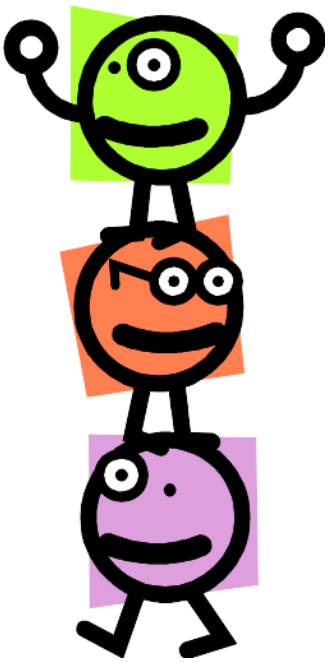
Découverte interactive et complète de motifs temporels intéressants à partir de traces d'interactions



Rapport de recherche pour le projet PROCOGEC (http://www.procogec.com)
Livrable de la tâche T-3.3 intitulée :
« méthodes et outils pour l'ingénierie des connaissances et l'apprentissage en situation collaborative »

Damien Cram

7 Novembre 2008

Table des matières

Chapitre 1

Introduction

Dans un précédent rapport d'avancement (Cram 2007*a*), nous avons justifié pourquoi nous envisageons l'assistance aux utilisateurs d'un système informatique par l'intermédiaire d'un cycle interactif d'abstraction des traces d'interactions. Nous avons précisé cette approche dans (Cram et al. 2008) et expliqué que ce cycle interactif doit mettre en jeu un *agent analyseur* à qui revient la charge de fouiller la trace d'interactions de l'utilisateur afin de lui proposer des motifs candidats à l'abstraction. C'est dans cette optique que nous avions parcouru l'état de l'art en matière de fouille de données temporelles (Cram 2007*b*).

L'objet de ce rapport est de proposer une première méthode de fouille de traces pour l'implémentation future de cet *agent analyseur*. Il existe de nombreux travaux d'extraction d'épisodes fréquents à partir de séquences d'évènements (Mannila et al. 1997, Casas-Garriga 2003, Méger 2004, Laxman et al. 2007*a*, Huang & Chang 2008). Ces méthodes permettent d'extraire des motifs soit «en série», soit «en parallèle», c'est-à-dire qu'elles permettent d'extraire soit des motifs qui spécifient totalement l'ordre d'apparition des évènements dans la séquence, soit des motifs qui ne spécifient aucun ordre. Dans la réalité du domaine d'application, il se trouve que ces deux aspects du problème sont trop limitatifs, car dans le cas général les situations réelles se décrivent avec plus de souplesse sur l'ordre des évènements. Le cas des motifs dits «hybrides» est traité dans certains, mais il s'avère que dans la pratique ils sont très difficiles à découvrir. De plus, ces méthodes ne permettent pas de quantifier temporellement les écarts entre les évènements du motif. À notre connaissance, la seule méthode qui permette d'extraire des motifs temporels en comblant ces manques est la *découverte de chroniques fréquentes* (Dousson & Duong 1999, Duong 2001), c'est pourquoi nous avons décidé de baser nos travaux sur celle-ci. En réalité ces méthodes présentent d'autres manques pour l'application aux traces d'interactions, mais comme nous ne les traitons pas dans ce rapport, nous ne les évoquerons qu'avec les travaux à venir. (cf. chapitre 4.1)

Il est largement admis par la communauté de «la découverte de connaissances à partir de données» que le processus de découverte implique des interactions avec l'utilisateur lors de ses différentes étapes (Han & Kamber 2006). Néanmoins, le problème de la découverte de connaissances à partir de données reste vu comme la recherche et l'optimisation de méthodes permettant de livrer toute la connaissance contenue dans les données à l'utilisateur, sans que ce dernier ait trop à intervenir. Cette vision du domaine provient du souci constant et

parfois biaisant qu'ont les informaticiens de vouloir tout automatiser dans une «boîte intelligente» afin de limiter au maximum les efforts que l'utilisateur doit fournir de lui-même. Nous pensons à l'inverse que le processus de découverte de connaissances à partir de données doit par nature s'effectuer en interaction avec l'utilisateur et qu'il y a beaucoup à tirer de l'intervention de l'utilisateur. C'est la méthode que nous proposons dans ce rapport en ce basant sur le travail existant de découverte de chroniques (Dousson & Duong 1999, Duong 2001).

La suite du rapport s'organise comme suit. Le chapitre 2 présente la méthode de découverte de chroniques fréquentes à partir de séquences de (Duong 2001) et ses principes, et pose les notations pour la suite. Le chapitre 3 propose une extension de cette méthode, assurant la complétude du processus de découverte et permettant facilement l'introduction d'heuristiques et de contraintes. Enfin, le chapitre 4 conclut sur le travail qui a été réalisé dans ce rapport et expose brièvement les futures extensions envisagées de notre méthode afin d'être plus adaptée encore à la structure des traces d'interactions et à nos besoins.

Chapitre 2

Processus de découverte de chroniques existant

2.1 Définitions

L'extraction de chroniques est un processus qui prend en entrée une *séquence d'évènements*, les données à fouiller, et renvoie en sortie un ensemble de motifs temporels sur les évènements, appelés *chroniques*. Avant de pouvoir définir ces concepts, on suppose qu'on dispose dans tout le rapport d'un ensemble fini de types d'évènement $\mathbb{E}$, et d'un ensemble $(\mathbb{T}, \leq_{\mathbb{T}})$ totalement ordonné, appelé *domaine temporel*, et muni d'un opérateur de composition interne $+_{\mathbb{T}}$ telle que $(\mathbb{T}, +_{\mathbb{T}})$ soit un groupe. Que $(\mathbb{T}, +_{\mathbb{T}})$ soit un groupe signifie notamment qu'il est possible d'additionner et de soustraire deux éléments de $\mathbb{T}$ tout en restant dans $\mathbb{T}$. Pour des raisons pratiques, on suppose que $\mathbb{E}$ est également totalement ordonné par $\leq_{\mathbb{E}}$.

2.1.1 Séquence d'évènements

Définition 1 (Séquence d'évènements)
Une séquence d'évènements est un ensemble d'évènements, où chaque évènement est un couple (e, t) tel que $e \in \mathbb{E}$, et $t \in \mathbb{T}$. e est le type d'évènement *de l'évènement et t la* date*. Formellement, une séquence $\mathcal{S}$ de types d'évènement de $\mathbb{E}$ sur $\mathbb{T}$ se définit comme suit :*

- $\mathcal{S} = \langle (e_i, t_i) \rangle_{i=1}^{n_{\mathcal{S}}}$;
- *pour tout i vérifiant $1 \leq i \leq n_{\mathcal{S}}$, on a $e_i \in \mathbb{E}$ et $t_i \in \mathbb{T}$;*
- *pour tout (i, j) vérifiant $1 \leq i < j \leq n_{\mathcal{S}}$, on a $t_i \leq_{\mathbb{T}} t_j$ ou $(t_i = t_j$ et $e_i \leq_{\mathbb{E}} e_j)$.*

Le nombre d'évènements de S (ici $n_{\mathcal{S}}$) est appelé la taille *de $\mathcal{S}$, et se note $|\mathcal{S}|$. On note $S(\mathbb{E}, \mathbb{T})$ l'ensemble des séquences d'évènements de $\mathbb{E}$ sur $\mathbb{T}$.*

Dans la suite du rapport, on suppose que $\mathbb{T}$ est l'ensemble des nombres relatifs $\mathbb{Z}$. La figure 2.1 donne en exemple, avec $\mathbb{E} = \{A, B, C\}$ et $\mathbb{T} = \mathbb{Z}$, une représentation de la séquence d'évènements $\mathcal{S}_0$, où :

$$\mathcal{S}_0 = \langle (A,1)(B,3)(A,4)(C,4)(A,7)(B,8)(C,9)(B,10)(B,12) \rangle$$

FIG. 2.1 – Exemple de séquence d'évènements : séquence $\mathcal{C}_0$.

2.1.2 Chronique

Définition 2 (Contrainte temporelle)
Une contrainte temporelle est un couple d'éléments de $\mathbb{T}$*, noté* $[i, j]$*, tel que* $i \leq_{\mathbb{T}} j$*.*

Si $\mathbb{T}$ possède une borne inférieure et une borne supérieure, alors notons respectivement $-\infty_{\mathbb{T}}$ et $\infty_{\mathbb{T}}$ ces évènements. Si ce n'est pas le cas, on ferme $\mathbb{T}$ en ajoutant à $\mathbb{T}$ l'élément $-\infty_{\mathbb{T}}$ tel que $\forall t \in \mathbb{T}, -\infty_{\mathbb{T}} \leq t$, et de même l'élément $\infty_{\mathbb{T}}$.

Définition 3 (Chronique)
Une chronique $\mathcal{C}$ *est un couple* $(\mathcal{A}, \mathcal{T})$ *où* $\mathcal{A}$ *est une liste de types d'évènement de* $\mathbb{E}$ *(*$\mathcal{A}$ *peut contenir plusieurs fois le même type d'évènement), et* $\mathcal{T}$ *est une matrice strictement triangulaire de contraintes temporelles et de taille* n *sur* $\mathcal{A}$*. On note :*

1. $\mathcal{A} = \{a_1, \ldots, a_n\}$,
2. $\mathcal{T} = (\tau_{ij})_{1 \leq i < j \leq n}$,
3. *pour tout* (i, j) *vérifiant* $1 \leq i < j \leq n$, $\tau_{ij} = [\tau_{ij}^-, \tau_{ij}^+]$, *avec* $(\tau_{ij}^-, \tau_{ij}^+) \in \mathbb{T}^2$*, ou aussi* $\tau_{ij} = a_i[\tau_{ij}^-, \tau_{ij}^+]a_j$.

Les a_i sont les types d'évènement de la chronique, parfois aussi appelés abusivement *évènements*. Chaque τ_{ij} est appelée la *contrainte temporelle* du couple de types d'évènement (a_i, a_j). Il est possible que $\tau_{ij}^- = -\infty_{\mathbb{T}}$ et que $\tau_{ij}^+ = \infty_{\mathbb{T}}$. Si $\tau_{ij} = [-\infty_{\mathbb{T}}, \infty_{\mathbb{T}}]$, alors on dira parfois abusivement qu'il n'y a pas de contraintes entre a_i et a_j.

Les évènements de $\mathcal{A}$ forment un épisode parallèle $a_1 \ldots a_n$. Dans la suite on dira réciproquement qu'une chronique est basée sur un épisode $a_1 \ldots a_n$ si son ensemble $\mathcal{A}$ est exactement constitué des a_i $(1 \leq i \leq n)$.

Les chroniques sont des réseaux de contraintes temporelles (Dechter et al. 1991) et peuvent se représenter comme des graphes de contraintes. La figure 2.2 donne l'exemple de la chronique $\mathcal{C}_0$ où $\mathcal{A}_0 = \{A, B, C\}$ et où $\mathcal{T}_0$ est composé des 3 contraintes suivantes : la contrainte $[2, 4]$ entre les types d'évènements A et B et notée $A[2, 4]B$, la contrainte $[1, 1]$ entre les types d'évènements B et C (notée $B[1, 1]C$) et la contrainte $[3, 5]$ entre les types d'évènements A et C (notée $A[3, 5]C$). On adopte pour les chroniques la notation suivante (avec $\mathcal{C}_0$) :

$$\mathcal{C}_0 = \langle ABC : A[2, 4]B, B[1, 1]C, A[3, 5]C \rangle$$

Les chroniques ainsi définies sont des motifs temporels qui signifient par exemple pour $\mathcal{C}_0$ que dans la séquence, deux évènements A et B doivent apparaître dans cet ordre et ne pas être espacés de moins de 2 unités de temps,

ni de plus de 4 unités de temps pour être représentés par la chronique. Pour alléger la représentation, lorsque la contrainte entre deux types d'évènement est $[-\infty_{\mathbb{T}}, \infty_{\mathbb{T}}]$, on ne représente pas la contrainte sur le graphe (cf. figure 2.3).

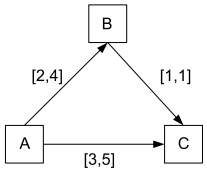


FIG. 2.2 – Exemple de chronique : chronique $\mathcal{C}_0$.

Définition 4 (Instance de chronique)
Une instance d'une chronique $\mathcal{C} = (\mathcal{A}, \mathcal{T})$*, en notant* $\mathcal{A} = (a_1, \ldots, a_n)$ *et* $\mathcal{T} = (\tau_{ij})_{1 \leq i < j \leq n}$*, est un ensemble de couples de la forme* $\{(a_1, t_{a_1}), \ldots, (a_n, t_{a_n})\}$*, tel que pour tout couple* (i, j) *vérifiant* $1 \leq i < j \leq n$*, on a* $t_{a_j} - t_{a_i} \in \tau_{ij}$*.*

On note $\mathcal{I}_{\mathcal{C}}$ l'ensemble des instances de $\mathcal{C}$.

Définition 5 (Occurrence de chronique)
Une occurrence d'une chronique $\mathcal{C} = (\mathcal{A}, \mathcal{T})$ *dans* $\mathcal{S}$ *est un ensemble d'évènements* $\mathcal{O} = \{(e_i, t_i)\}_{i=1}^{n}$ *tel que :*

1. *pour tout* i *vérifiant* $1 \leq i \leq n$*, on* $(e_i, t_i) \in \mathcal{S}$*,*
2. *pour tout* (e_i, t_i) *et* (e_j, t_j) *tels que* $e_i \leq_{\mathbb{E}} e_j$*, on a* $t_j - t_i \in \tau_{ij}$*.*

Autrement dit, une occurrence de $\mathcal{C} = (\mathcal{A}, \mathcal{T})$ dans $\mathcal{S}$ est un sous-ensemble noté $\mathcal{O}_{\mathcal{C}}$ d'évènements de $\mathcal{S}$ tel que tout couple d'évènements de $\mathcal{O}_{\mathcal{C}}$ vérifie la contrainte temporelle imposée par $\mathcal{T}$. Par exemple, parmi les occurrences de la chronique $\mathcal{C}_0$ dans la séquence $\mathcal{S}_0$, il y a :

$$(A, 1)(B, 3)(C, 4)$$
$$(A, 1)(B, 8)(C, 9)$$

2.1.3 Chroniques plus strictes

Définition 6 (Inclusion de contraintes temporelles)
Une contrainte temporelle $[a', b']$ *est dite* incluse *dans une une contrainte temporelle* $[a, b]$ *si et seulement si* $a' \geq a$ *et* $b' \leq b$*. On note alors* $[a', b'] \subseteq [a, b]$*.*

On dit également que $[a', b']$ est *plus stricte* que $[a, b]$.

