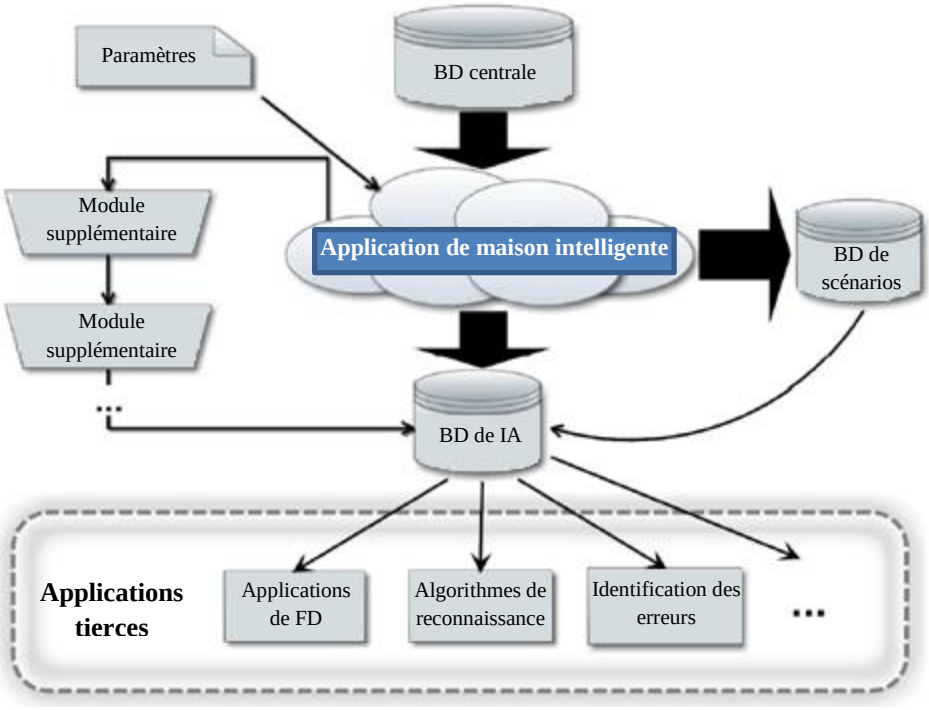
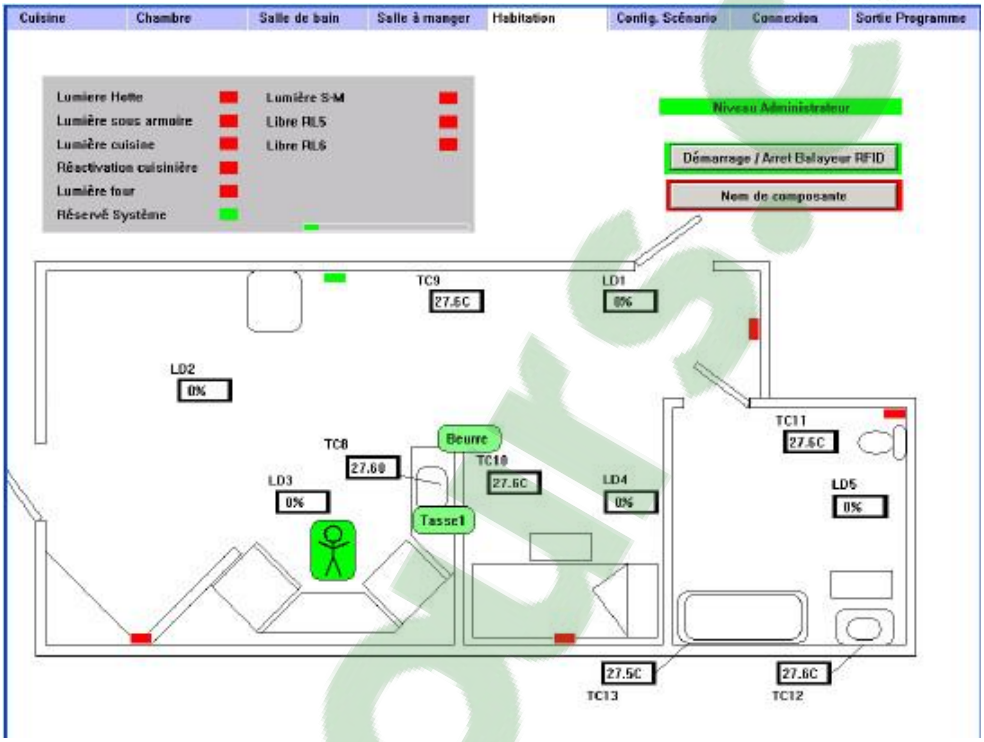


Figure 5-4. La couche logicielle au LIARA

La base de données principale enregistre les données reçues de l'automate. Ensuite, la couche logicielle recopie et transforme certaines données pour qu'elles deviennent homogènes créant ainsi une deuxième base de données que nous appelons base de données pour l'intelligence artificielle (BDIA). Cette couche comporte plusieurs paramètres modifiables pour mieux s'adapter aux différentes expériences menées au LIARA. Elle permet l'intégration de plusieurs modules pour la personnalisation de la transformation des données, ce qui permet aux étudiants et aux chercheurs de modifier l'architecture et le contenu de la BDIA. Elle permet aussi de sauvegarder les expériences et de les rejouer en transférant les données sauvegardées dans la BD de scénario à la BDIA. En fin, les approches proposées travaillent directement sur la BDIA sans avoir à

se préoccuper de l'hétérogénéité et de la complexité des données éliminées par la couche logicielle.

Afin de faciliter les tests au LIARA, un logiciel de visualisation a été intégré à la couche logicielle. La Figure 5-5 représente différentes captures d'écran de ce logiciel.



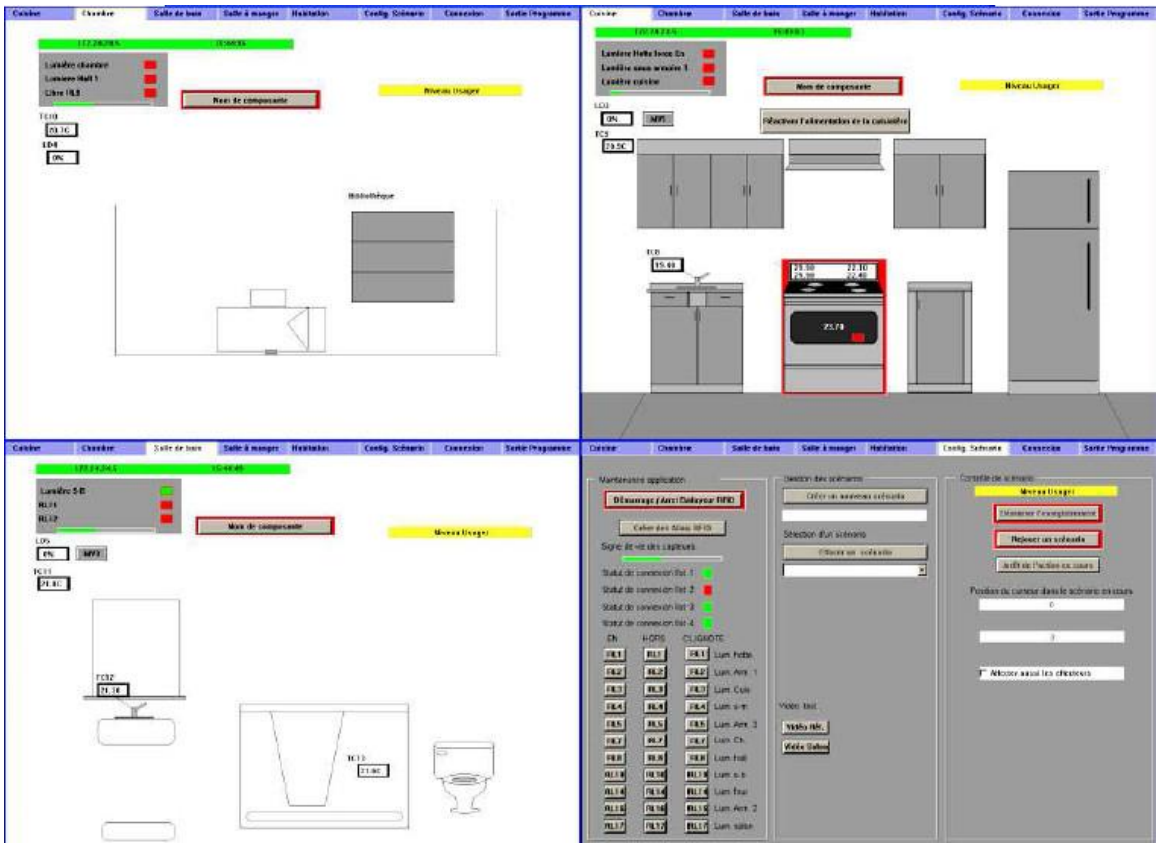


Figure 5-5. Différentes captures d'écran du logiciel de visualisation

Ce logiciel permet de visualiser toute la maison intelligente ou juste des parties. Il affiche aussi l'état (actif ou inactif) de certains senseurs booléens comme les capteurs infrarouges et les capteurs de lumière. Il permet aussi d'estimer les positions des objets sur lesquels des Tags RFID sont installés ainsi que la position de l'occupant de la maison. Cet affichage est d'une utilité intéressante parce qu'il permet de suivre graphiquement les expériences menées au LIARA, ce qui facilite la détection et la compréhension des raisons des erreurs rencontrées au cours des expériences.

5.3 Validation de notre approche

Lors de la validation de notre ancienne approche, nous ne possédions pas de données réelles enregistrées au LIARA après l'observation d'un malade d'Alzheimer

pendant une période assez longue (plus que quatre semaines). Alors nous avons utilisé une des bases de données disponibles sur Internet, et plus précisément celle de Kasteren [26] qui a été choisie parce que leur habitat intelligent ressemble beaucoup au nôtre. Jusqu'à ce jour et pour des raisons éthiques, nous n'avons pas pu inviter un patient de la maladie d'Alzheimer à venir vivre au LIARA afin de créer notre propre base de données. Pour valider notre nouvelle approche, nous ne pouvons pas utiliser la base de données de Kasteren parce que notre approche exploite plusieurs informations temporelles qui n'existent pas dans cette base de données. De plus, nous voulions tester si notre laboratoire est prêt pour accueillir une personne pour y vivre et que le matériel ainsi que le logiciel sont bien fonctionnels. Alors, nous avons décidé de créer notre propre base de données en simulant la procédure du réveil au LIARA par un étudiant volontaire. Comme notre approche vise à observer une personne pendant les deux premières années de son atteinte de la maladie d'Alzheimer, nous avons demandé à cet étudiant d'effectuer, pendant vingt-huit jours, la procédure du réveil à sa façon tout en commettant quelques erreurs reliées à cette maladie comme les erreurs d'initiation, de réalisation, etc.

L'étudiant arrivait chaque jour à cinq heures du matin. Il se mettait au lit puis programmait le réveil selon le jour de la semaine, la fin de semaine le réveil est programmé plus tard que d'habitude. Après, il se réveille, utilise la salle de bain puis se lave les mains et souvent il prend une douche. Ensuite, il se prépare une tasse de café et quitte la maison. Parmi les erreurs préméditées qu'il commettait, il commençait des activités sans les finir, il changeait l'ordre des actions dans une activité ou les sauter carrément.