Définition 7 (Chronique plus stricte)
Une chronique $\mathcal{C}' = (\mathcal{A}', \mathcal{T}')$ *est dite* plus stricte *qu'une chronique* $\mathcal{C} = (\mathcal{A}, \mathcal{T})$ *si et seulement si les deux conditions suivantes sont réalisées :*

1. $\mathcal{A} \subseteq \mathcal{A}'$*,*
2. *en notant* $\mathcal{A} = (a_1, \ldots, a_n)$ *et* $\mathcal{A}' = (a'_1, \ldots, a'_{n'})$*, et en notant* h *la fonction qui associe à tout* i'*, l'indice* i *de* $\mathcal{A}$ *tel que* $a_i = a'_{i'}$*, on a pour tout* $\tau'_{i'j'} \in \mathcal{T}'$*,* $\tau'_{i'j'} \subseteq \tau_{h(i')h(j')}$*.*

On note $\mathcal{C}' \preceq \mathcal{C}$*.*

On dit également que $\mathcal{C}'$ est *plus contrainte* que $\mathcal{C}$ ou que $\mathcal{C}$ est *plus générale* que $\mathcal{C}'$. Par abus, on dira même parfois que $\mathcal{C}'$ est incluse dans $\mathcal{C}$, même si pour cela il faut que $\mathcal{A} \subseteq \mathcal{A}'$. La figure 2.3 montre un exemple de chronique plus stricte.

FIG. 2.3 – Exemple d'inclusion de chroniques.

La relation définie par l'inclusion des chroniques est une relation d'ordre partielle sur l'ensemble des chroniques. À partir de n'importe quel ensemble fini de chroniques il est donc possible de construire un graphe acyclique orienté d'inclusion de chroniques semblable à celui de la figure 2.4.

Définition 8 (Chronique minimale)
Étant donné un ensemble de chroniques $C = \{\mathcal{C}_1, \mathcal{C}_2, \ldots, \mathcal{C}_n\}$*, une chronique* $\mathcal{C}_i$ *de* C *est dite* chronique minimale de C *s'il n'existe aucune autre chronique* $\mathcal{C}_j$ *de* C *telle que* $\mathcal{C}_j \preceq \mathcal{C}_i$*. L'ensemble des chroniques minimales de* C *est noté* $min(C)$*.*

Par exemple, sur la figure 2.4, les chroniques minimales sont $\mathcal{C}_3$, $\mathcal{C}_5$, $\mathcal{C}_6$, $\mathcal{C}_8$, $\mathcal{C}_9$, $\mathcal{C}_{12}$.

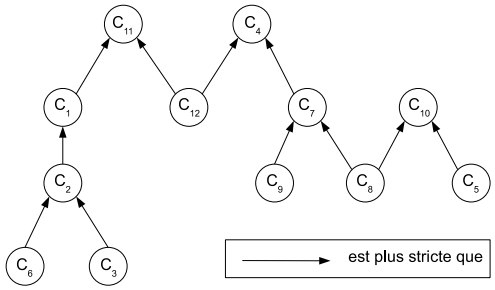


FIG. 2.4 – Graphe d'inclusion d'un ensemble de chronique $C = \{\mathcal{C}_1, \mathcal{C}_2, \ldots, \mathcal{C}_{12}\}$.

De la même manière on peut définir ce qu'est une chronique maximale d'un ensemble de chroniques.

Définition 9 (Chronique maximale)
Étant donné un ensemble de chroniques $C = \{\mathcal{C}_1, \mathcal{C}_2, \ldots, \mathcal{C}_n\}$*, une chronique* $\mathcal{C}_i$ *de* C *est dite* chronique maximale de C *s'il n'existe aucune autre chronique* $\mathcal{C}_j$ *de* C *telle que* $\mathcal{C}_i \preceq \mathcal{C}_j$*. L'ensemble des chroniques minimales de* C *est noté* $max(C)$*.*

2.1.4 Chroniques propagées et chroniques équivalentes

Une chronique est un graphe de contraintes, et de ce fait une chronique peut être *consistante* ou *inconsistante*. Une chronique $\mathcal{C} = (\mathcal{A}, \mathcal{T})$ est *k-consistante*

($k \geq 2$) si et seulement si tous les chemins de taille k sont consistants, c'est à dire si pour tout chemin passant par $k+1$ noeuds $a_{i_1}, \ldots, a_{i_{k+1}}$, on peut trouver des dates $t_{i_1}, \ldots, t_{i_{k+1}}$ telles que $(a_{i_1}, t_{i_1}), \ldots, (a_{i_{k+1}}, t_{i_{k+1}})$ soit une occurrence de $\mathcal{C}$. On dit qu'une chronique est *consistante* si pour tout $k > 2$ elle est *k-consistante*. D'après les travaux sur les graphes de contraintes de (Montanari 1974), si une chronique est *2-consistante* alors elle est *consistante*.

Du fait que les chroniques sont des graphes de contraintes, il est possible qu'au sein d'une chronique, certaines valeurs d'écarts entre certains évènements soient en pratique impossibles à atteindre si on souhaite satisfaire toute les contraintes. La figure 2.5 illustre ce phénomène. La chronique $\mathcal{C}'_0$ comporte la contrainte $A[3,7]B$, mais en réalité il est impossible que l'écart entre A et C prenne la valeur 6 ou 7, car cela violerait l'une des deux autres contraintes. $\mathcal{C}'_0$ est donc de ce fait équivalente à $\mathcal{C}_0$. Les algorithmes de *path consistency* comme `PC-2` (Mackworth & Freuder 1985) ont pour charge de propager les contraintes dans un réseau de contraintes afin de supprimer toutes les valeurs impossibles pour chaque noeud. En appliquant ces méthodes aux chroniques, on obtient pour chaque chronique $\mathcal{C}$ une chronique «propagée» $\mathcal{C}'$ plus stricte que $\mathcal{C}$ dite équivalent à $\mathcal{C}$. On dira que deux chroniques sont équivalents si elles ont la même chronique propagée. On note $Eq(\mathcal{C})$ l'ensemble des chroniques équivalentes à $\mathcal{C}$. On peut prouver qu'il existe une et une seule chronique minimale de $Eq(\mathcal{C})$, notée $prop(\mathcal{C})$. Sur la figure 2.5, on a $\mathcal{C}_0 = prop(\mathcal{C}'_0)$. Pour dire que deux chroniques $\mathcal{C}_1$ et $\mathcal{C}_2$ sont équivalentes on utilisera le symbole d'égalité «=» ($\mathcal{C}_1 = \mathcal{C}_2$).

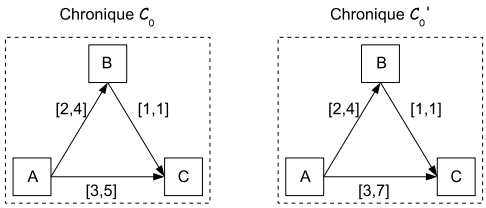


FIG. 2.5 – Deux chroniques équivalentes.

Une chronique *inconsistante* est une chronique qui ne peut pas avoir d'occurrence dans aucune séquence du fait de contraintes temporelles trop strictes. La figure 2.6 montre un exemple de chronique inconsistante. La contrainte entre A et C de $\mathcal{C}$ interdit que C soit à plus de 1 unité de temps après A, or d'après les contraintes entre A et B d'une part et B et C d'autre part, C ne peut arriver qu'au moins 3 unités de temps après A. L'algorithme `PC-2`, qui propage les domaines de valeur de noeud en noeud permet de détecter les inconsistances dès lors qu'un domaine est vide.

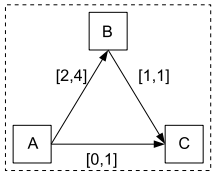


FIG. 2.6 – Exemple de chronique inconsistante.

2.2 Processus existant de découverte de chroniques

2.2.1 Comptage de chroniques dans une séquence d'évènements

Afin de pouvoir compter combien d'occurrences une chronique $\mathcal{C}$ peut avoir dans une séquence d'évènements $\mathcal{S}$, il faut définir précisément quelles sont les occurrences qui sont comptées. Avec la séquence $\mathcal{S} = \{(A,1)(A,3)(B,4)\}$, il y a au total deux occurrences de la chronique $\langle AB \rangle$, qui sont $(A,1)(B,4)$ et $(A,3)(B,4)$ (cf. définition 10). Le désavantage de cette mesure est que la fréquence d'une chronique ne représente pas vraiment le nombre d'occurrences qu'elle a dans une séquence selon un utilisateur. Un utilisateur aurait dans de nombreux cas plus tendance dire que dans $\mathcal{S}$ la seule occurrence de $\mathcal{C}$ est $(A,3)(B,4)$, $(A,1)$ étant un évènement de type A n'étant pas relié au processus qu'on cherche à reconnaître dans $\mathcal{S}$.

Dans cet esprit là, l'algorithme *CRS* (Chronicle Recognition System) (Dousson et al. 1993, Dousson et al. 2007), utilisé par (Duong 2001) et que nous utiliserons, ne compte dans une séquence d'évènements que des occurrences ayant des évènements distincts. Autrement dit, *CRS* interdit que le même évènement (dans notre exemple $(B,4)$) apparaisse dans deux occurrences distinctes d'une chronique (dans notre exemple $\mathcal{C}$). Pour ce faire, parmi un ensemble $\mathcal{O}$ (dans notre exemple $\mathcal{O} = \{(A,1)(B,4),(A,3)(B,4)\}$) d'occurrences d'une même chronique ayant des évènements communs, *CRS* ne compte que l'occurrence dite «au plus tôt» de $\mathcal{O}$ (dans notre exemple $(A,1)(B,4)$). Cette description informelle du processus de sélection des occurrences qui sont réellement comptées parmi toutes les occurrences ne suffit pas à spécifier précisément une mesure de fréquence satisfaisante. En effet, le processus décrit ainsi ne compterait que l'occurrence $(A,1)(B,2)$ de $\mathcal{C}$ dans $\mathcal{S} = \{(A,1)(B,2)(A,3)(B,4)\}$, alors qu'on peut potentiellement en compter deux distinctes $(A,1)(B,2)$ et $(A,3)(B,4)$. En réalité l'ensemble des occurrences comptées n'est pas spécifié formellement. Au lieu de cela, les auteurs de (Dousson et al. 1993) spécifient un algorithme de comptage et définissent la fréquence *au plus tôt* comme étant le nombre d'occurrences retourné par cet algorithme (cf. définition 11).

Nous aurons besoin par la suite de cette fréquence «au plus tôt», mais nous aurons également besoin d'une autre mesure de fréquence : la fréquence »au total«.

Définition 10 (Mesure de fréquence *au total*)
La mesure de fréquence au total *d'une chronique $\mathcal{C}$ dans une séquence $\mathcal{S}$ est, en notant $\mathcal{P}(\mathcal{S})$ l'ensemble des sous-séquences de $\mathcal{S}$:*

$$f_t(\mathcal{C},\mathcal{S}) = |\{s | s \in \mathcal{I}_\mathcal{C} \cap \mathcal{P}(\mathcal{S})\}|$$

Définition 11 (Mesure de fréquence *au plus tôt*)
La mesure de fréquence au plus tôt *d'une chronique $\mathcal{C}$ dans une séquence $\mathcal{S}$, notée $f_{au\ plus\ tôt}(\mathcal{C},\mathcal{S})$, est le nombre d'occurrence comptée par la procédure de comptage définie dans (Dousson et al. 1993).*

De ces deux définitions, il découle les deux propriétés suivantes.

Propriété 1
Pour toute chronique $\mathcal{C}$ et toute séquence $\mathcal{S}$, on a : $f_{CRS}(\mathcal{C}, \mathcal{S}) \leq f_t(\mathcal{C}, \mathcal{S})$

Preuve
Soient $\mathcal{C}$ un chronique et $\mathcal{S}$ une séquence. Soit o une occurrence prise en compte par f_{CRS}, on a $o \in \{s | s \in \mathcal{I}_\mathcal{C} \cap \mathcal{S}\}$, donc o est également prise en compte par f_t. Ceci prouve que $f_{CRS}(\mathcal{C}, \mathcal{S}) \leq f_t(\mathcal{C}, \mathcal{S})$.

Propriété 2 (Antimonotonie de la mesure de fréquence au plus tôt)
Soient $\mathcal{C}$ une chronique et $\mathcal{S}$ une séquence d'évènements. Pour toute sous-chronique $\mathcal{C}'$ de $\mathcal{C}$, on a $f_{CRS}(\mathcal{C}, \mathcal{S}) \leq f_{CRS}(\mathcal{C}', \mathcal{S})$.

Preuve
cf. (Duong 2001)

2.2.2 Formulation du problème de découverte

Le problème de découverte de chroniques fréquentes selon (Dousson & Duong 1999, Duong 2001) peut s'énoncer comme suit :

> Étant donnés un domaine temporel $\mathbb{T}$, un ensemble de types d'évènement $\mathbb{E}$, une séquence d'évènements $\mathcal{S} \in \mathbb{S}(\mathbb{E}, \mathbb{T})$, un seuil de fréquence f_{seuil} pour *CRS*, et un critère de sélection de la contrainte temporelle de chaque couple de types d'évènement, trouver toutes les chroniques de $\mathcal{S}$ donc le nombre d'occurrences dans $\mathcal{S}$ selon *CRS* est supérieure à f_{seuil} qui sont composées uniquement des contraintes sélectionnées par le critère de sélection.

Cette formulation n'est pas naturelle, car elle comporte un aspect, le «critère de sélection des contraintes temporelles», qui ne se trouve habituellement pas dans la formulation d'un problème de fouille. Nous verrons que ce critère de sélection permet pourtant de rendre la méthode proposée par (Dousson & Duong 1999, Duong 2001) praticable. La section 2.2.4 explique ce que peut être ce critère de sélection et détaille le processus de génération des chroniques de taille 2 à 1 contrainte.

2.2.3 Méthode Apriori

L'algorithme 1 décrit le procédé utilisé par (Duong 2001) pour résoudre le problème de découverte. Les lignes 1 à 6 constituent l'initialisation avec les chroniques de taille 2 à 1 contrainte. La découverte de $F_{1,0}$ est triviale (il suffit de compter les types d'évènement) ; la découverte de $F_{2,0}$ se fait par génération des épisodes parallèles de taille 2 à partir des types d'évènement fréquents, ce qui est traité par toutes les méthodes de fouille de séquences classiques (Agrawal & Srikant 1995, Pei et al. 2002, Zaki 2001). L'étape de découverte de ces chroniques (cf. ligne 3) est détaillée en section 2.2.4.

Ensuite l'algorithme procède comme suit. Pour tout $n > 2$, il découvre $F_{n,0}$ en générant d'abord l'ensemble $C_{n,0}$ des chroniques candidates de taille n à 0 contraintes à partir de $F_{n-1,0}$. Cela revient à effectuer l'étape de génération des méthodes de fouille de séquences classiques : c'est-à-dire générer les épisodes parallèles de taille n à partir des épisodes parallèles de taille $n-1$. Ensuite, *CRS* compte $C_{n,0}$ pour obtenir $F_{n,0}$ (cf. ligne 9).

Pour $k = 1$ (cf. ligne 10), il génère $C_{n,1}$ en prenant chaque chronique $\mathcal{C}$ de $F_{n,0}$ et pour les C_n^2 couples de types d'évènement (e_i, e_j) de $\mathcal{C}$, il regarde dans $F_{2,1}$ s'il y a une contrainte fréquente pour $e_i e_j$ et l'ajoute dans ce cas à $\mathcal{C}$.

Enfin, pour obtenir $F_{n,k}$ à partir de $F_{n,k-1}$ (lignes 11 à 15), la méthode consiste à générer et compter l'ensemble des candidates $C_{n,k}$ à partir de $F_{n,k-1}$, en couplant toute paire de chroniques $\mathcal{C}_1$ et $\mathcal{C}_2$ de $F_{n,k-1}$ étant basées sur le même épisode et ayant $k-2$ contraintes en commun.

Cette méthode de découverte est qualifiée de *méthode Apriori* car à chaque étape décrite ci-dessus, l'algorithme procède par *génération-comptage*.

Algorithme 1 Découverte de chroniques fréquentes

Entrées: Une séquence d'évènements $\mathcal{S}$; un seuil de fréquence f_{seuil} ; un critère de sélection c.

Sorties: Ensemble F des chroniques fréquentes de $\mathcal{S}$

1: Découvrir $F_{1,0}$
2: Découvrir $F_{2,0}$
3: Découvrir $F_{2,1}$ à partir de c
4: $F_2 \leftarrow F_{2,0} \cup F_{2,1}$
5: $n \leftarrow 2$
6: $F_n \leftarrow F_{2,1}$
7: **tant que** $F_n \neq \emptyset$ **faire**
8: $n \leftarrow n + 1$
9: Découvrir $F_{n,0}$ à partir de $F_{n-1,0}$
10: Découvrir $F_{n,1}$ à partir de $F_{n,0}$ et de $F_{2,1}$
11: **pour** k de 2 à C_n^2 **faire**
12: **si** $F_{n,k-1} \neq \emptyset$ **alors**
13: Découvrir $F_{n,k}$ à partir de $F_{n,k-1}$
14: **fin si**
15: **fin pour**
16: $F_n \leftarrow F_{n,0} \cup F_{n,1} \cup \ldots \cup F_{n,C_n^2}$
17: **fin tant que**
18: $F \leftarrow F_1 \cup F_2 \cup \ldots \cup F_{n-1}$
19: **retourner** F

2.2.4 Génération de chroniques candidates

Génération de chroniques candidates de taille 2 à 1 contrainte

Le processus de découverte de chroniques proposé par (Duong 2001) est basé sur le principe *Apriori* (cf. algorithme 1), c'est-à-dire que les chroniques de taille n à k contraintes temporelles sont crées à partir des chroniques de taille n à $k-1$ contraintes. Pour ce faire, la méthode consiste à «fusionner» deux chroniques fréquentes ayant $k-2$ contraintes en commun comme le montre la figure 2.2.4.

Les questions qui se posent alors sont de pouvoir générer des chroniques de taille n à 0 contrainte, puis de générer des chroniques de taille n à 1 contrainte à partir des chronique de taille n à 0 contrainte. La génération de chroniques de taille n à 0 contrainte n'est pas un problème nouveau. C'est en effet le problème qui a été largement traité par les diverses méthodes de fouilles de séquences (Agrawal & Srikant 1995, Pei et al. 2002, Zaki 2001). Pour cela, la

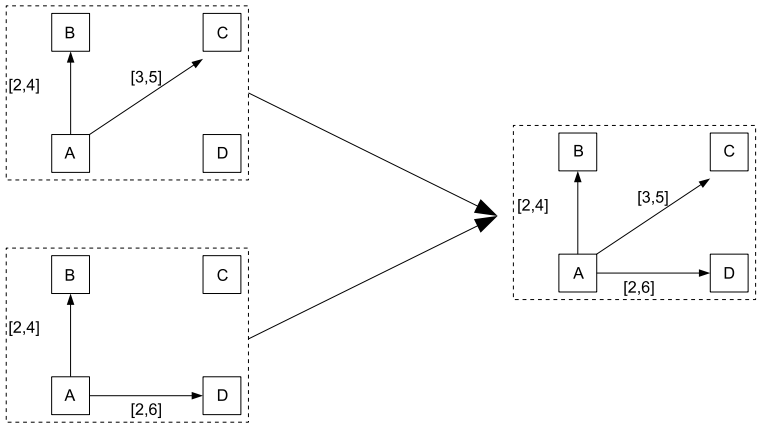


FIG. 2.7 – Génération d'une chronique de taille n à k contraintes à partir de deux chroniques de taille n à $k-1$ contraintes, avec $n=4$ et $k=3$.

Couple	Contraintes possibles	Retenue
AB	$[-4,4]$ $[-1,5]$ $[1,6]$ $[2,7]$ ...	$[-4,6]$
AC	$[-3,2]$ $[0,3]$ $[2,5]$ $[3,8]$	$[-3,2]$
BC	$[-8,-3]$ $[-6,-1]$ $[-4,1]$ $[-3,1]$ $[-1,6]$	$[-4,1]$

TAB. 2.1 – Base de contraintes pour la séquence $\mathcal{C}_0$, avec $f_{seuil}=2$ et $note=0,5$.

phase de génération génère pour chaque couple de séquences de taille $n-1$ ayant le même (n-2)-préfixe (par exemple pour n=5 : `ADGH` et `ADGI`) un candidat de taille n avec le même (n-2)-préfixe (`ADGHI`). Cette méthode a été prouvée être correcte et complète.

Pour résoudre le problème de la génération de chroniques de taille n à 1 contrainte, (Duong 2001) propose à l'étape $n=2$ d'établir une fois pour toute une base de contraintes dans laquelle chaque couple de types d'évènement se voit attribué une et une seule contrainte temporelle (cf. tableau 2.1). Pour calculer cette base de données, (Duong 2001) propose de procéder comme suit. Pour chaque couple de types d'évènement de la séquence d'évènements (prenons le couple (A,C)), il parcourt la séquence pour trouver toutes ses occurrences (pour le couple (A,C), les occurrences dans $\mathcal{C}_0$ sont $(A,1)(C,4)$, $(A,1)(C,9)$, $(A,4)(C,4)$, $(A,4)(C,9)$, $(A,7)(C,4)$, $(A,7)(C,9)$). Pour chaque occurrence, il calcule l'écart temporel entre les deux évènements (ce qui donne les écarts respectifs 3, 8, 0, 5, -3 et 2, soit l'ensemble des écart $\{-3,0,2,3,5,8\}$). Ensuite, une *note* de 0 à 1, passée avec la fréquence seuil f_{seuil} en paramètre d'entrée de l'extraction de chronique, spécifie le taux de couverture que doit avoir chaque contrainte sur la séquence à l'étape 2 (Avec $note=0.5$ et 6 occurrences de (A,C) il faut que la contrainte choisie couvre au moins $0.5\times 6=3$ soit 3 occurrences, ce qui donne pour (A,C) les contraintes temporelles candidates suivantes : $[-3,2]$, $[0,3]$, $[2,5]$, $[3,8]$, $[-3,3]$, $[0,5]$, $[2,8]$, $[-3,5]$, $[0,8]$, $[-3,8]$). Enfin, parmi les contraintes candidates, celle qui a les bornes les plus proches de 0 est retenue (pour (A,C) : $[-3,2]$). En procédant de la sorte pour chaque couple de la séquence $\mathcal{C}_0$, on obtient la base de contraintes du tableau 2.1.

Génération de chroniques candidates de taille n

Une fois cette base de contraintes établie, on peut générer à partir d'une chronique de taille n à 0 contrainte (disons pour $n = 3$, la chronique $\langle ABC \rangle$) C_n^2 chroniques de taille n à 1 contrainte (cf. figure 2.2.4 pour notre exemple) en ajoutant pour chacune une contrainte différente de la base de contraintes.

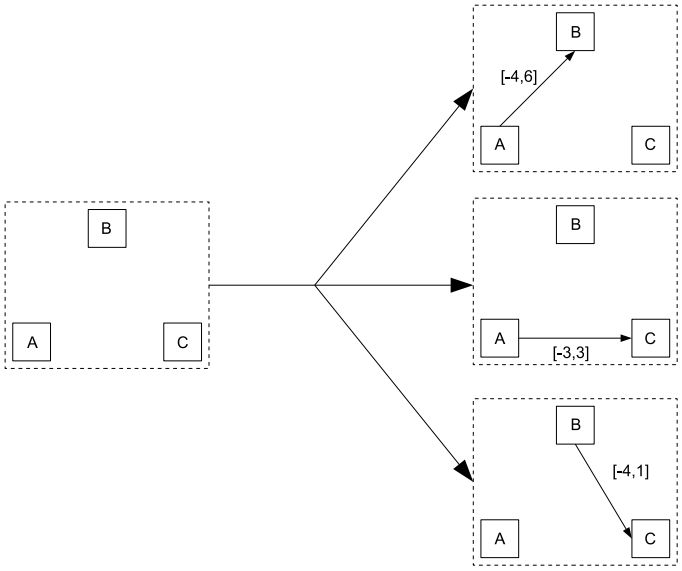


FIG. 2.8 – Génération des chroniques de taille n à 1 contrainte à partir d'une chronique de taille n à 0 contrainte et de la base de contraintes du tableau 2.1, avec $n = 3$.

Chapitre 3

Proposition d'un processus de découverte complète de chroniques

3.1 Limites de l'approche existante et motivations

Cette procédure de choix de la contrainte temporelle d'un couple de types d'évènement présente deux inconvénients majeurs :

1. le choix de la contrainte «la plus représentative», celle qui est retenue, nous paraît trop arbitraire. Avec notre exemple précédent, qu'est-ce qui justifie pour le couple (A, C) et notre séquence $\mathcal{S}_0$ que la contrainte $[-3, 3]$, plutôt que la contrainte $[0, 5]$, permettra de construire des chroniques de tailles supérieures plus souhaitées par l'utilisateur ?
2. pour chaque couple de types d'évènement, une seule contrainte est retenue pour toute la suite de la procédure d'extraction, ce qui signifie que si par exemple c'est la contrainte $[-3, 3]$ qui est retenue, il sera impossible de construire ultérieurement dans la procédure des chroniques autorisant la présence d'un C 5 unités de temps après A, comme le ferait la contrainte $[0, 5]$.

Pour illustrer ces deux inconvénients, prenons deux chroniques de taille n et comportant les types d'évènement A et C. Nous pensons que si l'une comporte la contrainte $[-3, 3]$ entre ces deux types et l'autre la contrainte $[0, 5]$, alors il est important de pouvoir faire la distinction entre ces deux chroniques et d'être potentiellement capable d'extraire chacune d'elle par la suite. La figure 3.1 vient appuyer cette idée. Elle montre trois chroniques de taille 3 qui semblent proches dans leur structures mais ayant deux à deux une seule contrainte qui diffère. Ces trois chroniques ont été construites à partir de 3 bases de contraintes différentes élaborées selon la méthode de (Duong 2001), sur la même séquence $\mathcal{S}_0$ et les mêmes paramètres d'entrée $f_{seuil} = 2$ et $note = 0.5$. Dans chaque cas, seul le choix de la contrainte la plus représentative pour un seul couple de types d'évènement diffère. Les occurrences de ces chroniques dans la séquence $\mathcal{S}_0$ sont

pourtant totalement différentes, ce qui montre l'importance que peut avoir le choix d'une seule contrainte sur le sens portée par la chronique pour l'utilisateur.

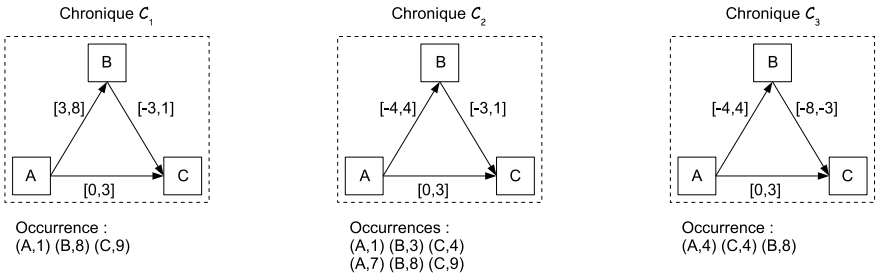


FIG. 3.1 – Trois chroniques de taille 3, générées à partir de 3 bases de contraintes différentes.

Un autre inconvénient de cette approche est qu'une chronique peut être non-fréquente parce qu'une de ces contraintes temporelles, par exemple $[2, 3]$ est trop restrictive, alors que la même chronique avec cette contrainte étant un peu plus élargie, par exemple $[2, 4]$ pourrait être fréquente. De la même manière, pour un couple de types d'évènement (e_i, e_j) d'une chronique fréquente, il est possible qu'en resserrant la contrainte temporelle entre e_i et e_j la chronique soit toujours fréquente. Une telle chronique, fréquente mais plus stricte, est considérée comme plus intéressante pour l'utilisateur puisqu'elle précise la position relation des évènements par rapport à la première.

3.2 Proposition d'une méthode complète de découverte de chroniques

Nous proposons pour remédier à ces inconvénients de réfléchir à une méthode complète d'extraction de chroniques d'une séquence d'évènements.

3.2.1 Utilisation d'une fenêtre pour le parcours de la trace

En préalable, nous devons introduire une notion supplémentaire, qui n'existe pas dans ce que nous avons présenté jusqu'à présent. Le processus de découverte de (Duong 2001) a été pensé pour s'appliquer au domaine de la propagation des alarmes dans les réseaux de télécommunication. Dans ce domaine, tous les évènements d'une séquence d'évènements, ou «journal d'alarmes», proviennent d'une courte période temporelle, si bien qu'il peut être judicieux de corréler à l'aide d'une chronique le premier et le dernier évènement d'un même journal. Dans notre domaine, les séquences d'évènements sont les traces d'interaction. Ces traces d'interaction ont une durée potentiellement infinie et contiennent de très nombreux évènements. Rechercher des corrélations entre des évènements très éloignés peut s'avérer très coûteux, et à quelques rares exceptions près cela s'avère également inutile. En effet, dans une trace d'interactions on s'attend à ce que la plus grande majorité des corrélations s'opère sur des évènements proches temporellement.