Pendant qu'il exécutait sa procédure, nous utilisions la couche logicielle pour enregistrer cette procédure et nous récupérions en même temps la base de données créée par cette couche (BDIA). Il faut signaler que nous avons modifié les paramètres de cette couche et nous avons ajouté un module pour adapter BDIA à nos besoins. Par exemple, nous avons fait en sorte que cette couche traite deux mesures de chaque senseur par seconde, sachant que nous pouvions aller jusqu'à cinq mesures par seconde, mais le peu de précision ajoutée ne valait pas la difficulté engendrée par le grand volume de données générées. Le module ajouté, traite principalement les Tags RFID en changeant les forces de signaux de chaque Tag par rapport aux quatre antennes RFID à une position estimée dans la maison.

Après vingt-huit jours d'enregistrement des différentes expériences, nous avons fusionné les vingt-huit bases de données créées pour obtenir une seule base de données BDIA qui englobe toutes les expériences. BDIA ainsi obtenu comprend un très grand volume de données qui la rend inexploitable. Pour avoir une idée sur la taille de cette base de données, il faut savoir que chaque senseur des 112 senseurs du LIARA envoie deux mesures par secondes et que chaque expérience des 28 que nous avons enregistrées dure entre 1h30 et 2 heures. Pour cette raison, notre approche commence par transformer BDIA à une nouvelle base de données (BDSA) composée de 28 lignes, une ligne par jour, et ne comporte que les senseurs activés avec leurs temps d'activation. C'est cette dernière base de données, BDSA, qui a été utilisée pour la validation de notre approche. Comme notre approche est divisée en trois distinctes étapes, nous avons décidé de valider l'algorithme principal de chaque étape séparément : la création des modèles d'activités, la

prédiction des temps de début de chaque activité et la prédiction du temps d'activation du prochain senseur de l'activité entamée.

5.3.1 Validation de la création des modèles d'activités

Pour cette étape de création des modèles d'activités, nous avons créé un nouvel algorithme qui fait partie du domaine d'extraction de motifs fréquents "frequent pattern mining". Nous voulions le comparer aux autres algorithmes de son domaine, mais sa particularité à trouver des séquences parfaitement ordonnées et de créer un intervalle temporel entre chaque deux éléments de la séquence a rendu la comparaison impossible. Alors, pour le valider, nous avons pensé à le tester sur différentes bases de données en évaluant le nombre de motifs fréquents trouvé et le temps d'exécution. Malheureusement, nous n'avons pas pu trouver des bases de données qui ressemblent à BDSA qui ne comporte que les senseurs activés avec leurs temps d'activation. Nous avons donc décidé de générer des données synthétiques pour la validation. Pour que ces données soient assez significatives, nous avons étudié le temps d'exécution dans le pire cas pour trouver les éléments qui l'influence.

Le pire cas arrive quand après chaque passage de la BD, aucune division n'est faite et chaque ligne est diminuée d'un seul élément, ce qui veut dire que toute la base de données est composée d'un seul même élément qui se répète N fois dans la plus longue ligne. Si le nombre de lignes est n et la fréquence minimale est F, le temps d'exécution Te peut être calculé selon la formule suivante :

$$Te = 2n\frac{N(N+1) - F(F+1)}{2}$$

Cette formule est trouvée selon l'analyse suivante :

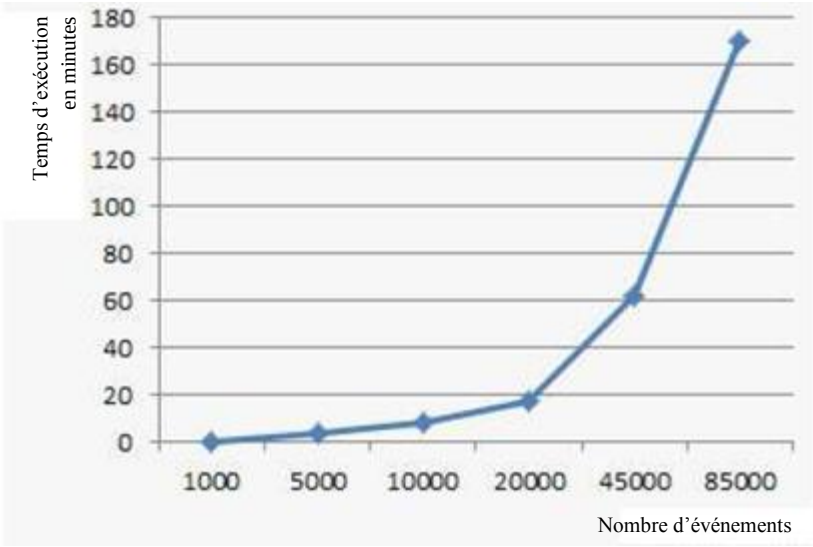
- Puisque la plus longue ligne est composée de *N* éléments et que le pire cas veut que les *n* lignes soient de la même longueur, l'algorithme passera *N* fois sur la BD avant de s'arrêter ce qui veut dire que $Te = nN^2$
- La substitution prendra le même temps, donc $Te = 2nN^2$
- Pour être plus précis, après chaque substitution *N* diminue de 1, donc ce n'est pas *N2*, mais 1+2+…+*N*, ce qui veut dire que $Te = 2nN\frac{N+1}{2}$
- Il faut aussi considérer que l'algorithme s'arrête quand *N* devient inférieur à *F*, donc ce n'est pas 1+2+…+*N* mais *F*+(*F*+1)+(*F*+2)+…+*N* qui est égale à

  $N\frac{N+1}{2} - F\frac{F+1}{2}$

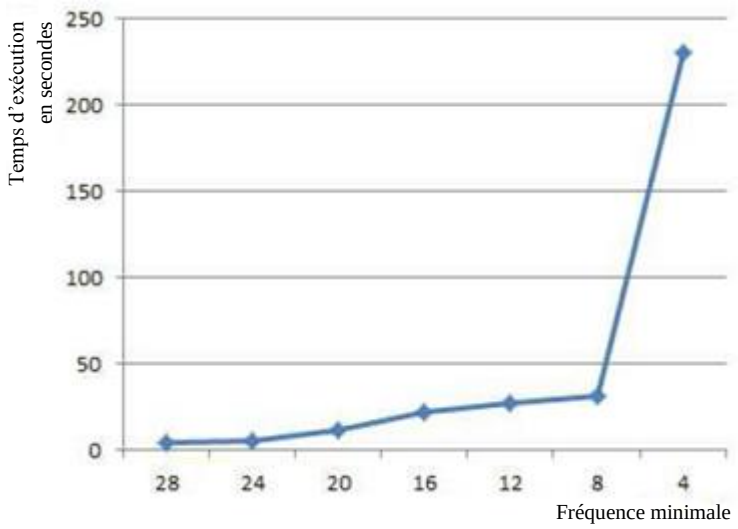
D'après la formule de calcul du temps d'exécution dans le pire cas, nous pouvons constater qu'il est plus influencé par la différence entre la taille de la plus longue ligne et la fréquence minimale. Quand *F* tend vers *N*, *Te* tend vers 0 et quand *F* tend vers 0, *Te* tend vers nN^2.

Dans tous les cas, la base de données sera lue *v* fois, où *v* est le nombre d'éléments de la plus longue séquence fréquente, homogène et fermée. Le temps d'exécution dépendra de la moyenne avec laquelle la base de données diminuera après chaque substitution.

À la lumière de cette étude, nous avons décidé de générer des bases de données de différentes tailles comportant des sous-séquences fréquentes moyennement longues. Ensuite, nous avons observé le temps d'exécution dépendamment de la taille de la base de données et de la fréquence minimale choisie. Le Graphe 5-1 et le Graphe 5-2 montrent les résultats obtenus.



Graphe 5-1. Temps d'exécution dépendamment de la taille de la BD

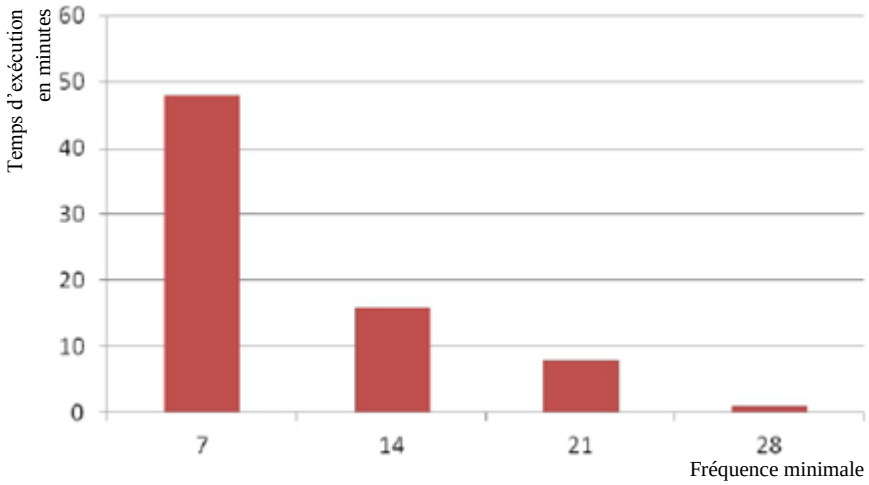


Graphe 5-2. Temps d'exécution dépendamment de la fréquence minimale

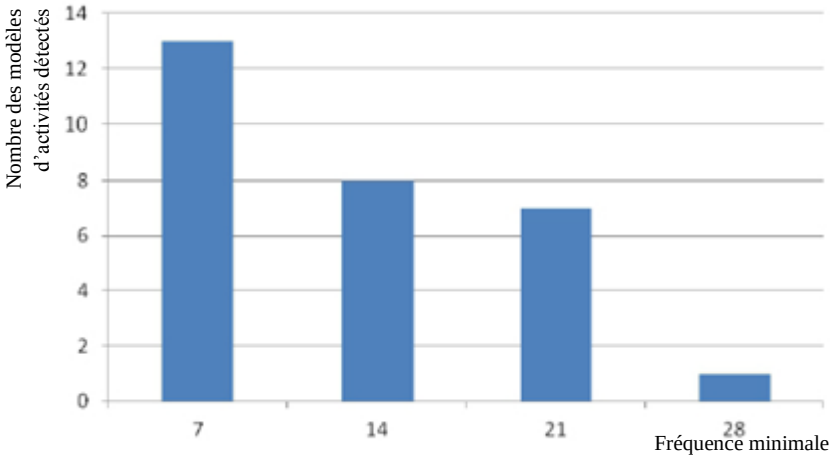
À partir du Graphe 5-1, nous pouvons voir que jusqu'à 20 000 événements, l'augmentation de la taille de la base de données augmente presque linéairement le temps d'exécution. Après ce nombre, le temps d'exécution augmente considérablement pour atteindre trois heures. La même observation peut être faite à partir du Graphe 5-2, le temps d'exécution augmente presque linéairement pendant que la fréquence minimale diminue jusqu'à 8. Après cette fréquence, il augmente considérablement. On peut

constater que ces observations confirment l'étude du pire cas; le temps d'exécution dépend plus de la différence entre la fréquence minimale et le nombre d'éléments de la plus longue séquence fréquente.

Comme nous n'avons aucun contrôle sur le nombre d'éléments de la plus longue séquence fréquente, nous avons testé notre algorithme sur BDSA tout en étudiant l'influence de la fréquence minimale sur le temps d'exécution et le nombre d'activités créées. Le Graphe 5-3 et 5-4 montrent les résultats obtenus.