Pour toutes ces raisons, nous avons décidé de limiter la recherche de chroniques fréquentes à celles contenues à l'intérieur d'une fenêtre de diamètre win_{max}. Par exemple, si on pose $win_{max} = 10s$, on s'interdit alors de construire des chroniques comme $\langle ABC : A[0s, 8s]B, B[2s, 4s]C, A[2s, 12s]C\rangle$, car les bornes des contraintes temporelles doivent être comprises entre $-win_{max}$ et $+win_{max}$. Évidemment, certaines situations typiques inscrites dans les traces peuvent mettre en jeu des évènements temporellement éloignés, mais nous faisons l'impasse sur ceux-là pour la moment. Le problème de la découverte de motifs temporels sur différentes échelles de temps est un problème difficile pour lequel nous n'avons pas pour l'heure trouvé de méthodes existantes pertinentes dans notre cadre.

3.2.2 Découverte complète de chroniques de taille 2 à 1 contrainte

Processus

Le processus de découverte complète de chronique passe, comme dans (Duong 2001), par la découverte complète des chroniques fréquentes de taille 2. Pour cela, on établit d'abord l'ensemble F_1 des chroniques fréquentes de taille 1. Ensuite on génère l'ensemble $C_{2,0}$ des chroniques candidates de taille 2 à 0 contrainte à partir de F_1 de la même manière que dans (Duong 2001), puis on compte $F_{2,0}$.

Ensuite, pour chaque chronique $\mathcal{C}$ de l'ensemble $F_{2,0}$ on génère un ensemble de chroniques de taille 2 à 1 contrainte d'une manière un peu différente à (Duong 2001), dont les étapes sont données ci-après.

Premièrement, en posant pour chaque chronique fréquente $\mathcal{C} \in F_{2,0}$, $\mathcal{C} = (\mathcal{A}, \mathcal{T})$ où $\mathcal{A} = \{a, b\}$ et $\mathcal{T} = \emptyset$, on détermine l'ensemble des occurrences de $\mathcal{C}$ dans $\mathcal{S}$. Notons $\mathcal{O}_\mathcal{C} = \{(a, t_{a_i})(b, t_{b_i})\}_{i=1}^{n}$ l'ensemble des occurrences de $\mathcal{C}$, où n est le nombre d'occurrences.

Ensuite, on calcule à partir de $\mathcal{O}_\mathcal{C}$ l'ensemble Δ des écarts entre le premier et le deuxième évènement de chaque occurrence de $\mathcal{C}$ dans $\mathcal{S}$, comme dans (Duong 2001). Cela donne une liste $\Delta = (\delta_i)_{i=1}^{n}$, où $\delta_i = t_{b_i} - t_{a_i}$. Supposons cet ensemble ordonné.

Enfin, on génère l'ensemble $\mathcal{K}$ des contraintes temporelles candidates pour la chronique $\mathcal{C}$ en faisant glisser sur la liste Δ une fenêtre de largeur $f_{seuil} - 1$, ce qui génère $n - f_{seuil} + 1$ chroniques, puis en faisant glisser une fenêtre de largeur f_{seuil}, puis $f_{seuil} + 1$, et ainsi de suite jusqu'à une fenêtre de largeur $n - 1$, la plus grande possible, et générant une seule contrainte temporelle candidate. Au total le nombre de contraintes temporelles générées par ce processus est $\sum_{k=n-f_{seuil}+1}^{n-1} k$.

Exemple

Reprenons l'exemple de la séquence $\mathcal{S}_0$ de la figure 2.1, avec une fréquence seuil $f_{seuil} = 2$. Pour cette séquence, on a :

$$F_{2,0} = \{\langle AB\rangle, \langle BC\rangle, \langle AC\rangle, \langle BB\rangle\}.$$

Reprenons l'exemple de la chronique $\langle AC\rangle$ pour effectuer la génération des chro-

niques de taille 2 *à* 1 *contrainte basée sur* `AC`*. On a successivement :*

$$\begin{aligned}\mathcal{O}(A,C) &= \{(A,1)(C,4),(A,1)(C,9),(A,4)(C,4),(A,4)(C,9),\\ &\quad (A,7)(C,4),(A,7)(C,9)\}\\ \textit{et } \Delta(A,C) &= (3,8,0,5,-3,2)\\ &= (-3,0,2,3,5,8)\end{aligned}$$

On construit ensuite $\mathcal{K}$ *en faisant glisser sur* Δ *des fenêtres de largeurs win, en commençant par* $win = f_{seuil} - 1 = 1$*, puis avec* $win = 2, \ldots,$ *puis* $win = n-1$ *:*

$$\begin{aligned}&win = 1 : \{[-3,0],[0,2],[2,3],[3,5],[5,8]\}\\ &win = 2 : \{[-3,2],[0,3],[2,5],[3,8]\}\\ &win = 3 : \{[-3,3],[0,5],[2,8]\}\\ &win = 4 : \{[-3,5],[0,8]\}\\ &win = 5 : \{[-3,8]\}\end{aligned}$$

On obtient donc :

$$\begin{aligned}\mathcal{K}(A,C) = \{&[-3,0],[0,2],[2,3],[3,5],[5,8],[-3,2],[0,3],[2,5],[3,8],[-3,3],\\ &[0,5],[2,8],[-3,5],[0,8],[-3,8]\}\end{aligned}$$

Les chroniques candidates de taille 2 à 1 occurrence basées sur `AC` *dans* $\mathcal{S}_0$ *pour* $f_{seuil} = 2$ *sont donc :*

$$C_{2,1}(A,C) = \{((A,C),[-3,0]),((A,C),[0,2]),\ldots,((A,C),[-3,8])\}$$

On obtient de même $C_{2,1}(A,B)$*,* $C_{2,1}(B,B)$*, et* $C_{2,1}(B,C)$*, ce qui donne finalement :*

$$C_{2,1} = C_{2,1}(A,B) \cup C_{2,1}(A,C) \cup C_{2,1}(B,B) \cup C_{2,1}(B,C)$$

Chaque chronique de l'ensemble $C_{2,1}$ ainsi bâti est ensuite comptée dans la séquence $\mathcal{S}$ pour obtenir $F_{2,1}$. Ce comptage pourrait sembler inutile du fait qu'on ait sélectionné uniquement les contraintes assurant une fréquence supérieure ou égale à f_{seuil}. Il est en fait nécessaire car la fréquence considérée ici est la fréquence *au total* (cf. définition 10), c'est-à-dire qu'on génère ainsi exactement toutes les chroniques $\mathcal{C}$ de taille 2 à 1 contrainte telles que $f_t(\mathcal{C},\mathcal{S}) \geq f_{seuil}$. Ce qui nous intéresse est l'ensemble des chroniques $\mathcal{C}$ telles que $f_{CRS}(\mathcal{C},\mathcal{S}) \geq f_{seuil}$, c'est pourquoi on doit compter $C_{2,1}$. Il est en revanche simple de prouver, à partir de la propriété 1, que ce processus de génération est complet, c'est-à-dire que toute chronique fréquente selon CRS (cf. définition 11) de taille 2 à 1 contrainte, est comprise dans l'ensemble $C_{2,1}$ ainsi généré.

Base de contraintes

Ainsi pour chaque couple de types d'évènements, nous ne disposons plus d'une contrainte, mais un ensemble de plusieurs contraintes temporelles. Des relations d'inclusion existent entre les différentes contraintes temporelles de cet ensemble. Nous définissons ici précisément la structure qui est composée par cet ensemble de contraintes muni de l'inclusion : c'est le graphe d'inclusion (cf. définition 12).

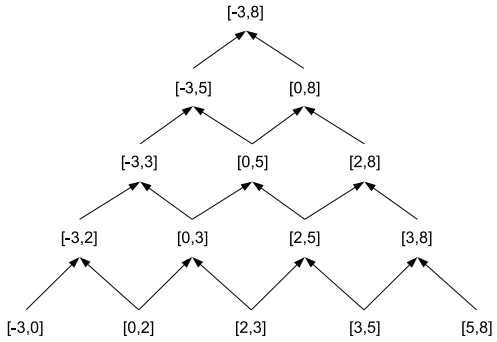


FIG. 3.2 – Graphe d'inclusion de contraintes temporelles.

Définition 12 (Graphe d'inclusion)
Soit $E = \{e_i\}_{i=1}^n$ un ensemble fini muni d'une relation d'ordre partielle $\subseteq_E$ appelée «relation d'inclusion». On dit que $\mathcal{G} = (e_i^1, e_i^2)_{i=1}^p$ est le graphe d'inclusion sur E si :

$$\forall i, (e_i^1, e_i^2) \Leftrightarrow e_i^1 \subseteq_E e_i^2$$

Un exemple de la manière dont les contraintes temporelles générées à cette étape peuvent être organisées en un graphe d'inclusion est donné par la figure 3.2. La figure 3.2 montre le graphe d'inclusion des contraintes temporelles fréquentes pour (A, C) dans $\mathcal{S}$, et issues de l'algorithme présenté dans la section 3.2.2. Ce graphe d'inclusion a des propriétés qu'on peut montrer être vraies pour tout graphe d'inclusion qui sera bâti sur des contraintes issues de cet algorithme. Premièrement, il n'a qu'un seul élément maximal qu'on appellera «racine», ensuite chaque noeud a zéro ou deux fils, puis il est acyclique. La figure 3.7 montre quant à elle un exemple de graphe d'inclusion sur des chroniques.

Le concept de *base de contraintes* dans le cadre de la découverte complète de chroniques se différentie fortement de celui de (Duong 2001), étant donné qu'un couple d'évènement pourra avoir plusieurs contraintes fréquentes associées. La définition de ce nouveaux type de base de contraintes se base sur la notion de graphe d'inclusion (cf. définition 13).

Définition 13 (Base de contraintes)
Soient $\mathbb{T}$ un domaine temporel et $\mathbb{E}$ un ensemble fini de types d'évènement. Une base de contraintes $\mathcal{D}$ est un ensemble de triplets $\{(e_i, f_i, \mathcal{G}_i)\}_{i=1}^n$ tel que :

1. *pour tout i vérifiant $0 \leq i \leq n$, $(e_i, f_i) \in \mathbb{E}^2$, et $\mathcal{G}_i$ est un graphe d'inclusion de contraintes temporelles définies sur $\mathbb{T}$.*
2. *pour tout couple (i, j) vérifiant $0 \leq i < j \leq n$, soit $e_i \prec_{\mathbb{E}} e_j$, soit $e_i = e_j$ et $f_i \prec_{\mathbb{E}} f_j$*

où $\prec_{\mathbb{E}}$ est l'ordre strict sur $\mathbb{E}$.

Le deuxième point de la définition 13 assure le fait qu'un couple de types d'évènement n'apparaisse qu'une seule fois dans la base. La figure 3.3 montre un exemple simple de base de contraintes, où seulement trois couples de types d'évènement (A, B),(A, C) et (B, C) sont fréquents, et où chaque couple de types

d'évènement ne contient que 3 contraintes fréquentes. Dans la suite, les graphes d'inclusion $\mathcal{G}_i$ seront les graphes d'inclusion construits à partir des contraintes temporelles fréquentes dans $\mathcal{S}$ du couple de types d'évènement (e_i, f_i) issues de l'algorithme présenté dans la section 3.2.2 pour la fréquence seuil f_{seuil}.

La base de contrainte construite par le processus décrit dans cette section vérifie la propriété 3, qui sera utile pour la suite.

Propriété 3
Dans une base de contrainte construite par le processus décrit dans la section 3.2.2, chaque graphe de contraintes $\mathcal{G}_{e_i f_i}$ possède un unique élément, noté $\mathcal{G}^{\top}_{e_i f_i}$, tel que :

$$\forall \tau \in \mathcal{G}_{e_i f_i}, \tau \subseteq \mathcal{G}^{\top}_{e_i f_i}$$

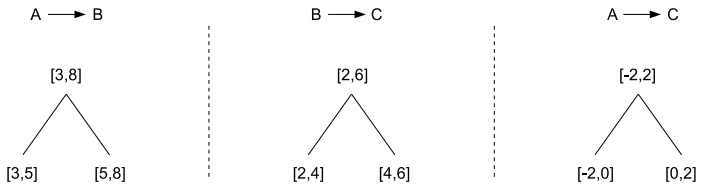


FIG. 3.3 – Exemple de base de contraintes $\mathcal{D}_0$.

Le théorème 1 servira également par la suite.

Théorème 1
Soient $(a, b) \in \mathbb{E}^2$ et (i, j) vérifiant $-win_{max} \leq i < j \leq win_{max}$ tels que $a[i, j]b$ soit une contrainte fréquente dans $\mathcal{S}$. Il existe alors (i', j') vérifiant $[i', j'] \subseteq [i, j]$ et $a[i', j']b \in \mathcal{D}(\mathcal{S}, f_{seuil})$, et tel que $\mathcal{O}_{a[i,j]b} = \mathcal{O}_{a[i',j']b}$.

Preuve
Notons $\mathcal{O}_{a[i,j]b} = \{(a, t_{a1})(b, t_{b1}), \ldots, (a, t_{ap})(b, t_{bp})\}$. Soit Λ l'ensemble des écarts $t_b - t_a$ de $\mathcal{O}_{a[i,j]b}$:

$$\Lambda = (t_{b1} - t_{a1}, \ldots, t_{bp} - t_{ap}) = (\delta_1, \ldots, \delta_p)$$

Soit $i' = min(\Lambda)$ et $j' = max(\Lambda)$. On a alors $[i', j'] \subseteq [i, j]$, car sinon un des éléments de Λ ne respecterait pas la contrainte $t_b - t_a \in [i, j]$. De plus $a[i', j']b \in \mathcal{D}(\mathcal{S}, f_{seuil})$. En effet, en suivant la procédure de construction des ensembles Δ de la section 3.2.2 appliquée à (a, b), on obtient l'ensemble $\Delta(a, b)$ de toutes les occurrences de $a[-\infty_{\mathbb{T}}, \infty_{\mathbb{T}}]b$ dans $\mathcal{S}$, donc $\Lambda \subseteq \Delta(a, b)$. En conséquence $[i', j']$ est construite à partir d'éléments de $\Delta(a, b)$ tels que $a[i', j']b$ soit fréquente, c'est donc que $a[i', j']b \in \mathcal{D}(\mathcal{S}, f_{seuil})$.

Enfin on a $\mathcal{O}_{a[i,j]b} = \mathcal{O}_{a[i',j']b}$. Pour démontrer cela, il faut reprendre l'algorithme CRS *de comptage des chroniques fréquentes au plus tôt. Ainsi, l'occurrence $(a, t_{a1})(b, t_{b1})$ est la 1^{re} occurrence de $a[i, j]b$ reconnue par* CRS *si et seulement si elle est la première occurrence de $a[i', j']b$ selon* CRS*. Pour cela il faut revenir à la procédure de comptage* CRS *telle que définie dans (Dousson et al. 1993). On procède de même pour la k^e occurrence de $a[i, j]b$ dans $\mathcal{S}$ jusqu'à p, et de même pour dire que $a[i, j]b$ et $a[i', j']b$ n'ont pas de $(p + 1)^e$ occurrence. On démontre ainsi que $\mathcal{O}_{a[i,j]b} = \mathcal{O}_{a[i',j']b}$.*□

Le théorème 1 peut être interprété en disant que la base de contraintes $\mathcal{D}$ contient à une inclusion prêt toutes les contraintes fréquentes dans $\mathcal{S}$. $\mathcal{D}$ a en réalité une autre propriété encore plus précise que celle-ci, mais que nous n'utiliserons pas : une telle contrainte $a[i', j']b$ est unique dans $\mathcal{D}$.

3.2.3 Découverte complète de chroniques de taille $n > 2$

Définitions

Supposons qu'étant donnés une séquence $\mathcal{S} \in \mathbb{S}(\mathbb{E}, \mathbb{T})$ et une fréquence seuil f_{seuil}, on ait l'ensemble $F_{2,1}$ des chroniques fréquentes de taille 2 à 1 contrainte. D'après la propriété d'antimonotonie (propriété 2), toute chronique fréquente de taille supérieure à 2 sera construite à partir des contraintes de la base de contraintes. On définit donc formellement la notion de *chronique construite à partir d'une base de contraintes.*

Définition 14 (Chronique construite depuis une base de contraintes) *On dit qu'une chronique $\mathcal{C}$ est construite à partir d'une base de contraintes $\mathcal{D}$ si toutes ses contraintes τ_{ij} appartiennent à $\mathcal{D}$. On note $C_{\mathcal{D}}$ l'ensemble des chroniques construites à partir de $\mathcal{D}$.*

FIG. 3.4 – Deux chroniques, l'une ($\mathcal{C}_1$) construite à partir de la base de contrainte de la figure 3.3, l'autre ($\mathcal{C}_2$) n'est pas construite à partir de cette base de contraintes, car $[1, 4]$ n'est pas une contrainte appartenant au couple du type d'évènement (A, B).

Comment découvrir toutes les chroniques fréquentes de taille supérieure à 2 ? L'idée est toujours d'appliquer le principe *Apriori*, c'est-à-dire de générer à chaque étape du processus, des chroniques candidates toujours plus contraintes. On définit pour cela la notion de chronique *directement plus contrainte* (cf. définition 15, figure 3.5 pour l'exemple).

Définition 15 (Chronique directement plus contrainte) *Soient $\mathcal{S} \in S(\mathbb{E}, \mathbb{T})$ une séquence d'évènements, f_{seuil} une fréquence seuil, $\mathcal{D}(\mathcal{S}, f_{seuil})$ la base de contraintes obtenue par l'algorithme présenté dans la section 3.2.2 appliqué à $\mathcal{S}$, et $\mathcal{C}$ et $\mathcal{C}'$ deux chroniques construites à partir de $\mathcal{D}(\mathcal{S}, f_{seuil})$. On dit que $\mathcal{C}'$ est* directement plus contrainte relativement à *$\mathcal{D}(\mathcal{S}, f_{seuil})$ que $\mathcal{C}$ si et seulement si :*

1. *$\mathcal{C}' \prec \mathcal{C}$ ($\mathcal{C}'$ est strictement plus contrainte que $\mathcal{C}$)*
2. *il n'existe pas de chronique $\mathcal{C}''$ construite à partir de $\mathcal{D}(\mathcal{S}, f_{seuil})$ et vérifiant $\mathcal{C}' \prec \mathcal{C}'' \prec \mathcal{C}$*

Pour dire que $\mathcal{C}'$ est directement plus contrainte que $\mathcal{C}$ par rapport à une base de contraintes $\mathcal{D}(\mathcal{S}, f_{seuil})$, on écrit $\mathcal{C}' \in Succ_{\mathcal{D}(\mathcal{S}, f_{seuil})}(\mathcal{C}_C)$.

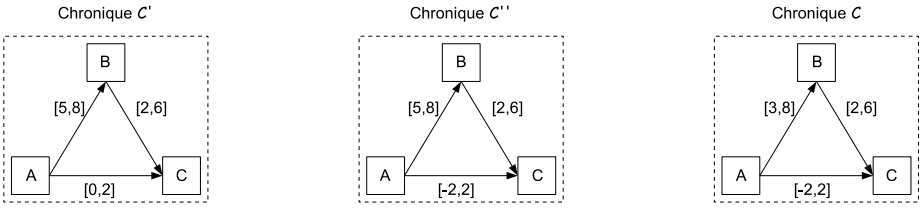


FIG. 3.5 – Trois chroniques. $\mathcal{C}'$ n'est pas directement plus contrainte que $\mathcal{C}$ relativement à la base de contraintes de la figure 3.3, car $\mathcal{C}' \prec \mathcal{C}'' \prec \mathcal{C}$. En revanche, $\mathcal{C}'$ est directement plus contrainte que $\mathcal{C}''$ et $\mathcal{C}''$ est directement plus contrainte $\mathcal{C}'$.

3.2.4 Opérateurs de génération de chroniques directement plus contraintes

Étant donnée une chronique $\mathcal{C}$ de taille n à k contraintes, il existe deux types d'opérateurs permettant d'obtenir une chronique $\mathcal{C}'$ qui soit *directement plus contrainte* que $\mathcal{C}$. Ces deux types d'opérateur sont :

1. l'ajout d'un nouveau type d'évènement à $\mathcal{C}$,
2. la restriction d'une des contraintes temporelles de $\mathcal{C}$.

Le premier type d'opérateur correspond à ce que (Duong 2001) désigne par la génération de chroniques de taille $n+1$ à k contraintes à partir de chroniques de taille n à k contraintes. Le deuxième type d'opérateur correspond à la génération de chroniques de taille n à $k+1$ contraintes à partir de chroniques de taille n à k contraintes.

La figure 3.6 montre un exemple d'application de ces deux types d'opérateurs dans le cas de base de contraintes $\mathcal{D}_0$ et pour la chronique de taille 2 $\langle AB : A[3,8]B\rangle$. Il existe dans $\mathcal{D}_0$ deux contraintes plus strictes que $[3,8]$ pour le couple (A,B). On dispose donc de deux opérateurs de type 2, un pour chaque contrainte plus stricte, ce qui donne deux chroniques de taille 2 directement plus strictes de taille 2. Par ajout d'un nouveau type d'évènement à $\mathcal{C}$ (opérateur de type 1) on obtient 3 nouvelles chroniques directement plus strictes de taille 3. Cependant, parmi ces trois nouvelles chroniques, deux ne peuvent pas être fréquentes. En effet, la chronique basée sur `AAB` (resp. la chronique basée sur `ABB`) ne peut pas être fréquente car elle est plus stricte que la chronique de taille 2 basée sur `AA` (resp. basée sur `BB`) qui n'est pas dans $\mathcal{D}_0$, c'est-à-dire qui n'est pas fréquente. Ces chroniques directement plus fréquentes dont on sait qu'elles ne peuvent pas être fréquentes sur sont barrées sur la figure 3.6. On obtient ainsi trois chroniques directement plus strictes à l'état (2). En procédant de même pour la première chronique trouver, on obtient une nouvelle chronique directement plus contrainte (cf. état (3)), puis quatre à l'état (4), et ainsi de suite.

Une fois ces deux opérateurs définis, il est possible de voir le graphe d'inclusion de toutes les chroniques comme un graphe d'états à parcourir. La figure 3.7 montre le graphe d'inclusion de chroniques complet résultant de l'application systématique de l'opérateur n° 2 à une chronique de taille 3 pour la base de contraintes $\mathcal{D}_0$.

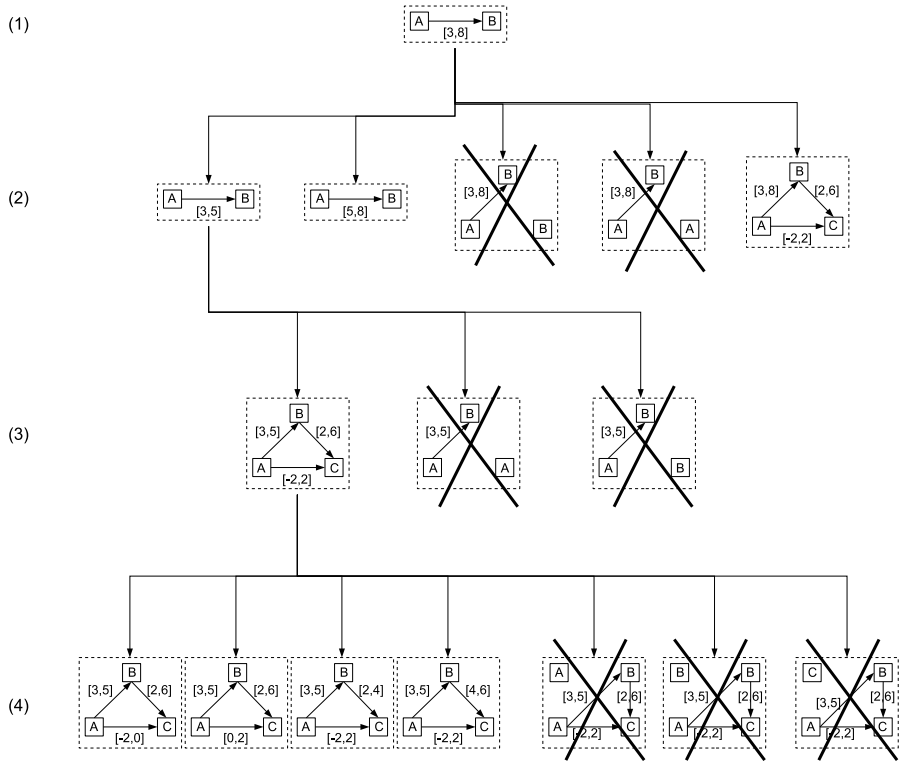


FIG. 3.6 – Génération de chroniques plus strictes par application des opérateurs.

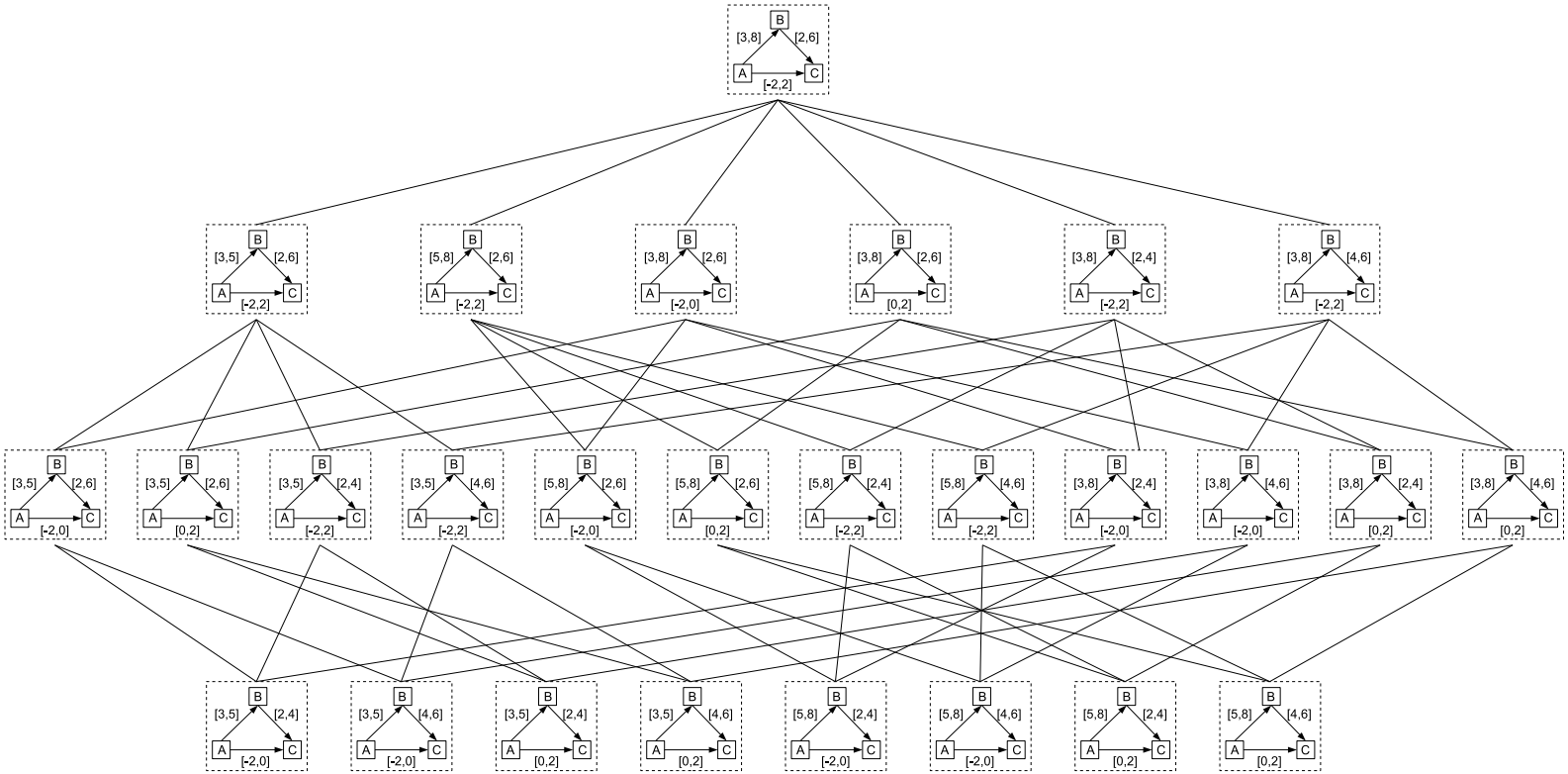


FIG. 3.7 – Graphe d'inclusion de chroniques de taille 3.