Graphe 5-3. Temps d'exécution dépendamment de la fréquence minimale



Graphe 5-4. Nombre de modèles d'activités créées dépendamment de la fréquence minimale

D'après les deux Graphes, une fréquence minimale assez grande (28) permet de minimiser le temps d'exécution, mais en même temps elle ne permet pas de créer tous les modèles d'activités. Une fréquence minimale égale à 28 signifie que nous cherchons des activités qui ont été effectuées, de la même façon, au moins une fois par jour. En présentant la manière avec laquelle nous avons mené nos expériences, nous avons bien précisé que certaines activités ne sont pas effectuées tous les jours comme *prendre une douche*. De plus, il nous arrive d'effectuer parfois une activité d'une manière différente, sans oublier que l'étudiant qui menait ces expériences préméditait des erreurs. Donc, il est tout à fait normal qu'il y ait juste une activité qui a été créée pour une fréquence minimale égale au nombre de jours d'observation.

Pour une fréquence minimale assez petite (7), n'importe quelle paire de senseurs a plus de chance de devenir fréquente, ce qui explique le nombre élevé de motifs fréquents qui ont été découverts et considérés comme des modèles d'activités. Le temps d'exécution est très élevé lui aussi à cause du petit nombre de coupures effectué pendant chaque substitution.

Les meilleurs résultats ont été obtenus en choisissant des fréquences minimales moyennes, assez petites pour permettre aux modèles d'activités de devenir fréquents malgré les erreurs et assez grandes pour empêcher des séquences hasardeuses de devenir fréquentes et être considérées comme des modèles d'activités. La valeur 21, dans notre cas, donne le meilleur compromis entre le nombre de modèles d'activités créés et le temps d'exécution. Elle a permis de créer les six modèles d'activités recherchés, plus une séquence hasardeuse qui a été considérée comme modèle d'activité (voir l'Annexe A pour la signification des senseurs) :

1. TP1, TP6, TP1 (*Se réveiller*)
2. MV3, LD5, DB7 (*Utiliser toilette*)
3. TP2, DB6, DB6, MV3, TP2 (*Se laver les mains*)
4. DB4, GBD, DB4 (*Prendre une douche*)
5. Bou, DB2, DB2, CA1, Tas, CA1, CA2, Caf, CA2, CA5, Suc, CA5, CB2, Cui, CB2, Bou, Cui, Tas (*Préparer le café*)
6. MV4, CP1, CP1 (*Quitter la maison*)

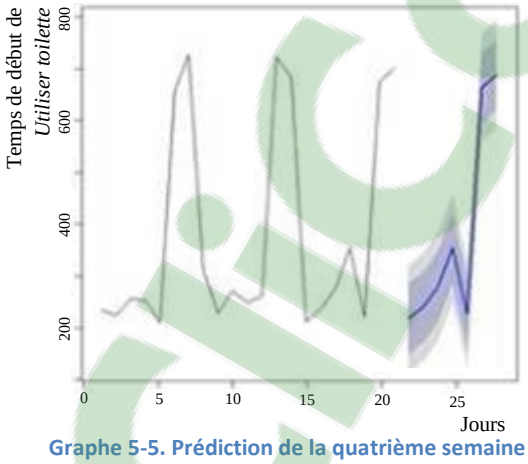
La validation de notre algorithme a montré sa capacité à créer tous les modèles d'activités. Même s'il peut se tromper en considérant des séquences hasardeuses comme des modèles d'activités ce qui augmentera sans doute le temps d'exécution des prochaines étapes de notre approche, il permettra tout de même l'assistance du patient dans ses différentes activités.

5.3.2 Validation de la prédiction des temps de débuts des activités

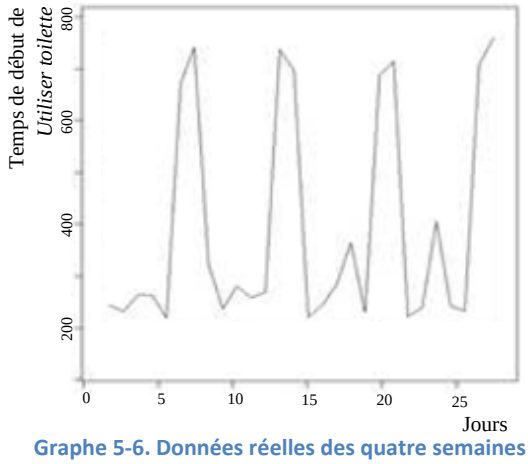
La deuxième étape de notre approche consiste à reconnaître l'activité entamée. Pour réduire le nombre d'hypothèses, nous nous basons sur les temps de débuts habituels des activités pour ne considérer que celles avec un temps de début prédit assez proche du moment de la reconnaissance. La précision de l'algorithme utilisé pour la prédiction joue donc un rôle essentiel dans la réussite de cette étape. Pour valider cet algorithme, nous avons utilisé les trois premières semaines de la base de données BDSA comme données d'apprentissage et nous avons réservé les données de la dernière semaine pour valider les prédictions de la même période trouvées par notre algorithme.

Comme détaillé dans le chapitre précédent, notre algorithme de prédiction des temps de débuts des activités commence par créer une série temporelle pour chaque modèle d'activité. Donc, nous parcourons les 21 premiers jours de la base de données

BDSA et pour chaque modèle d'activité nous enregistrons pour chaque jour le temps d'activation de son premier senseur qui est le temps de début de cette activité. Ensuite pour chaque série temporelle, nous trouvons les trois paramètres d, p et q qui donnent les meilleures prédictions avec le modèle $ARIMA(p, d, q)$ puis nous l'utilisons pour prédire les temps de début du lendemain. Pour valider ces prédictions, les valeurs prédites pour la quatrième semaine et les valeurs réelles de la quatrième semaine ont été comparées pour chaque série temporelle. 83% des temps de début des modèles d'activités ont été bien prédit. Seuls les temps de début de *Quitter maison* n'ont pas pu être prédis ce qui indique que la notion d'autocorrélation n'existe pas dans cette série temporelle. Pour avoir une idée sur les résultats des prédictions, les valeurs des trois premières semaines et les valeurs prédites pour la quatrième semaine de l'activité *utiliser toilette* sont affichés dans le Graphe a Figure 5-10, tandis que la Figure 5-11 montre l'affichage des données réelles des quatre semaines de la même activité.



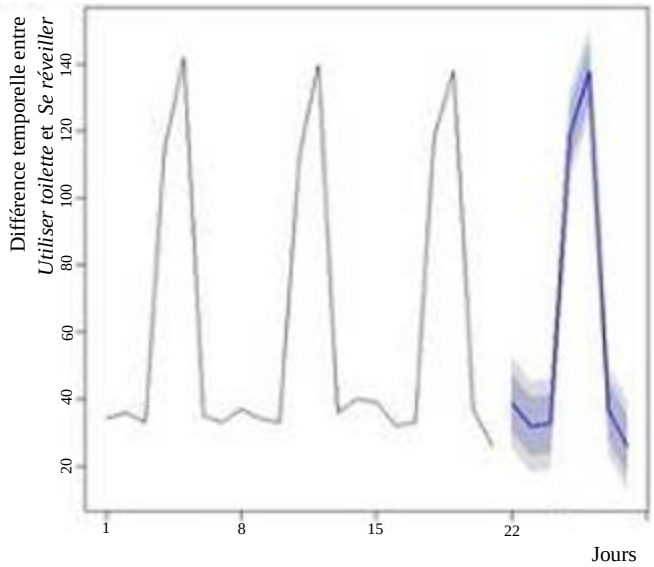
Graphe 5-5. Prédiction de la quatrième semaine



Graphe 5-6. Données réelles des quatre semaines

Le modèle qui a été choisi pour cette série temporelle est $ARIMA(1,0,0)$, ce qui veut dire que la série temporelle est stationnaire $(d = 0)$ et la valeur minimale du AICC est 358.67 a été trouvée par les paramètres $p = 1$ et $q = 0$. La comparaison des graphiques de ces

deux Figures montre que les valeurs prédites sont assez proches des valeurs réelles sauf les valeurs du 24ème jour; la valeur prédite est égale à 3101.941 alors que la valeur réelle est égale à 4559. Avant d'analyser les causes possibles de cette différence, il faut s'assurer que c'est cette même valeur qui serait prédite pour cette journée. Notre algorithme, comme expliqué dans le chapitre précédent, quand il détecte une différence entre le temps prédit pour une activité et le temps réel où elle s'est déroulée, il recalcule d'une autre façon les temps prédits pour toutes les activités qui se déroulent après elle. L'analyse des prédictions de l'activité *Se réveiller*, qui précède *Utiliser toilette*, montre que la seule journée qui a été mal prédite est la même, 24ème journée. Le temps de début de l'activité *Utiliser toilette* serait donc recalculé en créant une nouvelle série temporelle composée de la différence entre les deux activités pour chaque journée. Les prédictions de la quatrième semaine de cette nouvelle série temporelle sont présentées au Graphe 5-7.

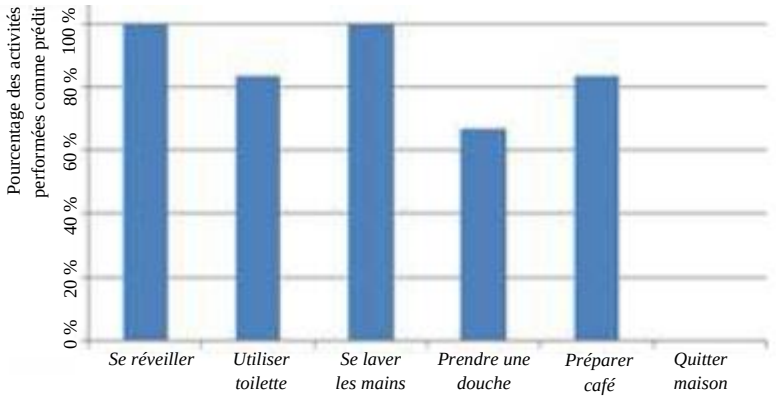


Graphe 5-7. Prédiction des différences temporelles entre Se réveiller et Utiliser toilette

Sachant que le temps réel où l'activité *Se réveiller* s'est déroulée à la 24ème journée est 4546 et que la différence prédite entre les deux activités pour la même journée est 12.714, le nouveau temps prédit pour *Utiliser toilette* est :

$$TDPa = 4546 + 12.714 = 4558.71 \approx 4559$$

Pour évaluer les résultats des autres activités, Le Graphe 5-8 affiche le pourcentage des prédictions réussies pour chaque activité.



Graphe 5-8. Pourcentage des prédictions par activité

À partir de ce Graphe, nous pouvons voir que les temps de début prédits pour *Se réveiller* et *Se laver les mains* correspondaient parfaitement aux temps réels. Le résultat de *Se réveiller* est un peu attendu parce que l'étudiant utilisait un réveil et quittait son lit quelques minutes avant ou après. Pour *Se laver les mains*, le fait qu'elle suivait immédiatement *Utiliser toilette* a contribué à sa bonne prédiction puisqu'elle est souvent corrigée par la deuxième partie de notre algorithme. Les résultats de *Prendre une douche* n'étaient pas optimaux et c'est tout à fait justifiable parce qu'elle n'a pas été effectuée régulièrement. Le pire résultat fut pour *Quitter maison* qui n'a pas pu être prédite.

La validation de cet algorithme a montré qu'il est plus efficace pour les activités menées assez régulièrement dont les futurs temps de début dépendent des anciennes valeurs ou des valeurs des activités qui la précèdent. Donc, il sera très efficace pour la majorité des AVQs visées par notre approche comme : *Prendre médicament*, *Se nourrir*, etc.

5.3.3 Validation de la prédiction des temps d'activation des senseurs

La première étape de notre approche consiste à créer les modèles d'activités qui sont sous la forme : S_1, $S_2[t_{min}\text{-}t_{max}]$, $S_3[t'_{min}\text{-}t'_{max}]$ … . L'avantage de créer des modèles de la sorte est que le prochain senseur est bien connu et même le temps de son activation est estimé. Par exemple, une fois le senseur S_1 est activé, nous nous attendons à ce que le senseur S_2 soit activé après t_{min} secondes ou au plus tard après t_{max} secondes.

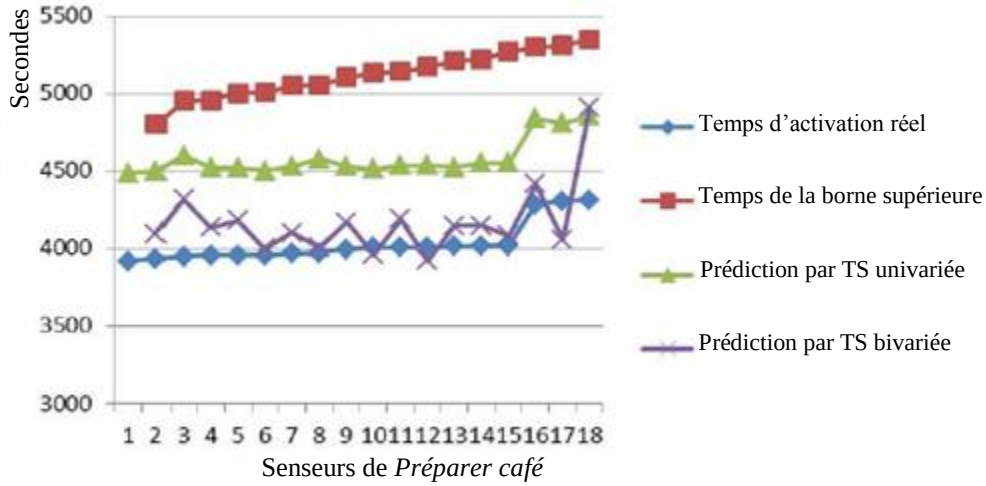
La deuxième étape de notre approche permet de reconnaître l'activité entamée. En d'autres termes, elle permet de spécifier quel modèle d'activité est en cours d'exécution parmi tous les modèles d'activités créés dans la première étape. Donc, dès que la personne assistée commence une activité, que nous arrivons à reconnaître, nous connaissons les prochaines étapes et les temps approximatifs de leurs déroulements. La question qui se pose maintenant est : comment détecter les erreurs de la personne assistée et quand est-ce que nous devons déclencher les effecteurs pour l'aider en lui indiquant la prochaine étape?

Dans le chapitre précédent, nous avons répondu à cette question de trois manières différentes :

- Signaler une erreur et envoyer le message d'aide quand le prochain senseur n'est pas activé avant la borne supérieure de son intervalle temporel du modèle d'activité reconnu (t_{max}).
- Créer une série temporelle univariée composée des temps d'activation du senseur par apport au début de l'activité par jour. Ensuite, prédire son temps d'exécution pour le jour de l'assistance et l'utiliser à la place de la borne supérieure.

- Au lieu d'une série univariée, utiliser une série bivariée composée des temps d'activations du senseur en question et du senseur qui le précède.

Pour comparer les trois méthodes, nous avons utilisé les trois premières semaines pour créer les séries temporelles et nous avons réservé les données de la quatrième semaine pour les tests. Les résultats pour toutes les activités, à l'exception de *Quitter maison*, étaient semblables à ceux de l'activité *Préparer café* présentés au Graphe 5-9.



Graphe 5-9. Les temps réels et prédits des senseurs

Le Graphe 5-9 montre la grande différence entre les temps réels d'activation des senseurs et les bornes supérieures des intervalles temporels des modèles d'activités. Notre recherche a été motivée par cette grande différence qui peut représenter la période de temps où le patient de la maladie d'Alzheimer a du mal à se souvenir de sa prochaine action. Afin de montrer l'énorme impact de cette grande différence entre les temps réels d'activation des senseurs et les bornes supérieures, supposons que la personne assistée avait besoin d'aide pour toutes les actions de *Préparer café*, cette activité sera effectuée en environ 5 heures au lieu de 6,5 minutes. En utilisant les séries univariées, les résultats des prédictions se sont nettement améliorés et la différence entre les temps réels d'activation et ceux prédits a diminué par la moitié. Mais les prédictions les plus précises

ont été obtenues en utilisant les séries temporelles bivariées, ce qui prouve que les valeurs précédentes d'un senseur S_1 aident dans la prédiction des futures valeurs du senseur qui le suit S_2. Dans le domaine des séries temporelles multivariées, on dit que S_1 Granger-causes S_2.

Pour terminer la comparaison entre les trois méthodes, il faut signaler que pour l'exemple précédent, *Préparer café* ne prendrait que 48 minutes en utilisant des séries temporelles bivariées. Certes, il y a une grande amélioration, mais il reste encore du travail à faire pour se rapprocher du temps réel de l'activité qui est de 6,5 minutes.

La prédiction des temps d'activations des senseurs est une tâche très importante pour la réussite de l'assistance technologique. Malheureusement, peu de recherches ont ciblé cette problématique. Il est vrai que notre méthode utilisant les séries temporelles multivariées améliore grandement les résultats obtenus et semble une solution efficace quand la personne assistée ne commet pas beaucoup d'erreurs, mais il est évident qu'il faut encore l'améliorer si nous voulons l'utiliser pour des patients de la maladie d'Alzheimer dans des états plus avancés.

5.4 Conclusion

Nous avons commencé ce chapitre par la description des différents choix matériels et logiciels qu'il faut faire lors de la conception d'une maison intelligente. Nous avons vu comment le choix des senseurs est dicté par le type d'assistance visé; par exemple une assistance non intrusive qui préserve l'intimité de la personne assistée évite les caméras et les senseurs portables. Ensuite, nous avons utilisé cette première partie pour décrire et

justifier les différents choix technologiques installés au LIARA, notre maison intelligente utilisée pour les tests des différentes approches des étudiants et des chercheurs.

Après avoir donné une idée détaillée sur le LIARA, nous avons décrit comment les expériences s'y sont déroulées en spécifiant que c'est un étudiant volontaire qui a simulé la procédure du réveil pendant 28 jours.

La dernière partie de ce chapitre a été consacrée à la validation séparée de trois importantes étapes de notre approche. Nous avons constaté comment notre premier algorithme était capable de créer tous les modèles d'activités même s'il peut considérer des séquences hasardeuses comme des modèles d'activités. Les résultats de l'algorithme de la deuxième étape ont été plus que satisfaisants en arrivant à bien prédire les temps de début de 83% des activités et il a surtout montré son efficacité à prédire les AVQs visées par notre approche. En fin, les séries temporelles multivariées utilisées dans la troisième étape ont donné les meilleurs résultats dans la prédiction des temps d'activation des senseurs et ont prouvé qu'elles peuvent être utilisées quand la personne assistée ne commet pas trop d'erreurs.

Chapitre 6

6 Conclusion générale

L'assistance des personnes âgées et ceux atteints de dysfonctionnements cognitifs comme les patients de la maladie d'Alzheimer présente un grand défi aux sociétés occidentales. L'assistance technologique est la solution envisageable pour remédier aux différents inconvénients de l'assistance traditionnelle. En effet, des agents ambiants peuvent être développés pour épauler les personnes aidantes dans leur travail, ce qui allégera les ressources humaines et financières des systèmes de santé de ces pays. L'émergence de plusieurs domaines, comme l'informatique ubiquitaire et l'intelligence artificielle, a rendu la conception d'une telle assistance possible. Des senseurs de petite taille capables d'effectuer des mesures très précises ont vu le jour et sont intégrés, d'une façon transparente, à la vie de tous les jours de l'utilisateur pour améliorer son quotidien. En même temps, des techniques capables de traiter d'énormes quantités de données, comme celles générées par les senseurs, ont été développées dont les techniques de la fouille de données. Plusieurs approches ont déjà été proposées pour répondre aux nombreux problèmes reliés aux différentes étapes de l'assistance technologique, surtout l'étape de la reconnaissance d'activités qui est sans doute l'étape la plus complexe de ce processus.

À la lumière de ces approches, nous avons proposé, au cours de cette thèse, une approche complète qui présente des solutions aux différentes étapes de l'assistance technologique. Notre approche a été inspirée de quelques lacunes observées dans différents travaux, surtout le travail réalisé au cours de mon mémoire de maitrise. Elle exploite essentiellement les données temporelles souvent ignorées par les autres approches malgré leur importance dans la reconnaissance d'activités et la détection d'erreurs. Elle utilise différentes techniques de la fouille de données temporelles ainsi que plusieurs modèles de prédiction des séries temporelles. Dans la suite de ce dernier chapitre, une révision des objectifs fixés au départ sera conduite, suivie d'une évaluation générale de notre approche montrant ses atouts, ses limitations connues ainsi que les prochaines améliorations prévues.

6.1 Réalisation des objectifs fixés

Pour arriver à concevoir une approche complète permettant l'assistance technologique des personnes atteintes de la maladie d'Alzheimer, qui est l'objectif ultime de ce travail, nous nous sommes imposé des objectifs intermédiaires cités à l'introduction.

Le premier objectif visé consiste en une étude approfondie de la maladie d'Alzheimer pour nous permettre de proposer des solutions aux différentes erreurs que ses patients sont susceptibles de commettre. La lecture et l'analyse de plusieurs livres et publications sur cette maladie [10], [12], [13] nous ont permis d'adapter notre approche à certaines erreurs non considérées par la majorité des approches existantes, comme les erreurs d'initiation. En fait, ces approches se basent seulement sur les récentes actions

détectées pour reconnaître l'activité entamée. Le problème avec les erreurs d'initiation c'est que l'agent-acteur n'est plus capable d'amorcer son activité. Donc, aucune action ne sera effectuée ni détectée, ce qui empêche ces approches de reconnaître l'activité et de l'assister. Une méthode de prédiction des temps de début des activités a été intégrée à notre approche dont un de ses objectifs est la réponse à ce genre d'erreurs.

Le grand volume des données générées par les senseurs rend l'utilisation des techniques de la fouille de données quasiment indispensable. Notre deuxième objectif consistait donc à maitriser ces techniques afin de choisir parmi elles celles qui répondront le mieux aux différents problèmes rencontrés lors de la conception de notre approche. L'étude exhaustive des techniques de la fouille de données nous a permis de choisir l'algorithme C-means pour l'intégrer dans notre nouvel algorithme de création des modèles d'activités. Elle nous a permis aussi de comprendre les approches existantes de reconnaissance d'activités qui utilisent ces techniques.

Notre troisième objectif consistait à effectuer un état de l'art sur les différents types d'approches de reconnaissance d'activités déjà existantes. La lecture et la compréhension d'un grand nombre de publications [7], [8], [14], [29] destinées à différentes catégories de reconnaissance d'activités comme les approches basées sur la vision, sur les senseurs portables, sur les objets, les approches supervisées et non supervisées et les approches en ligne et hors ligne, nous a permis de bien situer notre travail et de le perfectionner en profitant des avantages et en évitant les inconvénients des autres approches étudiées. Elle nous a permis aussi d'éviter de développer une approche déjà existante sans apporter une réelle contribution.