Définition 16 (Chroniques équivalentes dans une séquence)
Soit $\mathcal{S} \in S(\mathbb{E}, \mathbb{T})$, deux chroniques $\mathcal{C}_1$ et $\mathcal{C}_2$ sont dites équivalentes dans $\mathcal{S}$ *si et seulement si $\mathcal{O}_{\mathcal{C}_1} = \mathcal{O}_{\mathcal{C}_2}$, c'est-à-dire si elles ont les mêmes occurrences. On note alors $\mathcal{C}_1 \sim_{\mathcal{S}} \mathcal{C}_2$. Formellement :*

$$\mathcal{C}_1 \sim_{\mathcal{S}} \mathcal{C}_2 \Leftrightarrow \mathcal{O}_{\mathcal{C}_1} = \mathcal{O}_{\mathcal{C}_2}$$

Théorème 2
Soient une séquence $\mathcal{S} \in \mathbb{S}(\mathbb{E}, \mathbb{T})$, une fréquence seuil f_{seuil}, $\mathcal{C}$ une chronique fréquente dans $\mathcal{S}$, et $\mathcal{D}(\mathcal{S}, f_{seuil})$ la base de contraintes de $\mathcal{S}$ pour f_{seuil}. Il existe une chronique $\mathcal{C}'$ construite à partir de $\mathcal{D}(\mathcal{S}, f_{seuil})$, telle que :

1. $\mathcal{C}' \sim_{\mathcal{S}} \mathcal{C}$
2. $\mathcal{C}' \preceq \mathcal{C}$

Preuve
L'idée est de construire à partir de $\mathcal{C}$ la chronique $\mathcal{C}'$ ayant les mêmes types d'évènements, mais où chaque contrainte τ'_{ij} de $\mathcal{C}'$ est une contrainte maximale de l'ensemble $\tau' | \tau' \in \mathcal{D}(\mathcal{S}, f_{seuil}) \wedge \tau' \subseteq \tau_{ij} \wedge \mathcal{O}_{\tau'} = \mathcal{O}_{\tau_{ij}}$ (l'existence est assurée par le théorème 1).

On a alors soit $\mathcal{C} = \mathcal{C}'$, auquel cas $\mathcal{C}$ est déjà elle-même construite à partir de $\mathcal{D}(\mathcal{S}, f_{seuil})$, soit au contraire $\mathcal{C} \neq \mathcal{C}'$, auquel cas il existe (i, j) avec $1 \leq i < j \leq |\mathcal{C}|$ tel que $\tau_{ij} \neq \tau'_{ij}$. Dans ce cas $\tau_{ij}^- < \tau_{ij}^{-\prime}$ ou $\tau_{ij}^+ > \tau_{ij}^{+\prime}$. Supposons qu'on soit dans le cas $\tau_{ij}^- \neq \tau_{ij}^{-\prime}$. Quelque soit l'occurrence $(e_i, t_i)(e_j, t_j)$ de $\mathcal{S}$, $\mathcal{C}$ et $\mathcal{C}'$ seront purement équivalentes pour l'algorithme de comptage CRS*, car il n'existe pas de valeurs d'écart possible δ pour e_i et e_j tel que $\delta \in [\tau_{ij}^-, \tau_{ij}^+] - [\tau_{ij}^{-\prime}, \tau_{ij}^{+\prime}]$. En effet si tel était le cas, il existerait au moins une contrainte de $\mathcal{D}(\mathcal{S}, f_{seuil})$ ayant pour borne δ (car $[\tau_{ij}^{-\prime}, \tau_{ij}^{+\prime}] \in \mathcal{D}(\mathcal{S}, f_{seuil})$, donc d'après le processus de fabrication de $\mathcal{D}(\mathcal{S}, f_{seuil})$, toute contrainte plus générale que $[\tau_{ij}^{-\prime}, \tau_{ij}^{+\prime}]$ et construite à partir d'écarts entre e_i et e_j dans $\mathcal{S}$ est dans $\mathcal{D}(\mathcal{S}, f_{seuil})$) et donc τ'_{ij} ne serait plus chronique maximale de l'ensemble ci-dessus, car la contrainte $[\delta, \tau_{ij}^+]$ serait dans cet ensemble.* □

Ce théorème aide à formuler notre problème de découverte complète en permettant de se limiter à la découverte de chroniques construites à partir de la base de contraintes.

3.2.5 Formulation du problème de découverte complète de chronique

Étant donnés :
- un ensemble fini de types d'évènement $\mathbb{E}$,
- un domaine temporel totalement ordonné $\mathbb{T}$,
- une séquence d'évènements $\mathcal{S} \in \mathbb{S}(\mathbb{E}, \mathbb{T})$,
- une fréquence seuil f_{seuil},
- un diamètre de fenêtre temporelle win_{max},
- $\mathcal{D}(\mathcal{S}, f_{seuil})$ la base de contraintes construite par l'algorithme présenté dans la section 3.2.2,

le problème de découverte de chroniques consiste à extraire toutes les chroniques minimales $\mathcal{C}$ construites à partir de $\mathcal{D}(\mathcal{S}, f_{seuil})$ telles que :

1. $f_{CRS}(\mathcal{C}) \geq f_{seuil}$,
2. toutes les contraintes temporelles de $\mathcal{C}$ sont incluses dans l'intervalle :

$$[-win_{max}, +win_{max}]$$

3.2.6 Proposition d'algorithme

Détail des procédures appelées :

- `PRENDRE_PREMIER(OUVERTES)` : choisit et retire de l'ensemble `OUVERTES` la chronique qui y figure en tête. Cela présuppose qu'on dispose d'une fonction de classement sur les chroniques qui indique laquelle choisir en premier. Cette fonction, appelée heuristique, influencera le nombre d'itérations nécessaires avant de trouver de «bonnes» chroniques fréquentes.
- `CONTIENT_PLUS_STRICTE(FRÉQUENTES,`$\mathcal{C}$`)` : renvoie vrai s'il existe dans l'ensemble `FRÉQUENTES`, une chronique $\mathcal{C}'$ telle que $\mathcal{C}' \preceq \mathcal{C}$. L'appel à cette méthode à la ligne 13 permet d'éviter de compter $\mathcal{C}$ dans $\mathcal{S}$ alors qu'on sait qu'une de ses sous-chronique est déjà fréquente.
- `AJOUTER_MINIMALE(`$\mathcal{C}$`,FRÉQUENTES)` ajoute la chronique $\mathcal{C}$ à `FRÉQUENTES` et supprime de `FRÉQUENTES` toute chronique $\mathcal{C}'$ qui devient alors non minimale. Lorsqu'on appelle cette procédure, on sait déjà que $\mathcal{C}$ est minimal dans `FRÉQUENTES`.
- `AJOUTER_MAXIMALE(`$\mathcal{C}$`,NON_FRÉQUENTES)` : ajoute de la même manière $\mathcal{C}$ à `NON_FRÉQUENTES` en supprimant toutes celles qui deviennent non maximales. On sait également que $\mathcal{C}$ est déjà maximale.
- `GÉNÉRER_SUCCESSEURS(`$(\mathcal{C})$`)` : génère tous les successeurs de $\mathcal{C}$ selon les deux types opérateurs définis en section 3.2.4.
- `EST_CONSISTANTE(`$\mathcal{C}'$`)` : renvoie `faux` si $\mathcal{C}'$ n'est pas pas consistante, `vrai` sinon.

3.3 Analyse de l'approche proposée

3.3.1 Complétude et terminaison

Complétude La complétude de l'algorithme 2 est assurée par le fait que l'ensemble des chroniques construites à partir de $\mathcal{D}(\mathcal{S}, f_{seuil})$ est bien parcouru par la méthode qui consiste à générer tout les successeurs d'une chronique et en initialisant le processus avec les chroniques $\mathcal{G}^{\top}_{e_i e_j}$ (où $(e_i, e_j) \in \mathbb{E}^2$), c'est à dire avec les chroniques de taille 2 ayant la contrainte la plus générale de $\mathcal{D}(\mathcal{S}, f_{seuil})$. Ceci est vrai car quelle que soit la chronique $\mathcal{C} = (\mathcal{A}, \mathcal{T})$ construite à partir de $\mathcal{D}(\mathcal{S}, f_{seuil})$, on a trivialement $\mathcal{C} \preceq \mathcal{G}^{\top}_{e_i e_j}$ pour tout $(e_i, e_j) \in \mathcal{A}^2$ vérifiant $e_i \neq e_j$. De plus, la propriété d'antimonotonicité de «$\preceq$» pour f_{CRS} permet de s'assurer que les tests de minimalité (lignes 13 et 25) et de maximalité (lignes 10 et 26) n'écartent aucune chronique fréquente. On a de plus le résultat intéressant suivant : il n'y a pas besoin de propager les chronique générées car elles sont nécessairement déjà propagées (cf. théorème 3).

Théorème 3
Toute chronique fréquente issue de l'algorithme 2 est déjà propagée.

Algorithme 2 Algorithme de découverte complète de chroniques fréquentes

Entrées: $\mathbb{E}$; $\mathbb{T}$; $\mathcal{S} \in \mathbb{S}(\mathbb{E}, \mathbb{T})$; f_{seuil} ; $\mathcal{D}(\mathcal{S}, f_{seuil})$
Sorties: Ensemble FRÉQUENTES des chroniques fréquentes minimales dans $\mathcal{S}$

1: FRÉQUENTES$\leftarrow \emptyset$
2: OUVERTES $\leftarrow \emptyset$
3: **pour** chaque $(e_i^1, e_i^2, \mathcal{G}_i) \in \mathcal{D}(\mathcal{S}, f_{seuil})$ **faire**
4: $\quad \mathcal{C} \leftarrow ((e_i^1, e_i^2), \mathcal{G}_i^\top)$
5: $\quad$ Ajouter $\mathcal{C}$ à FRÉQUENTES
6: $\quad$ OUVERTES $\leftarrow$ OUVERTES $\cup \{\mathcal{C}\}$
7: **fin pour**
8: **répéter**
9: $\quad \mathcal{C} \leftarrow$ PRENDRE_PREMIER(OUVERTES)
10: $\quad$ **si** CONTIENT_PLUS_GÉNÉRAL(NON_FRÉQUENTES,$\mathcal{C}$) **alors**
11: $\quad\quad$ fin d'itération, reprendre la prochaine itération à la ligne 8
12: $\quad$ **fin si**
13: $\quad$ **si** CONTIENT_PLUS_STRICTE(FRÉQUENTES,$\mathcal{C}$) est faux **alors**
14: $\quad\quad f_{CRS}(\mathcal{C}) \leftarrow$COMPTER($\mathcal{C}, \mathcal{S}$)
15: $\quad\quad$ **si** $f_{CRS}(\mathcal{C}) \geq f_{seuil}$ **alors**
16: $\quad\quad\quad$ AJOUTER_MINIMALE($\mathcal{C}$,FRÉQUENTES)
17: $\quad\quad$ **sinon**
18: $\quad\quad\quad$ AJOUTER_MAXIMALE($\mathcal{C}$,NON_FRÉQUENTES)
19: $\quad\quad\quad$ fin d'itération, reprendre la prochaine itération à la ligne 8
20: $\quad\quad$ **fin si**
21: $\quad$ **sinon**
22: $\quad\quad Succ(\mathcal{C}) \leftarrow$GÉNÉRER_SUCCESSEURS($\mathcal{C}$)
23: $\quad\quad$ **pour** chaque $\mathcal{C}' \in Succ(\mathcal{C})$ **faire**
24: $\quad\quad\quad$ **si** EST_CONSISTANTE($\mathcal{C}'$) **alors**
25: $\quad\quad\quad\quad$ **si** CONTIENT_PLUS_STRICTE(FRÉQUENTES,$\mathcal{C}$) est faux **alors**
26: $\quad\quad\quad\quad\quad$ **si** CONTIENT_PLUS_GÉNÉRAL(NON_FRÉQUENTES,$\mathcal{C}$) est faux **alors**
27: $\quad\quad\quad\quad\quad\quad$ {N'ajouter $\mathcal{C}'$ à OUVERTES que si $\mathcal{C}'$ apporte potentiellement une information nouvelle.}
28: $\quad\quad\quad\quad\quad\quad$ OUVERTES $\leftarrow$ OUVERTES $\cup \{\mathcal{C}'\}$
29: $\quad\quad\quad\quad\quad$ **fin si**
30: $\quad\quad\quad\quad$ **fin si**
31: $\quad\quad\quad$ **fin si**
32: $\quad\quad$ **fin pour**
33: $\quad$ **fin si**
34: **jusqu'à** OUVERTES $\neq \emptyset$
35: **retourner** FRÉQUENTES

Preuve
Ce théorème est vrai car l'algorithme 2 ne retourne que des chroniques minimales. Supposons donc une chronique $\mathcal{C}$ fréquente minimale dans $\mathcal{S}$ qui soit pourtant non propagée. Soit donc $\mathcal{C}' = prop(\mathcal{C}) \neq \mathcal{C}$, on sait que $\mathcal{C}' \prec \mathcal{C}$ (inclusion stricte car $\mathcal{C}$ n'est pas propagée). Comme on a d'une part $\mathcal{C} \sim_{\mathcal{S}} \mathcal{C}'$ (deux chroniques équivalentes sont par définition a fortiori *équivalentes dans $\mathcal{S}$) et d'autre part $\mathcal{C}'$ construite à partir de $\mathcal{D}(\mathcal{S}, f_{seuil})$ (sinon il existe d'après le théorème 2 $\mathcal{C}'' \in \mathcal{D}(\mathcal{S}, f_{seuil})$ telle que $\mathcal{C}'' \prec \mathcal{C}'$ et $\mathcal{C}' \sim_{\mathcal{S}} \mathcal{C}''$ et on poursuit alors le raisonnement avec $\mathcal{C}''$), on en déduit que $\mathcal{C}'$ est également fréquente et donc retournée par l'algorithme 2. Ce n'est pas possible car sinon $\mathcal{C}$ ne serait plus minimale, car $\mathcal{C} \prec \mathcal{C}'$.*□

Terminaison La terminaison de l'algorithme 2 est assurée par le raisonnement qui suit. Soit n_{max} la taille de la chronique la plus grande parmi toutes les chroniques fréquentes minimales de $\mathcal{S}$ construites à partir de $\mathcal{D}(\mathcal{S}, f_{seuil})$. On est alors certain que `GÉNÉRER_SUCCESSEURS`, qui ne s'applique qu'à des chroniques fréquentes, ne générera jamais de chroniques de taille $n_{max} + 2$, car alors c'est qu'une chronique de taille $n_{max} + 1$ serait fréquente. Soit N le nombre total de chroniques (fréquentes et non-fréquentes) construites à partir de $\mathcal{D}(\mathcal{S}, f_{seuil})$. Comme les tests des lignes 25 et 26 assurent qu'on ne mettra jamais deux fois la même chronique dans l'ensemble `OUVERTES`, on est certain qu'à la i^{e} itération de la ligne 8, il reste $N - i$ chroniques potentielles à traiter, donc que l'algorithme 2 aura au pire encore $N - i$ itérations à faire. Le paramètre $N - i$ est donc un paramètre décroissant de l'algorithme, ce qui prouve la terminaison.