Le quatrième objectif visait la création d'une nouvelle approche pour l'assistance technologique conçue à la lumière des étapes précédentes. Notre approche, expliquée au chapitre 4, est une approche à l'insu, basée sur les objets, hors-ligne et non supervisée. Elle travaille directement sur les données générées par les senseurs. Le grand volume de ces données représentait le premier problème que nous avons rencontré. La solution que nous avons proposée consiste à commencer notre approche par une étape de réduction de données en ne considérant que les senseurs activés ainsi que leurs temps d'activation au lieu de toutes les mesures de tous les senseurs. Ensuite, nous avons divisé le processus de reconnaissance d'activités en trois étapes; la création des modèles d'activités, la réduction du nombre d'hypothèses et la recherche de l'activité entamée. Pour la création des modèles d'activités, nous avons développé un nouvel algorithme capable d'extraire les sous-séquences fréquentes et homogènes, qui représentent les modèles d'activités, à partir des données transformées. Il faut noter que les modèles d'activités créés par cet algorithme comportent l'estimation du temps habituel entre deux senseurs adjacents représentée sous forme d'intervalles temporels. Pour la réduction du nombre d'hypothèses, nous avons utilisé le modèle ARIMA des séries temporelles pour prédire le temps de début de chaque modèle d'activité en fonction du jour de la semaine. Ces prédictions nous ont permis de définir l'activité la plus probable (utilisée pour répondre aux problèmes d'initiation et celui de l'équiprobabilité) et de rechercher l'activité entamée parmi un sous ensemble limité de modèles d'activités sélectionnés selon leurs distances du temps de reconnaissance (réponds au problème du nombre d'hypothèses). Pour la recherche de l'activité entamée, nous avons programmé un réseau bayésien, composé du sous-ensemble de l'étape précédente, dont les probabilités initiales sont

déterminées selon les distances entre les temps de début prédits et le temps de reconnaissance. Ces probabilités sont par la suite mise à jour selon l'appartenance des récents senseurs activés aux modèles d'activités. La dernière étape de notre approche consiste à détecter les erreurs éventuelles. La solution proposée dans cette approche consiste à utiliser le modèle VAR des séries temporelles pour prédire les temps d'activations des senseurs qui composent l'activité reconnue et de déclencher une erreur quand le temps courant dépasse le temps d'activation d'un senseur sans qu'il soit activé.

Notre dernier objectif visé consiste à valider notre approche dans un contexte réel d'assistance. Nous avons donc effectué nos expériences au LIARA où une centaine de senseurs y sont installés. Avec l'impossibilité d'observer une personne atteinte de la maladie d'Alzheimer pour une assez longue période au sein du LIARA, nous avons simulé, pendant 28 jours, la procédure du réveil qui comporte six activités : *Se réveiller*, *Utiliser toilette*, *Se laver les mains*, *Prendre une douche*, *Préparer le café* et *Quitter la maison*. Les résultats étaient satisfaisants et nous ont permis de déterminer les points forts et les faiblesses de notre approche.

Il faut signaler que les différentes étapes de notre approche ont fait l'objet de plusieurs publications [19], [35], [38] et que l'approche complète a été publiée dans le fameux journal Springer de l'intelligence ambiante et l'informatique humanisée "Ambient Intelligence and Humanized Computing" (AIHC) [9].

6.2 Les apports de notre approche

Notre approche proposée dans cette thèse comporte plusieurs apports intéressants, non pas seulement dans le domaine de la reconnaissance d'activités, mais aussi dans le

domaine de l'extraction des motifs fréquents. En effet, le nouvel algorithme que nous avons développé pour la création des modèles d'activités appartient à ce dernier domaine et peut être utilisé pour découvrir des sous-séquences fréquentes et fermées où l'ordre des éléments est parfaitement respecté. De plus, nous avons volontairement divisé cet algorithme en deux parties. La première partie permet de transformer une base de données temporelle en une base de données non temporelle avec la possibilité d'extraire les informations temporelles pertinentes par la suite. Cette première partie permet donc aux différents algorithmes qui ne considèrent pas les données temporelles de travailler sur des bases de données temporelles.

En ce qui concerne le domaine de la reconnaissance d'activités, cet algorithme a la spécificité de travailler sur une base de données de dimensionnalité très réduite comparée à la base de données originale utilisée par les autres approches comme celle d'Ordonez et al. [8]. Effectivement, en ne considérant que les senseurs activés et leurs temps d'activation, les données sont considérablement réduites sans perdre des informations pertinentes. Un autre apport très intéressant de cet algorithme est qu'il est le seul, à notre connaissance, qui crée des modèles d'activités comportant une estimation du temps habituel entre deux senseurs adjacents, ce qui aide grandement lors de l'étape de détection d'erreurs.

La réduction du nombre d'hypothèses est un problème majeur dans la reconnaissance d'activité, mais peu de recherches ont proposé une réelle solution à ce problème. Dans notre approche, ce problème est considéré comme la problématique de la recherche et nous lui avons proposé une solution basée sur des contraintes temporelles. En effet, nous avons utilisé les séries temporelles pour prédire les temps de début des

modèles d'activités dans le temps. Ces temps prédits sont ensuite utilisés pour discriminer les modèles peu probables d'être effectués au moment de la reconnaissance.

La prédiction des temps de début des modèles d'activités a été utilisée, dans notre approche, pour répondre à un autre problème plus particulier causé par les erreurs d'initiations qui empêchent l'agent acteur d'amorcer son activité et d'activer les senseurs adéquats. Sans l'activation des senseurs, l'activité ne peut pas être reconnue. Dans ce cas, seuls les temps prédits sont utilisés pour reconnaître l'activité qui doit être entamée.

Le dernier apport de notre approche est la prédiction des temps d'activations des senseurs qui composent l'activité reconnue. Cette prédiction améliore nettement la détection d'erreurs et permet à l'agent ambiant d'envoyer les messages d'aides au moment opportun.

6.3 Les limitations connues de notre approche

Bien que l'approche proposée a répondu à tous les objectifs fixés au départ et les résultats de chacune de ses étapes ont été satisfaisants, nous sommes conscients qu'elle souffre de quelques limitations et que des améliorations s'imposent pour qu'elle soit efficace dans le cadre d'une assistance réelle. La première limitation vient du fait que nous avons visé dès le départ de n'exploiter que les données temporelles. Certes, elles nous ont permis de capturer quelques habitudes importantes qui reflètent le comportement normal de la personne observée, comme les temps de début et le temps entre chaque deux actions successives de ses différentes activités, mais l'utilisation d'autres contraintes est indispensable pour capturer d'autres aspects de son comportement normal. Par exemple, les contraintes spatiales peuvent indiquer les lieux où les activités

se déroulent d'habitude. De telles informations peuvent être utilisées, par exemple, dans la réduction du nombre d'hypothèses en considérant juste les activités qui se déroulent dans l'emplacement de l'agent-acteur.

Une deuxième limitation de notre approche vient du fait que c'est une approche hors-ligne. Il est vrai que nous avons prévu d'exécuter notre algorithme de création des modèles d'activités chaque mois pour prendre en considération les changements d'habitudes de l'agent-acteur, mais pendant ce mois, tout changement d'habitudes sera considéré comme une erreur où augmentera le temps d'exécution de notre approche. Par exemple, si la personne observée décide de ne plus prendre sa douche le matin, cette activité sera toujours considérée dans le réseau bayésien de recherche de l'activité entamée qui se crée le matin et l'agent-acteur recevra des messages l'incitant à effectuer cette activité tous les matins jusqu'à la prochaine création des nouveaux modèles d'activités.

La dernière limitation connue de notre approche se manifeste quand les techniques des séries temporelles sont incapables de prédire les temps de début de l'activité entamée. Dans ce cas, le système de réduction du nombre d'hypothèses ne fait que ralentir la reconnaissance d'activités parce que le réseau bayésien créé ne contiendra jamais l'activité recherchée. Il faut noter qu'il existe des activités qui doivent être incluses dans tous les réseaux bayésiens parce qu'elles peuvent être effectuées à n'importe quel moment de la journée, comme *Boire* par exemple.

6.4 Les développements futurs

Avant de détailler les prochains développements envisagés pour notre approche, nous pensons qu'il serait intéressant de la tester dans le cadre d'une assistance réelle des personnes atteintes de la maladie d'Alzheimer. Une telle expérience nous aidera à constater l'ampleur des limitations citées à la section précédente et nous fera, sans doute, découvrir d'autres limitations auxquelles nous n'avons pas pensé. Elle nous permettra aussi de mettre le doigt sur les modifications à apporter à notre approche et nous guidera à définir les développements nécessaires.

Pour l'instant, un des développements qui nous semble essentiel concerne la détection des erreurs après la reconnaissance de l'activité entamée. Il est vrai que notre style de vie et nos habitudes font en sorte qu'on se crée des routines pour effectuer nos activités, surtout pour les AVQs. Cela n'empêche pas que, de temps en temps, on puisse effectuer une activité de manières différentes. Par exemple, pour l'activité *Se réveiller*, la plupart du temps, on va arrêter le réveil puis quitter le lit, mais ça peut arriver qu'on quitte le lit d'abord et après on arrête le réveil. Cette remarque ne pose pas de problème pour la création des modèles d'activités parce que la façon routinière d'effectuer l'activité la rendra toujours fréquente (ça dépend évidemment de la valeur de la fréquence minimale choisie). Par contre, des erreurs seront signalées chaque fois que l'agent effectue l'activité d'une manière différente. Pour remédier à ce problème, un système de détection d'erreurs plus sophistiqué doit être développé. Ce système doit effectuer des tests plus avancés avant d'inférer qu'une erreur s'est produite. Un de ces tests auquel on peut penser maintenant stipule que : si le dernier senseur activé est différent du prochain

senseur de l'activité reconnue, mais il lui appartient quand même, aucune erreur ne sera signalée.

Le nouveau système de détection d'erreurs pourra même proposer une solution au problème des activités entrecroisées avec quelques modifications au système de reconnaissance d'activités. Effectivement, avec les deux systèmes actuels, quand l'agent-acteur essaie d'effectuer plus qu'une activité en même temps, toute activation d'un senseur différent du prochain senseur dans l'activité reconnue sera signalée comme erreur. Le système de reconnaissance d'activités doit donc permettre la reconnaissance de plus qu'une activité et le système de détection d'erreurs doit effectuer des tests plus complexes pour prédire le prochain senseur qui sera activé parmi les senseurs de toutes les activités reconnues.

6.5 Bilan personnel sur cette recherche

Comme conclusion finale, j'aimerai bien parler de cette expérience intéressante de mener un projet de recherche. Certes, mon investissement dans ce travail fastidieux m'a coûté 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 avec le domaine de la reconnaissance d'activités au sein d'un habitat intelligent ainsi qu'avec le domaine émergeant de la fouille de données. Elle m'a permis également d'assister à la naissance et au développement du LIARA et faire partie de sa formidable équipe. Tout au long de cette recherche, j'ai pu développer mes habiletés communicationnelles et contribuer modestement dans mon domaine de recherche en présentant mes publications à différentes conférences [19], [33], [35], [37], [38], [86] et au prestigieux journal

Springer AIHC [9]. Après cette introduction positive à la recherche, j'ai hâte de commencer une carrière en tant que chercheur et participer, comme je peux, aux avancements scientifiques. À la fin de cette thèse, je tiens à remercier toutes les personnes qui m'ont soutenu, d'une façon ou d'une autre, dans ma quête d'acquisition du savoir, des compétences et de l'expérience.

Annexe A

Les senseurs du LIARA

TP*	Tapis de pression
MV*	Détecteur de mouvement
LD*	Capteur de lumière
DB*	Écoulement d'eau
CA*	Capteur électromagnétique posé sur une armoire
CB*	Capteur électromagnétique posé sur un tiroir
CP*	Capteur électromagnétique posé sur une porte
***	Tag RFID posé sur un objet

Annexe B

Les séries temporelles

Les données obtenues d'observations collectées séquentiellement dans le temps sont extrêmement courantes. Nous observons les précipitations annuellement, la température quotidiennement, les indices de prix mensuellement, les activités cardiaques chaque milliseconde, l'abondance d'une espèce animale annuellement, etc. Presque dans tous les domaines des séries temporelles sont enregistrées. Analyser ce type de données temporelles nous aide à comprendre le passé et d'expliquer certaines variations observées, de prédire le futur et d'étudier les liens avec d'autres séries.

Définition

Une série temporelle (ou chronologique) est la suite d'observations d'une famille de variables aléatoires réelles notées $(X_t) \in \theta$ où l'ensemble θ est appelé espace de temps qui peut être discret; $\theta \subset Z$ ou continu; $\theta \subset R$ [18].

Composantes d'une série temporelle

Généralement, une série temporelle (X_t) est la résultante des trois composantes fondamentales suivantes :

- La tendance (Z_t) représente l'évolution à long terme de la série étudiée. Elle traduit le comportement "moyen" de la série;

- La saisonnalité ou composante saisonnière (S_t) qui correspond à un phénomène qui se répète à intervalles de temps réguliers (périodiques);
- Le bruit ou la composante résiduelle (ε_t) qui correspond à des fluctuations irrégulières, en général de faible intensité, mais de nature aléatoire.

D'autres composantes peuvent être remarquées dans une série temporelle, mais elles sont moins importantes à étudier et plus difficiles à prédire comme les phénomènes accidentels ou les phénomènes cycliques qui sont des phénomènes qui se répètent, mais contrairement à la saisonnalité, ils se répètent sur des durées qui ne sont pas fixes et généralement plus longues.

Modélisation d'une TS

Un modèle est une image simplifiée de la réalité qui vise à traduire les mécanismes de fonctionnement du phénomène étudié et permet de mieux les comprendre, en supposant que la série est régie par une certaine fonction. On distingue principalement deux types de modèles : les modèles déterministes et les modèles stochastiques.

- ***Les modèles déterministes***

Les modèles déterministes supposent que l'observation de la série à la date t est une fonction du temps t et d'une variable ε_t centrée faisant office d'erreur au modèle et représentant la différence entre la réalité et le modèle proposé : $X_t = F(t, \varepsilon_t)$. De plus, ε_t est supposée être dé-corrélée (on ne peut pas exprimer une de ses valeurs en fonction des autres). Il existe deux types de modèles déterministes:

- Le modèle additif : la variable X_t s'écrit comme la somme de trois termes :

$$X_t = Z_t + S_t + \varepsilon_t$$

où Z_t représente la tendance (déterministe), S_t la saisonnalité (déterministe aussi) et ε_t les composantes aléatoires dé-corrélées.

- Le modèle multiplicatif : la variable X_t s'écrit comme le produit de la tendance et d'une composante de saisonnalité :

$$X_t = Z_t(1 + S_t)(1 + \varepsilon_t)$$

- ***Les modèles stochastiques***

Les modèles stochastiques sont du même type que les modèles déterministes sauf que les variables de bruit ε_t possèdent une structure de corrélation non nulle. ε_t est une fonction des valeurs passées (plus ou moins lointaines suivant le modèle) et d'un terme d'erreur η_t:

$$\varepsilon_t = G(\varepsilon_t - 1, \varepsilon_t - 2, \dots, \eta_t)$$

où η_t est un **bruit blanc**. C'est-à-dire une variable aléatoire de moyenne nulle non corrélée.

Il existe plusieurs méthodes pour décider du type du modèle à utiliser. La méthode analytique consiste à calculer les moyennes et les écarts-types de chacune des périodes considérées, puis de trouver la droite des moindres carrés $\sigma = a\bar{X} + b$. Si a est nul, c'est le modèle additif qui doit être choisi, sinon c'est le modèle multiplicatif.

Pour expliquer les termes statistiques que nous venons d'utiliser et bien d'autres nécessaires à la compréhension de la suite de l'annexe, nous présentons les définitions suivantes :

Le point moyen : $\bar{X} = \frac{X_1 + X_2 + \cdots + X_n}{n} = \frac{1}{n} \sum_{k=1}^{n} X_k$

La variance : $V(X) = \frac{1}{n} \sum_{k=1}^{n} (X_k - \bar{X})^2$

L'écart type : $\sigma(X) = \sqrt{V(X)}$

La covariance : $cov(X, Y) = \frac{1}{n} \sum_{k=1}^{n} (X_k - \bar{X})(Y_k - \bar{Y})$

L'autocovariance : $Y_X(t + h, t) = cov(X_{t+h}, X_t)$

La droite de régression $Y = aX + b$ passe par le point moyen $G(\bar{X}, \bar{Y})$, avec $a = \frac{cov(X,Y)}{V(X)}$

Stationnarité

La stationnarité est la propriété qui indique si la série temporelle évolue dans le temps ou non. Si la TS n'évolue pas dans le temps, c'est-à-dire que les observations successives de la série sont identiquement distribuées, mais pas nécessairement indépendantes, elle est stationnaire. En d'autres termes, une TS est stationnaire si sa moyenne, sa variance et son autocorrélation ne changent pas au cours du temps.

Différentiation d'une TS

On appelle différence d'ordre 1 de la série X_t les quantités : $\Delta X_t = X_t - X_{t-1}$ qui peut s'écrire aussi : $\Delta X_t = (1 - B)X_t$ où B est l'opérateur de retard (noté aussi L pour lag). En générale, une différence d'ordre d s'écrie : $\Delta^d = (1 - B)^d$

Modèle Autorégressif

Un modèle autorégressif, $AR(p)$ est une fonction linéaire où la série Y_t (au temps t) s'exprime par la combinaison "linéaire" de ses p précédentes valeurs. Il répond à l'équation :

$$Y_t = \alpha + \varphi_1 Y_{t-1} + \varphi_2 Y_{t-2} + \cdots + \varphi_p Y_{t-p} + \varepsilon_t$$

où, α est une constante, φ_1, φ_2, ….. , φ_p sont les coefficients autorégressifs et ε_t est l'erreur aléatoire observée au temps t.

Modèle de Moyenne Mobile

Un modèle de moyenne mobile $MA(q)$ est une fonction linéaire où les valeurs de la série Y_t s'expriment par la combinaison "linéaire" des q erreurs aléatoires précédentes. Un modèle $MA(q)$ répond à l'équation :

$$Y_t = \alpha - \theta_1 \varepsilon_{t-1} - \theta_2 \varepsilon_{t-2} - \cdots - \theta_q \varepsilon_{t-q} + \varepsilon_t$$

Où, α est une constante, θ_1, θ_2, …, θ_q sont les paramètres de moyenne mobile, ε_{t-1}, ε_{t-2},…, ε_{t-q} sont les q valeurs d'erreurs précédentes et ε_t est un terme d'erreur aléatoire qui entache l'essai au temps t.

ARIMA

Les modèles ARIMA, acronyme de AutoRegressive Integrated Moving-Average qui sont aussi appelés modèles de Box et Jenkins, sont essentiellement utilisés à des fins de prévision des valeurs futures, de détermination des valeurs manquantes dans une série ou d'identification de la structure de la série temporelle. Un modèle ARIMA s'exprime

en fonction de la notation (p, d, q) où p est l'ordre de termes autorégressif, d représente l'ordre de différenciation et q est l'ordre de la moyenne mobile.

L'analyse des séries temporelles par les modèles ARIMA repose sur trois étapes: l'identification, la détermination et la vérification. La première étape de l'analyse consiste à identifier le modèle ARIMA qui pourrait générer la série temporelle étudiée en estimant les paramètres d, p et q. La seconde étape d'estimation consiste à déterminer les coefficients des termes de l'équation. Enfin, la troisième étape, celle de la vérification consiste à contrôler le degré d'ajustement du modèle ARIMA à la série étudiée.

L'étape d'identification des modèles ARIMA repose sur deux outils : les fonctions d'autocorrélations (ACF) et les fonctions d'autocorrélations partielles (PACF). L'étude des ACF et PACF permet de mettre en évidence l'existence d'une relation interessais. Précisément, c'est la présence d'autocorrélations et d'autocorrélations partielles significatives qui apparaissent sous la forme de pics dans les graphes des ACF et PACF qui la révèle.

- Une ACF met en évidence le degré de corrélation de la série avec elle-même en fonction de l'accroissement du décalage (lag h) à pas de 1. Son étude est essentielle à la détermination du nombre de termes de différenciation I(d) et de moyenne mobile MA(q);
- La PACF met en évidence le degré de corrélation entre deux valeurs "espacées" d'un décalage (lag h) alors que les valeurs intermédiaires sont contrôlées. La PACF est essentielle pour déterminer l'ordre du terme autorégressif AR(p).

Bibliographie

[1] "Rising Tide: The Impact of Dementia on Canadian Society." [Online]. Available: http://www.alzheimer.ca/~/media/Files/national/Advocacy/ASC_Rising_Tide_Full_Report_e.pdf.

[2] A. Fink and G. Doblhammer, "Risk of Long-Term Care Dependence for Dementia Patients is Associated with Type of Physician: An Analysis of German Health Claims Data for the Years 2006 to 2010," *Journal of Alzheimer's Disease*, vol. Preprint, no. Preprint. IOS Press, pp. 1–10.

[3] Y. Hong-Qi, S. Zhi-Kun, and C. Sheng-Di, "Current advances in the treatment of Alzheimer's disease: focused on considerations targeting Aβ and tau.," *Transl. Neurodegener.*, vol. 1, no. 1, p. 21, Jan. 2012.

[4] C. Ramos, J. C. Augusto, and D. Shapiro, "Ambient Intelligence—the Next Step for Artificial Intelligence," *IEEE Intell. Syst.*, vol. 23, no. 2, pp. 15–18, 2008.

[5] V. Ricquebourg, D. Menga, D. Durand, B. Marhic, L. Delahoche, and C. Loge, "The Smart Home Concept : our immediate future," in *2006 1ST IEEE International Conference on E-Learning in Industrial Electronics*, 2006, pp. 23–28.

[6] M. Van Tassel, J. Bouchard, B. Bouchard, and A. Bouzouane, "Guidelines for increasing prompt efficiency in smart homes according to the resident's profile and task characteristics," *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 6719 LNCS, pp. 112–120, 2011.

[7] N. Ravi, N. Dandekar, P. Mysore, and M. L. Littman, "Activity recognition from accelerometer data," pp. 1541–1546, Jul. 2005.

[8] F. J. Ordóñez, J. A. Iglesias, P. De Toledo, A. Ledezma, and A. Sanchis, "Online activity recognition using evolving classifiers," *Expert Syst. Appl.*, vol. 40, no. 4, pp. 1248–1255, 2013.

[9] M. T. Moutacalli, A. Bouzouane, and B. Bouchard, "The behavioral profiling based on times series forecasting for smart homes assistance," *J. Ambient Intell. Humaniz. Comput.*, 2015.