3.3.2 Amélioration de l'accès aux ensembles `FRÉQUENTES` et `NON_FRÉQUENTES`

À l'exécution, les ensembles `FRÉQUENTES` et `NON_FRÉQUENTES` font l'objet de grands nombres d'opérations de recherche et d'insertion. C'est pourquoi il est important de gérer ces opérations de manière efficace.

La manière naïve de gérer ces deux ensembles, prenons par exemple l'ensemble `NON_FRÉQUENTES`, consiste à déclarer qu'ils ne sont pas structurés. Dans ce cas `NON_FRÉQUENTES` est une liste non ordonnées de chroniques et le test de présence d'une chronique maximal `CONTIENT_PLUS_GÉNÉRAL` s'effectuera en comparant chaque chronique à la chronique d'entrée. La complexité de ce processus est donc en $O(n)$, où n est la taille de l'ensemble `NON_FRÉQUENTES`. De même, la procédure `AJOUTER_MAXIMALE` parcourra chaque chronique de l'ensemble et testera si elle est plus stricte ou non que la chronique d'entrée, puis ajoutera la chronique d'entrée à l'ensemble. Cette procédure naïve s'effectue également en $O(n)$.

Pour l'ensemble `NON_FRÉQUENTES` non n'avons pas trouvé de stratégie permet de réduire de manière significative les coûts des opérations de test et d'insertion. En revanche, nous proposons d'organiser l'ensemble `FRÉQUENTES` en un arbre de séquences, dans lequel chaque noeud représente une séquence sur les types d'évènements et contient l'ensemble des chroniques basées sur cette séquence. La figure 3.8, montre comment est structuré cet ensemble.

Avec cette structure de données, la procédure `CONTIENT_PLUS_STRICTE` appliquée à `FRÉQUENTES` s'effectue comme suit. En notant $\mathcal{C} = (\mathcal{A}, \mathcal{T})$, soit $a_1 \ldots a_n$ l'épisode sur lequel est basée $\mathcal{C}$. On entre alors dans l'arbre de séquences par le

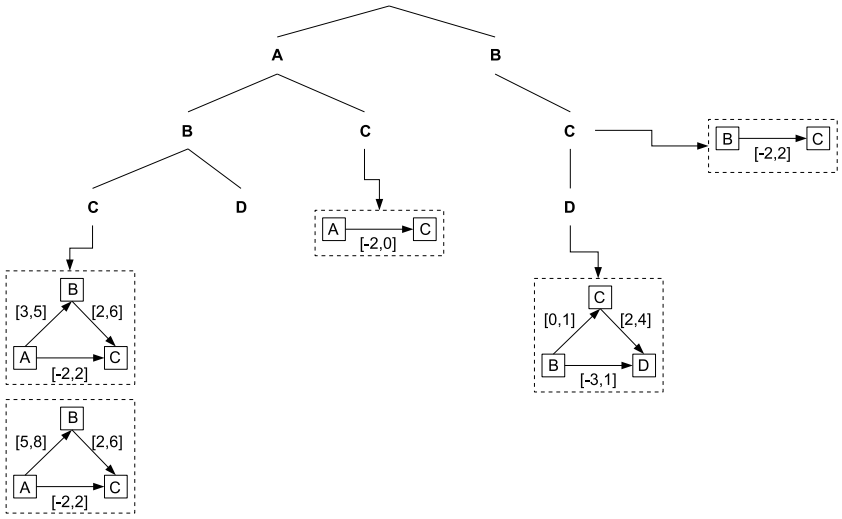


FIG. 3.8 – Structure d'arbre de séquence pour l'ensemble FRÉQUENTES.

noeud a_1. S'il n'y a pas de noeud a_1, c'est qu'il n'y a pas dans FRÉQUENTES de chroniques ayant le type d'évènement a_1. Sinon, on parcourt les enfants de a_1 à la recherche de a_2, s'il ne le trouve pas alors $\mathcal{C}$ est minimale, et ainsi de suite jusqu'au noeud a_n. Si a_n a des enfants, alors $\mathcal{C}$ n'est pas minimale, sinon il faut appliquer le test de généralité à chaque chronique $\mathcal{C}'$ basée sur $a_1 \ldots a_n$. Si on trouve parmi ces chroniques une chronique $\mathcal{C}'$ telle que $\mathcal{C} \preceq \mathcal{C}'$, alors $\mathcal{C}$ n'est pas minimale, $\mathcal{C}$ est minimale sinon.

La complexité au pire de ce processus, c'est-à-dire dans le cas où tous les éléments de FRÉQUENTES sont dans le même noeud, est de n comparaisons, où n est le nombre d'éléments dans FRÉQUENTES, auquel s'ajoute le temps de recherche du noeud représentant la séquence sur laquelle est basée la chronique. Dans le cas moyen, et dans ce qu'on observe dans la pratique, les chroniques sont équiréparties dans les noeuds, ce qui implique qu'il y a au plus deux voire trois chroniques dans le noeud. Dans ce cas, la complexité dépend uniquement du temps de recherche dans l'arbre des types d'évènement, et donc du nombre de types d'évènements $|\mathbb{E}|$. Pour calculer cette complexité au pire des cas, qui est loin de ce qu'on observe dans la pratique, on considère que l'arbre est complet, c'est-à-dire que chaque noeud possède $|\mathbb{E}|$ fils. Par la relation d'ordre totale $\preceq_{\mathbb{E}}$ il est possible d'organiser l'ensemble des fils d'un noeud dans un arbre binaire de recherche, et donc de rechercher le prochain fils en $O(ln|\mathbb{E}|)$. Pour une chronique dont l'épisode est de taille l, la recherche s'effectuera donc en $O(ln|\mathbb{E}|^l)$. La complexité est la même pour l'insertion, même si en pratique l'insertion peut prendre plus de temps, car il faut vérifier dans les noeuds parents qu'il n'y a pas de chroniques qui sont devenues non-minimales.

Avec cette stratégie, les temps de recherche de chroniques minimales et d'insertion sont bons, mais l'ensemble n'est pas toujours minimal. En effet, reprenons l'exemple de l'ensemble FRÉQUENTES de la figure 3.8. Imaginons que dans cet ensemble la chronique fréquente $\langle BC : B[1,6]C\rangle$ ait été ajoutée en dernier. Cette chronique n'est pas minimale car les deux chroniques basées sur ABC sont plus strictes. Cependant les méthodes de recherche et d'insertion exposées ci-

dessus n'ont pas été capables de le détecter, car les chroniques basées sur `ABC` n'appartiennent pas à la même branche que $\langle BC : B[1,6]C\rangle$. En conséquence, notre ensemble `FRÉQUENTES`, n'est pas constitué que de chroniques minimales, mais plutôt de chroniques «minimales dans une même branche». Pour ne trouver que les chroniques minimales, il faudra effectuer un post-traitement au processus de fouille qui supprime de cet ensemble toutes les chroniques qui sont non-minimales. Si on voulait maintenir en temps réel un ensemble `FRÉQUENTES` de chroniques uniquement minimales, il faudrait alors organiser cet ensemble en treillis de séquences, mais l'insertion dans un treillis est très coûteuse, ce qui nous a poussé à faire le choix présenté ici. On pourra à l'avenir tester si l'utilisation d'un treillis pour l'ensemble `FRÉQUENTES` nuit à l'efficacité en pratique.

3.3.3 Complexité

Le calcul de la complexité dépend beaucoup du nombre de contraintes de la base de contraintes $\mathcal{D}(\mathcal{S}, f_{seuil})$. En effet, la figure 3.7 montre que pour une base de trois couples de types d'évènement ayant chacune trois contraintes, il est possible de construire 27 chroniques de taille 3.

On appelle $n_{\mathcal{D}}$ le nombre de couples d'évènements de la base de contraintes. Supposons que tous les couples de la base aient le même nombre n_τ de contraintes. On a C_n^2 contraintes dans une chronique de taille n. Si on a pour chaque contrainte n_τ possibilités, alors on a pour cette chronique $n_\tau^{C_n^2}$ chroniques possibles rien qu'en combinant les choix de contrainte pour chaque couple. Le nombre d'épisodes sur $\mathbb{E}$ de taille n est $|\mathbb{E}|^n$, donc le nombre de chroniques de taille qui existe n est $|\mathbb{E}|^n \times n_\tau^{C_n^2}$.

La tableau 3.1 donne une évaluation du nombre de chronique que peut avoir à traiter le processus de découverte complète, dans la cas d'une base de contraintes simple, la base $\mathcal{D}_0$ ($n_\tau = 3$ et $|\mathbb{E}| = 3$). Le nombre de la troisième colonne est le nombre de chroniques qui devront potentiellement être traitées dans la boucle de la ligne 8 de l'algorithme 2, dans le cadre de la découverte complète. Dans la pratique, si dans une séquence la taille maximale parmi les chroniques fréquentes est l_{max}, alors l'algorithme 2 ne générera pas de chroniques de taille supérieure à $l_{max}+1$, et donc le nombre de chroniques potentiellement traitées est celui de la ligne $l_{max}+1$. Un deuxième phénomène limite dans la pratique le nombre réel de chroniques de taille n par rapport à celui évalué dans le tableau 3.1, il s'agit du fait que la base de contraintes $\mathcal{D}(\mathcal{S}, f_{seuil})$ ne contient pas les chroniques de taille 2 non fréquentes, ce qui empêche les chroniques basées sur des séquences dont toutes les sous-séquences de taille 2 n'appartiennent pas à $\mathcal{D}(\mathcal{S}, f_{seuil})$ d'être générées (cf. chroniques barrées sur la figure 3.6). En conséquence, le facteur de complexité $|\mathbb{E}|^n$ devient plus faible.

Il est également important de noter que dans la boucle de la ligne 8 de l'algorithme 2, l'étape la plus critique est l'étape de comptage (ligne 14), qui est en $O(|\mathcal{S}|)$, avec $|\mathcal{S}|$ potentiellement très grand. Grâce aux tests de minimalité et de maximalité, il se trouve qu'un très petit nombre de ces chroniques se trouvent effectivement comptées.

La complexité du comptage a été donnée par (Dousson et al. 1993), elle est de $O(Kn^2|\mathcal{S}|)$, où l est la taille de la chronique. Enfin, le dernier critère de complexité de cet algorithme est le nombre de successeurs de chaque chronique. Ce nombre influe sur la complexité car à chaque itération de l'algorithme 2,

n	Nombre de chr. de taille n	Nombre de chr. de taille inférieure à n
2	9	9
3	729	756
4	59049	59805
5	14348907	14408712
6	10460353203	10474761915
7	22876792454961	22887267216876
8	150094635296999121	150117522564215997

TAB. 3.1 – Évaluation du nombre de chroniques pour la base de contraintes $\mathcal{D}_0$ ($n_\tau = 3$ et $|\mathbb{E}| = 3$), avec $f_{seuil} = 2$.

on boucle sur tous ses successeurs (cf. ligne 23) pour les ajouter à l'ensemble `OUVERTES`. L'opération d'ajout d'une chronique à l'ensemble `OUVERTES`peut paraître anodin mais en fait elle consiste à n'ajouter la chronique dans cet ensemble que si elle n'y est pas déjà présente, et comporte donc une phase implicite de test de présence. Pour une chronique $\mathcal{C}$ de taille n, il y a $|\mathbb{E}|$ opérateurs d'ajout d'un type d'évènement et au plus 2 contraintes plus strictes pour chacune des C_n^2 contraintes. Le nombre de successeurs est donc au plus $n_{Succ(\mathcal{C})} = |\mathbb{E}| + 2 \times C_n^2$, c'est-à-dire à $n(n-1) + |\mathbb{E}|$.

3.3.4 Première étude empirique et performances observées

Parmi tous les facteurs de complexité que nous avons énuméré précédemment, il est à prévoir que certains voire un seul soient limitatifs sur l'exécution du processus de découverte. Pour tester cela et pour contribuer à définir les points d'améliorations de la méthode exposée, nous avons mené une première série de tests sur un jeu de données connu.

Complétude

Dans un premier temps, nous avons appliqué l'algorithme 2 à la séquence $\mathcal{S}_0$. Le premier constat est que sur cette séquence, l'algorithme 2, qui est complet, extrait plus de chroniques que la méthode non complète exposée dans (Duong 2001). Le tableau 3.2 récapitule cette comparaison, en ne tenant compte pour la méthode complète que des chroniques ayant un des contraintes incluses dans $[-4, 4]$ pour limiter le nombre de chroniques à faire apparaître dans le tableau. On constate que chaque chronique fréquente extraite est bien minimale dans l'ensemble des chroniques extraites (aucune chronique n'est plus générale qu'une autre).

Observation de certains paramètres de la résolution

Pour observer comment se comporte l'algorithme, nous avons mesuré (toujours avec $\mathcal{S}_0$) à chaque itération les paramètres suivants : *Nb fréquents*, *Nb non-fréquents*, *Nb successeurs* et *Nb comptages*. Nous avons ensuite reporté ces mesures sur les graphes des figures 3.9 et 3.10. Le premier motif de satisfaction est que malgré la grande combinatoire des chroniques de taille 3 ayant 3 types d'évènements, seulement 75 chroniques ont été comptés dans $\mathcal{S}_0$ pour

Méthode utilisée	Chroniques extraites
Méthode non-complète (Duong 2001)	$\langle ABC : A[1,2]B, B[2,3]C, A[1,2]C\rangle$
Méthode complète (algorithme 2)	$\langle BC : B[-4,-3]C\rangle$ $\langle ABC : A[-1,1]B, B[0,2]C, A[-1,1]C\rangle$ $\langle ABC : A[1,2]B, B[2,3]C, A[-1,1]C\rangle$ $\langle ABC : A[2,3]B, B[2,3]C, A[-1,1]C\rangle$ $\langle ABC : A[3,4]B, B[0,2]C, A[-4,1]C\rangle$

TAB. 3.2 – Comparaison des chroniques extraites à partir de $\mathcal{S}_0$ par la méthode non-complète et la méthode complète (avec $win_{max} = 4$).

extraire toutes les chroniques fréquentes de tableau 3.2, alors que potentiellement 756 étaient candidates (cf. tableau 3.1). De plus, une dizaine de chroniques dans chaque ensemble `FRÉQUENTES` et `NON_FRÉQUENTES` sont nécessaires pour permettre cela, ce qui signifie que les tests de minimalité et de maximalité peuvent être très courts. En revanche, le nombre cumulé de successeurs créés et donc d'itérations de la ligne 23 de l'algorithme 2 est plus grand que le nombre d'itérations total, ce qui engendre une perte de temps parfois importante puisque le processus de génération a une complexité élevée. Ce nombre de successeurs créés semble être en évolution linéaire par rapport au nombre d'itérations.