[10] Norris, Sonya, and C. D. de la recherche parlementaire, *La maladie d'Alzheimer*. Direction de la recherche parlementaire, 2002.

[11] S. Gauthier, "Advances in the pharmacotherapy of Alzheimer's disease.," *CMAJ*, vol. 166, no. 5, pp. 616–23, Mar. 2002.

[12] C. Baum and D. F. Edwards, "Cognitive performance in senile dementia of the Alzheimer's type: the Kitchen Task Assessment.," *Am. J. Occup. Ther.*, vol. 47, no. 5, pp. 431–6, May 1993.

[13] C. Baum, D. F. Edwards, and N. Morrow-Howell, "Identification and measurement of productive behaviors in senile dementia of the Alzheimer type.," *Gerontologist*, vol. 33, no. 3, pp. 403–8, Jun. 1993.

[14] A. Jurek, C. Nugent, Y. Bi, and S. Wu, "Clustering-based ensemble learning for activity recognition in smart homes.," *Sensors (Basel).*, vol. 14, no. 7, pp. 12285–304, 2014.

[15] N. K. Suryadevara, S. C. Mukhopadhyay, R. Wang, and R. K. Rayudu, "Forecasting the behavior of an elderly using wireless sensors data in a smart home," *Eng. Appl. Artif. Intell.*, vol. 26, no. 10, pp. 2641–2652, 2013.

[16] D. J. C. Vikramaditya Jakkula, "Mining Sensor Data in Smart Environment for Temporal Activity Prediction," *13th ACM SIGKDD Int. Conf. Knowl. Discov. Data Min.*, 2007.

[17] E. H. Spriggs, F. De La Torre, and M. Hebert, "Temporal segmentation and activity classification from first-person sensing," *2009 IEEE Conf. Comput. Vis. Pattern Recognition, CVPR 2009*, pp. 17–24, 2009.

[18] G. E. P. Box, G. M. Jenkins, G. C. Reinsel, and G. M. Ljung, *Time Series Analysis: Forecasting and Control*. Wiley, 2015.

[19] M. T. Moutacalli, B. Bouchard, and A. Bouzouane, "Sensors Activation Times Prediction in Smart Home," in *8th ACM International Conference on Pervasive Technologies Related to Assistive Environments*, 2015.

[20] C. F. Schmidt, N. S. Sridharan, and J. L. Goodson, "The plan recognition problem: An intersection of psychology and artificial intelligence," *Artif. Intell.*, vol. 11, no. 1–2, pp. 45–83, Aug. 1978.

[21] L. R. Rabiner, "A tutorial on hidden Markov models and selected applications in speech recognition," *Proc. IEEE*, vol. 77, no. 2, pp. 257–286, 1989.

[22] J. Cheng, R. Greiner, J. Kelly, D. Bell, and W. Liu, "Learning Bayesian networks from data: An information-theory based approach," *Artif. Intell.*, vol. 137, no. 1–2, pp. 43–90, May 2002.

[23] J. Han, *Data Mining: Concepts and Techniques*. Morgan Kaufmann Publishers, 2001.

[24] H. A. Kautz, "A formal theory of plan recognition and its implementation," pp. 69–124, Jul. 1991.

[25] W. Wobcke, "Two Logical Theories of Plan Recognition," *J. Log. Comput.*, vol. 12, no. 3, pp. 371–412, Jun. 2002.

[26] T. van Kasteren, A. Noulas, G. Englebienne, and B. Kröse, "Accurate activity recognition in a home setting," in *Proceedings of the 10th international conference on Ubiquitous computing - UbiComp '08*, 2008, p. 1.

[27] M. Ghazvininejad, H. R. Rabiee, N. Pourdamghani, and P. Khanipour, "HMM based semi-supervised learning for activity recognition," in *Proceedings of the 2011 international workshop on Situation activity & goal awareness - SAGAware '11*, 2011, p. 95.

[28] T. Inomata, F. Naya, N. Kuwahara, F. Hattori, and K. Kogure, "Activity recognition from interactions with objects using dynamic Bayesian network," in *Proceedings of the 3rd ACM International Workshop on Context-Awareness for Self-Managing Systems - Casemans '09*, 2009, pp. 39–42.

[29] E. Kim, S. Helal, and D. Cook, "Human Activity Recognition and Pattern Discovery.," *IEEE Pervasive Comput.*, vol. 9, no. 1, p. 48, Jan. 2010.

[30] H. K. Donald J. Patterson, Lin Liao, Dieter Fox, "Inferring High-Level Behavior from Low-Level Sensors," *Proc. Fifth Int. Conf. Ubiquitous Comput.*, pp. 73–89, 2003.

[31] E. Alpaydin, *Introduction to Machine Learning*. MIT Press, 2014.

[32] J. Wu, *Advances in K-means Clustering: A Data Mining Thinking*. Springer Science & Business Media, 2012.

[33] M. T. Moutacalli, V. Marmen, A. Bouzouane, and B. Bouchard, "Activity pattern mining using temporal relationships in a smart home," in *2013 IEEE Symposium on Computational Intelligence in Healthcare and e-health (CICARE)*, 2013, pp. 83–87.

[34] J. Wang and J. Han, "BIDE: efficient mining of frequent closed sequences," in *Proceedings. 20th International Conference on Data Engineering*, pp. 79–90.

[35] M. T. Moutacalli, A. Bouzouane, and B. Bouchard, "New frequent pattern mining algorithm tested for activities models creation," in *IEEE Symposium Series on Computational Intelligence (SSCI)*, 2014.

[36] S. Han, Y. Dang, S. Ge, and D. Zhang, *Frequent Pattern Mining*. Springer, 2012.

[37] A. Bouzouaneuqacca, B. Bouchard, and B. Boucharduqacca, "Unsupervised Activity Recognition using Temporal Data Mining," no. c, pp. 15–20, 2012.

[38] M. T. Moutacalli, K. Bouchard, B. Bouchard, and A. Bouzouane, "Activity prediction based on time series forcasting," in *AAAI Workshop on Artificial Intelligence Applied to Assistive Technologies and Smart Environments (ATSE'14)*, 2014.

[39] U. Fayyad, G. Piatetsky-Shapiro, and P. Smyth, "From Data Mining to Knowledge Discovery in Databases," *AI Magazine*, vol. 17, no. 3. p. 37, 15-Mar-1996.

[40] M. H. Dunham, *Data Mining: Introductory And Advanced Topics*. Pearson Education, 2006.

[41] E. Dumbill, *Planning for Big Data*. "O'Reilly Media, Inc.," 2012.

[42] Ubiquitous Computing Fundamentals. CRC Press, 2009.

[43] C. Lynch, "Big data: How do your data grow?," *Nature*, vol. 455, no. 7209, pp. 28–9, Sep. 2008.

[44] A. Boicea, F. Radulescu, and L. I. Agapin, "MongoDB vs Oracle -- Database Comparison," in *2012 Third International Conference on Emerging Intelligent Data and Web Technologies*, 2012, pp. 330–335.

[45] R. Hecht and S. Jablonski, "NoSQL evaluation: A use case oriented survey," in *2011 International Conference on Cloud and Service Computing*, 2011, pp. 336–341.

[46] K. Shvachko, H. Kuang, S. Radia, and R. Chansler, "The Hadoop Distributed File System," in *2010 IEEE 26th Symposium on Mass Storage Systems and Technologies (MSST)*, 2010, pp. 1–10.

[47] S. J. Russell and P. Norvig, *Artificial Intelligence: A Modern Approach*. Prentice Hall, 2010.

[48] J. Saragih, "Principal regression analysis," in *CVPR 2011*, 2011, pp. 2881–2888.

[49] P. Borne, M. Benrejeb, and J. Haggège, *Les réseaux de neurones: présentation et applications*. Editions OPHRYS, 2007.

[50] R.-E. Fan, K.-W. Chang, C.-J. Hsieh, X.-R. Wang, and C.-J. Lin, "LIBLINEAR: A Library for Large Linear Classification," *J. Mach. Learn. Res.*, vol. 9, pp. 1871–1874, Jun. 2008.

[51] D. A. Grossman and O. Frieder, *Information Retrieval: Algorithms and Heuristics*. Springer Science & Business Media, 1998.

[52] S. Santini and R. Jain, "Similarity measures," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 21, no. 9, pp. 871–883, 1999.

[53] M. Rifaie, K. Kianmehr, R. Alhajj, and M. J. Ridley, "Data warehouse architecture and design," in *2008 IEEE International Conference on Information Reuse and Integration*, 2008, pp. 58–63.

[54] Data mining federal efforts cover a wide range of uses. DIANE Publishing, 2004.

[55] G. Benoît, "Data mining," *Annu. Rev. Inf. Sci. Technol.*, vol. 36, no. 1, pp. 265–310, Feb. 2005.

[56] M. Shahnawaz, A. Ranjan, and M. Danish, "Temporal Data Mining : An Overview," no. 1, pp. 20–24, 2011.

[57] S. M. Weiss, N. Indurkhya, T. Zhang, and F. Damerau, *Text Mining: Predictive Methods for Analyzing Unstructured Information*. Springer Science & Business Media, 2010.

[58] A. A. Freitas, "On rule interestingness measures," *Knowledge-Based Syst.*, vol. 12, no. 5–6, pp. 309–315, Oct. 1999.

[59] A. U. Tansel, J. Clifford, S. Gadia, S. Jajodia, A. Segev, and R. Snodgrass, "Temporal databases: theory, design, and implementation," *Benjamin-Cummings Pub Co*, 1993.

[60] L. Chittaro, C. Combi, and G. Trapasso, "Data mining on temporal data: a visual approach and its clinical application to hemodialysis," *J. Vis. Lang. Comput.*, vol. 14, no. 6, pp. 591–620, Dec. 2003.

[61] T. Mitsa, "Temporal Data Mining," *Chapman & Hall/CRC*, Mar. 2010.

[62] D. Hand, H. Mannila, and P. Smyth, *Principles of Data Mining*, vol. 2001. 2002.

[63] J. T. Behrens, "Principles and procedures of exploratory data analysis," *Psychol. Methods*, pp. 131–160, 1997.

[64] J. Strickland, *Predictive Modeling and Analytics*. Lulu.com, 2014.

[65] D. Lo and S.-C. Khoo, "Mining patterns and rules for software specification discovery," *Proc. VLDB Endow.*, vol. 1, no. 2, pp. 1609–1616, Aug. 2008.

[66] H. Jantan, A. R. Hamdan, and Z. A. Othman, "Potential Data Mining Classification Techniques for Academic Talent Forecasting," in *2009 Ninth International Conference on Intelligent Systems Design and Applications*, 2009, pp. 1173–1178.

[67] S. Laxman and P. Sastry, "A survey of temporal data mining," *Sadhana*, vol. 31, no. April, pp. 173–198, 2006.

[68] B. Gold, N. Morgan, and D. Ellis, *Speech and Audio Signal Processing: Processing and Perception of Speech and Music*, vol. 1. John Wiley & Sons, 2011.

[69] C. C. Tappert, C. Y. Suen, and T. Wakahara, “The state of the art in online handwriting recognition,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 12, no. 8, pp. 787–808, 1990.

[70] Z. Ren, J. Yuan, J. Meng, and Z. Zhang, “Robust Part-Based Hand Gesture Recognition Using Kinect Sensor,” *IEEE Trans. Multimed.*, vol. 15, no. 5, pp. 1110–1120, Aug. 2013.

[71] I. H. Witten, E. Frank, and M. A. Hall, *Data Mining: Practical Machine Learning Tools and Techniques: Practical Machine Learning Tools and Techniques*. Elsevier, 2011.

[72] E. G. M. de Lacerda, A. C. P. L. F. de Carvalho, and T. B. Ludermir, “A study of cross-validation and bootstrap as objective functions for genetic algorithms,” in *VII Brazilian Symposium on Neural Networks, 2002. SBRN 2002. Proceedings.*, 2002, pp. 118–123.

[73] R. Xu and D. Wunsch, *Clustering*, vol. 3. John Wiley & Sons, 2008.

[74] C. K. R. Charu C. Aggarwal, *Data Clustering: Algorithms and Applications*. CRC Press, 2013.

[75] S. Valsamidis, S. Kontogiannis, I. Kazanidis, T. Theodosiou, and A. Karakos, “A Clustering Methodology of Web Log Data for Learning Management Systems.,” *Educ. Technol. Soc.*, vol. 15, no. 2, pp. 154–167, Nov. 2011.

[76] H.-S. Guan and Q. Jiang, “Cluster financial time series for portfolio,” in *2007 International Conference on Wavelet Analysis and Pattern Recognition*, 2007, vol. 2, pp. 851–856.

[77] N. Osato, M. Itoh, H. Konno, S. Kondo, K. Shibata, P. Carninci, T. Shiraki, A. Shinagawa, T. Arakawa, S. Kikuchi, K. Sato, J. Kawai, and Y. Hayashizaki, “A computer-based method of selecting clones for a full-length cDNA project: simultaneous collection of negligibly redundant and variant cDNAs.,” *Genome Res.*, vol. 12, no. 7, pp. 1127–34, Jul. 2002.

[78] M.-S. Yang, “A survey of fuzzy clustering,” *Math. Comput. Model.*, vol. 18, no. 11, pp. 1–16, Dec. 1993.

[79] J. C. Bezdek, R. Ehrlich, and W. Full, “FCM: The fuzzy c-means clustering algorithm,” *Comput. Geosci.*, vol. 10, no. 2–3, pp. 191–203, Jan. 1984.

[80] T. A. Schonhoff and A. A. Giordano, *Detection and Estimation Theory and Its Applications*. Prentice Hall, 2006.

[81] R. Agrawal and R. Srikant, “Fast Algorithms for Mining Association Rules in Large Databases,” pp. 487–499, Sep. 1994.

[82] S. Moens, E. Aksehirli, and B. Goethals, “Frequent Itemset Mining for Big Data,” in *2013 IEEE International Conference on Big Data*, 2013, pp. 111–118.

[83] J. Han, J. Pei, and Y. Yin, “Mining frequent patterns without candidate generation,” *ACM SIGMOD Rec.*, vol. 29, no. 2, pp. 1–12, Jun. 2000.

[84] X. Yan, J. Han, and R. Afshar, “CloSpan: Mining Closed Sequential Patterns in Large Databases.,” in *Proceedings of the Third SIAM International Conference on Data Mining, San Francisco, CA, USA, May 1-3, 2003*, 2003.

[85] W. Wang and J. Yang, *Mining Sequential Patterns from Large Data Sets*. Springer Science & Business Media, 2006.

[86] M. T. Moutacalli, “Une approche de reconnaissance d’activités utilisant la fouille de données temporelles.” mémoire de maîtrise, Université du Québec À Chicoutimi, 11-Apr-2012.

[87] M. F. Schwartz, M. Segal, T. Veramonti, M. Ferraro, and L. J. Buxbaum, “The Naturalistic Action Test: A standardised assessment for everyday action impairment,” *Neuropsychol. Rehabil.*, Oct. 2010.

[88] W. A. Rogers, B. Meyer, N. Walker, and A. D. Fisk, “Functional limitations to daily living tasks in the aged: a focus group analysis.,” *Hum. Factors*, vol. 40, no. 1, pp. 111–25, Mar. 1998.

[89] C. W. Geib and R. P. Goldman, “Plan recognition in intrusion detection systems,” in *Proceedings DARPA Information Survivability Conference and Exposition II. DISCEX’01*, 2001, vol. 1, pp. 46–55.

[90] L. Chen and I. Khalil, *Activity Recognition in Pervasive Intelligent Environments*, vol. 4. Paris: Atlantis Press, 2011.

[91] J. Crowley, *Computer Vision Systems: Third International Conference, ICVS 2003, Graz, Austria, April 1-3, 2003, Proceedings, Volume 3*. Springer Science & Business Media, 2003.

[92] T. V. Duong, H. H. Bui, D. Q. Phung, and S. Venkatesh, “Activity Recognition and Abnormality Detection with the Switching Hidden Semi-Markov Model,” in *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR’05)*, vol. 1, pp. 838–845.

[93] J. Barbič, A. Safonova, J.-Y. Pan, C. Faloutsos, J. K. Hodgins, and N. S. Pollard, “Segmenting motion capture data into distinct behaviors,” pp. 185–194, May 2004.

[94] Y. Zhang, M. Alder, and R. Togneri, “Using Gaussian Mixture Modeling for Phoneme Classification,” *Univ. West. Aust.*, Dec. 1993.

[95] T. Cover and P. Hart, “Nearest neighbor pattern classification,” *IEEE Trans. Inf. Theory*, vol. 13, no. 1, pp. 21–27, Jan. 1967.

[96] A. P. Daniel Olguín Olguín, “Human activity recognition: Accuracy across common locations for wearable sensors,” *PROC. 10TH INT. SYMP. WEARABLE Comput.*, pp. 11–13, 2006.

[97] scar D. Lara and M. A. Labrador, “A mobile platform for real-time human activity recognition,” in *2012 IEEE Consumer Communications and Networking Conference (CCNC)*, 2012, pp. 667–671.

[98] Z. Xiang and L. Zhang, “Research on an Optimized C4.5 Algorithm Based on Rough Set Theory,” in *2012 International Conference on Management of e-Commerce and e-Government*, 2012, pp. 272–274.

[99] U. Maurer, A. Smailagic, D. P. Siewiorek, and M. Deisher, “Activity Recognition and Monitoring Using Multiple Sensors on Different Body Positions,” in *International Workshop on Wearable and Implantable Body Sensor Networks (BSN’06)*, pp. 113–116.

[100] D. Ding, R. A. Cooper, P. F. Pasquina, and L. Fici-Pasquina, “Sensor technology for smart homes.,” *Maturitas*, vol. 69, no. 2, pp. 131–6, Jul. 2011.

[101] S. A. K. A. Omari and P. Sumari, “An Overview of Mobile Ad Hoc Networks for the Existing Protocols and Applications,” *Int. J. Appl. Graph Theory Wirel. Ad Hoc Networks Sens. Networks*, vol. 2, no. 1, pp. 87–110, Mar. 2010.

[102] Z. Wang and X. Sun, “An Efficient Spam Filtering Algorithm Based on NPE,” in *2008 IEEE International Symposium on Knowledge Acquisition and Modeling Workshop*, 2008, pp. 1102–1104.

[103] G. F. James F. Allen, “Actions and Events in Interval Temporal Logic,” *J. Log. Comput.*, vol. 4, no. 5, pp. 531–579, 1994.

[104] P. P. Angelov, “Evolving Fuzzy-Rule-Based Classifiers From Data Streams,” *IEEE Trans. Fuzzy Syst.*, vol. 16, no. 6, pp. 1462–1475, Dec. 2008.

[105] P. Angelov and D. Filev, “Simpl_eTS: a simplified method for learning evolving Takagi-Sugeno fuzzy models,” in *The 14th IEEE International Conference on Fuzzy Systems, 2005. FUZZ ’05.*, pp. 1068–1073.

[106] P. P. Angelov and D. P. Filev, “An Approach to Online Identification of Takagi-Sugeno Fuzzy Models,” *IEEE Trans. Syst. Man Cybern. Part B*, vol. 34, no. 1, pp. 484–498, Feb. 2004.

[107] P. Angelov, C. Xydeas, and D. Filev, “On-line identification of MIMO evolving Takagi- Sugeno fuzzy models,” in *2004 IEEE International Conference on Fuzzy Systems (IEEE Cat. No.04CH37542)*, vol. 1, pp. 55–60.

[108] P. J. Brockwell and R. A. Davis, *Introduction to Time Series and Forecasting*. Springer Science & Business Media, 2013.

[109] D. Kwiatkowski, P. C. B. Phillips, P. Schmidt, and Y. Shin, “Testing the null hypothesis of stationarity against the alternative of a unit root: How sure are we that economic time series have a unit root?,” *J. Econom.*, vol. 54, no. 1–3, pp. 159–178, 1992.

[110] L. K. Sen and M. Shitan, “The Performance of AICC as an Order Selection Criterion in ARMA Time Series Models,” *GE, Growth, Math methods*, Jul. 2003.

[111] D. J. Cook, M. Youngblood, E. O. Heierman, K. Gopalratnam, S. Rao, A. Litvin, and F. Khawaja, “MavHome: an agent-based smart home,” in *Proceedings of the First IEEE International Conference on Pervasive Computing and Communications, 2003. (PerCom 2003).*, pp. 521–524.

[112] L. Kaila, J. Mikkonen, A.-M. Vainio, and J. Vanhala, “The eHome - a Practical Smart Home Implementation,” Jan. 2008.

[113] H. Pigot, B. Lefebvre, J. G. Meunier, B. Kerhervé, A. Mayers, and S. Giroux, “The role of intelligent habitats in upholding elders in residence,” 2003.

[114] S. Helal, W. Mann, H. El-Zabadani, J. King, Y. Kaddoura, and E. Jansen, “The Gator Tech Smart House: a programmable pervasive space,” *Computer (Long. Beach. Calif).*, vol. 38, no. 3, pp. 50–60, Mar. 2005.

[115] T. Lee and A. Mihailidis, “An intelligent emergency response system: preliminary development and testing of automated fall detection.,” *J. Telemed. Telecare*, vol. 11, no. 4, pp. 194–8, Jan. 2005.

[116] C. Phua, J. Biswas, A. Tolstikov, V. S. F. Foo, W. Huang, P. Jayachandran, A. A. P. Wai, H. Aloulou, and M. A. Feki, "Plan recognition based on sensor produced micro-context for eldercare," *International Workshop on Context-Awareness in Smart Environments: Background, Achievements and Challenges*. pp. 39–46, 16-Jun-2009.

[117] S. S. Intille, "Designing a home of the future," *IEEE Pervasive Comput.*, vol. 1, no. 2, pp. 76–82, Apr. 2002.

[118] M. Bouet and A. L. dos Santos, "RFID tags: Positioning principles and localization techniques," in *2008 1st IFIP Wireless Days*, 2008, pp. 1–5.

[119] P. Evensen and H. Meling, "Sensor virtualization with self-configuration and flexible interactions," in *Proceedings of the 3rd ACM International Workshop on Context-Awareness for Self-Managing Systems - Casemans '09*, 2009, pp. 31–38.

[120] J. Lapointe, B. Bouchard, J. Bouchard, A. Potvin, and A. Bouzouane, "Smart homes for people with Alzheimer's disease," in *Proceedings of the 5th International Conference on PErvasive Technologies Related to Assistive Environments - PETRA '12*, 2012, p. 1.

[121] C. Hekimian-Williams, B. Grant, and P. Kumar, "Accurate localization of RFID tags using phase difference," in *2010 IEEE International Conference on RFID (IEEE RFID 2010)*, 2010, pp. 89–96.