Mesure du taux de génération multiple d'une même chronique

Afin d'étudier plus en profondeur le problème de la génération des successeurs, nous avons mesuré en fonction du nombre d'itérations le nombre total de successeurs nouveaux (cf. figure 3.10). En effet, pour chaque chronique $\mathcal{C}$, il existe plusieurs chroniques $\mathcal{C}'$ telles que $\mathcal{C}$ est le successeur de $\mathcal{C}'$. Le problème est que chacune de ces $\mathcal{C}'$ générera $\mathcal{C}$ parmi ses successeurs, et donc qu'une même chronique peut-être générée un grand nombre de fois. La figure 3.10 montre que sur les premières itérations (les 10 à 20 premières), à peu près toutes les chroniques générées par `GÉNÉRER_SUCCESSEURS` sont nouvelles, mais que sur la suite, les chroniques nouvelles générées se font de plus en plus rares, l'évolution du nombre de chroniques nouvelles générées étant en rapport logarithme avec le nombre d'itérations. Cela signifie que l'algorithme effectue trop d'opérations pour trouver peu de nouvelles candidates. Pour des données plus complexes requérant un très grand nombre d'itérations, on peut donc naturellement attendre que la courbe des nouvelles chroniques générées soit écrasée sous celle du nombre total de chroniques générées, ce qui signifierait que l'algorithme perdrait du temps à «tourner en rond».

Facteur $|\mathcal{S}|$

Pour mesurer l'impact du facteur $|\mathcal{S}|$ sur la complexité, nous avons mis bout à bout x fois la séquence $\mathcal{S}_0$ avec un écart temporel Δ entre deux occurrences successives de $\mathcal{S}_0$ tel que $\Delta > win_{max}$, afin qu'il ne soit pas possible d'avoir des chroniques fréquentes à cheval sur deux occurrences de $\mathcal{S}_0$. En procédant ainsi, on s'assure que si on lance l'algorithme 2 avec $f_{seuil} = 2 \times x$, où 2 est le f_{seuil} choisi pour nos mesures sur une seule occurrence de $\mathcal{S}_0$, on obtiendra alors les mêmes chroniques fréquentes, avec le même nombre d'itérations et la

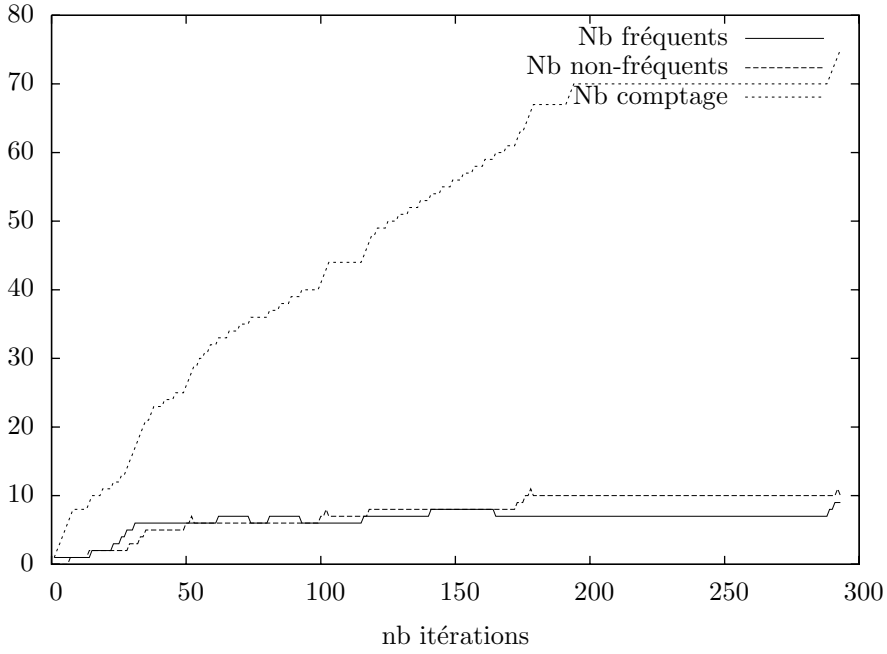


FIG. 3.9 – Évolution du nombre de chroniques dans les ensembles `FRÉQUENTES` et `NON_FRÉQUENTES`, ainsi que du nombre total de passes de comptage effectuées en fonction de l'itération.

même évolution de chacun des paramètres *Nb fréquents*, *Nb non-fréquents*, *Nb successeurs*, *Nb successeurs nouveaux* et *Nb comptages* en fonction de l'itération.

La figure 3.11 représente l'évolution du temps d'extraction en fonction de la taille de la séquence d'évènement $\mathcal{S}$. L'implémentation de l'algorithme 2 ayant servi à faire ces mesures souffrant de nombreux points non-optimisés, il ne faut pas retenir le temps absolu, mais seulement l'évolution. La courbe du temps total d'extraction ne prend pas en compte le temps mis pour élaboré $\mathcal{D}(\mathcal{S}, f_{seuil})$, si bien que le temps représenté sur la figure ne prend en compte que les itérations de la ligne 8 de l'algorithme 2. La courbe montre une évolution linéaire en fonction de la taille. Elle montre également que la plus grande partie du temps d'extraction est dédié au comptage des chroniques, sachant que quelque soit la taille de $\mathcal{S}$, seulement 75 comptages sont nécessaires à l'extraction complète (cf. figure 3.9). Ceci justifie notre choix de l'introduction des ensembles `FRÉQUENTES` et `NON_FRÉQUENTES` dans l'algorithme 2 et nous encourage encore plus à réduire par tous les moyens le nombre de comptage effectué. Une fois cette observation faite, la linéarité observée n'est pas surprenante car la complexité du comptage des chroniques par *CRS* est en $O(|\mathcal{S}|)$ (Dousson et al. 1993). Le temps de traitement autre que le comptage semble être constant en fonction de la taille de $|\mathcal{S}|$, ce qui découle directement du fait que l'évolution des paramètres de résolution est la même quelque soit la taille.

Facteur n

Le cas de la séquence $\mathcal{S}_0$ avec une largeur de fenêtre égale à 4 est tel qu'il n'existe pas dans la séquence des chroniques fréquentes de taille supérieure à 3. Ceci rend l'exécution de l'algorithme 2 immédiate ($385ms$) mais nous savons d'après le tableau 3.1 que plus la taille de la chronique fréquente la plus grande est grande, plus le temps d'exécution est long, avec un rapport exponentiel. Afin de nous assurer que la séquence comporte au moins une chronique de taille 4, nous avons ajouté l'évènement $(B, 4)$ à $\mathcal{S}_0$, en s'attendant à ce que l'algorithme trouve des occurrences de chroniques fréquentes basées sur l'épisode $ABBC$. L'exécution de l'algorithme 2 sur cette nouvelle séquence est de $3758ms$. En ajoutant l'évènement $(A, 8)$, on s'assure qu'il existe des chroniques fréquentes basée sur $AABBC$ et donc de taille 5. Sur cette nouvelle séquence, l'algorithme 2 ne répond pas dans les cinq premières minutes, car il ne trouve plus de nouvelles chroniques : «tourne en rond».

Facteur $|\mathbb{E}|$

Pour mesurer l'impact de $|\mathbb{E}|$, nous avons généré de manière aléatoire des séquences de même taille ($|\mathcal{S} = 100|$), avec une même largeur de fenêtre win_{max} ($win_{max} = 4$) mais avec un nombre d'évènements différents. Nous avons utiliser un $f_{seuil} = 2$, ce qui est très faible pour une séquence de 100 évènements et qui engendre potentiellement un très grand nombre de chroniques fréquentes, mais nous l'avons fait dans le but de pousser l'algorithme dans ses retranchements. Dans chaque séquence, chaque type d'évènement a le même nombre d'occurrences. La répartition des évènements dans chaque séquence est effectuée de la manière suivante. Après chaque nouvel évènement (E, t) introduit, nous introduisons un nouvel évènement (E', t'), où E' est choisi au hasard parmi dans $\mathbb{E}$ et $t' = t + \delta$, avec δ un nombre choisi au hasard entre 0 et 5. Nus mesurons

le temps total de découverte de toutes les chroniques ainsi que le temps dédié au comptage. Les résultats de ces mesures ont été reportés sur le graphe de la figure 3.12. Ce qu'on observe, c'est qu'en dessous de 9 types d'évènements, l'algorithme «tourne en rond». Au delà, de 9, plus $|\mathbb{E}|$ est grand, plus le temps de découverte est court, et ce en rapport inverse avec $|\mathbb{E}|$. Ceci est du au fait que pour des grands nombres de types d'évènements et avec l'équirépartition des types d'évènement dans la séquence, il est peu probable que des chroniques de grande taille soient fréquentes au sein d'une même fenêtre de largeur win_{max}. En effet, avec cette équirépartition des types d'évènement, une largeur de fenêtre fixe et un $|\mathbb{E}|$ grandissant, la probabilité de présence de chaque type d'évènement dans une même fenêtre baisse, et la probabilité de trouver une chroniques fréquente avec aussi. En fait, on peut voir le facteur $|\mathbb{E}|$ comme une variable du facteur plus limitant n (taille de la chronique fréquente la plus grande). $|\mathbb{E}|$ joue donc indirectement sur le temps d'exécution, via n.

3.3.5 Améliorations futures

Le premier critère d'amélioration à étudier concerne le fait que pour des paramètres d'entrée critiques (eg. $|\mathbb{E}|$ trop faible, f_{seuil} trop faible, win_{max} trop grand, etc), l'algorithme a tendance à «tourner en rond». Plus précisément, au bout d'un grand nombre d'itérations, on s'aperçoit que l'algorithme, pourtant loin d'avoir trouvé toutes les chroniques, n'arrive pas à générer de chronique qui soit «nouvelle» au sens où elle est digne d'être comptée (cf. figure 3.10). Cela est du au fait que même si les chroniques fréquentes sont loin d'être toutes trouvées, les ensembles `FRÉQUENTES` et `NON_FRÉQUENTES` élaguent suffisamment l'espace de recherche pour que les chroniques nouvelles fréquentes soit rares, compte tenu des chroniques de l'espace `OUVERTES` qu'il reste à traiter. Autrement dit, dans ces moments où l'algorithme «tourne en rond», ces deux ensembles sont assez représentatifs des chroniques de l'ensemble `OUVERTES`, ce qui signifie que très peu de chroniques parmi les chroniques de `OUVERTES` permettent de générer des chroniques vraiment nouvelles. Une première amélioration à l'algorithme 2 serait de lui donner une indication permettant de déterminer dans `OUVERTES` les chroniques les plus prometteuses en terme de nouveauté des successeurs.

Ce problème est très proche d'un autre problème de l'algorithme. Même sans considérer cet effet de «tourner en rond», l'algorithme une fois terminé aura eu à générer plusieurs fois la même chronique via la méthode `GÉNÉRER_SUCCESSEURS`. Ceci provient du fait qu'une chronique possède de nombreux prédécesseurs, et d'autant plus que win_{max} et $\mathbb{E}$ sont grands. Une optimisation qui peut s'avérer intéressante consisterait à trouver une stratégie permettant, au moment de la génération, d'avoir des connaissances sur les chroniques ayant déjà été antérieurement générées. Il est probable qu'une telle stratégie nécessite une grande capacité de mémoire pour stocker ces connaissances, auquel cas il faudrait ensuite mesurer si le jeu en veut la chandelle.

Mais le problème des successeurs pose surtout le problème de l'heuristique sous-jacente au choix de la chronique à traiter ensuite. Ce choix s'effectue via la méthode `PRENDRE_PREMIER`. Pour l'heure nous choisissons le premier élément de `OUVERTES` selon une relation d'ordre purement arbitraire, notamment basée sur l'ordre lexicographique des épisodes sur lesquels les chroniques sont basées. Ceci a pour effet de traiter d'abord toutes les chroniques d'une même «famille», où une «famille» de chroniques serait constituer des chroniques basées sur des

épisodes et des contraintes plus ou moins proches. Par exemple, l'algorithme traitera en priorité toutes les chroniques basées sur `AB`, puis toutes les chroniques basées sur `ABB` et ses sur-épisodes, puis toutes les chroniques basées sur `ABC` et ses sur-épisodes, remettant ainsi à beaucoup plus tard le traitement des chroniques basées sur `EG`, pourtant potentiellement très différentes en terme de sémantique portée. Il est possible que ce soit cette heuristique arbitraire qui soit pour l'heure responsable de l'effet de «tourner en rond». Il serait intéressant d'appliquer des heuristiques qui prennent en compte des critères élaborées comme la «nouveauté» de la chronique par rapport aux chroniques existante, mais encore l'«utilité», la «concision», la «confiance», etc. On peut trouver de l'inspiration pour ces heuristiques dans les travaux qui traitent des mesures d'intérêts des motifs extraits pour le post-traitement en fouille de données (Geng & Hamilton 2006). Le fait que l'algorithme soit guidé par des heuristiques pertinentes est un enjeu de taille, car si les premières chroniques extraites sont jugées satisfaisantes par l'utilisateur, alors l'algorithme n'aura pas besoin de s'exécuter plus longtemps, même s'il n'en est qu'au millième de toutes les chroniques candidates.

Enfin, le dernier critère d'amélioration que nous traitons ici concerne de manière plus générale les performances de l'algorithme. En l'état, il a une capacité limitée en terme de longueur de séquence, et taille de fenêtre et nombre de types d'évènement à cause de la grande combinatoire des chroniques candidates (cf. section 3.3.3). Une des manières la plus intéressante de permettre à cet algorithme de s'exécuter en temps raisonnable avec de tels paramètres moins restreints serait d'intégrer des contraintes spécifiées par l'utilisateur. On trouve dans la littérature de nombreux travaux traitant du problème de la fouille de données temporelles sous-contraintes (Wojciechowski 2001, Antunes & Oliveira 2004, Pei et al. 2007, de Amo & Furtado 2007) et dont on peut s'inspirer. (Pei et al. 2002) propose même une classification des contraintes et de leur propriétés sur le processus de fouille. On peut légitiment s'attendre à ce que certaines contraintes aient un grand effet sur le temps d'exécution. En effet, considérons le cas d'une séquence contenant des chroniques de taille 6. Dans ce cas, le tableau 3.1 donne un nombre bien trop grand de chroniques candidates pour pouvoir être toutes traitées en temps raisonnable. Si en revanche, l'utilisateur spécifie qu'il veut que les chroniques extraites soient basées sur l'épisode `ABBC`, alors cela peut diviser jusqu'à 10^4 (cf. le nombre de chroniques de taille 4 du tableau 3.1) le nombre de chroniques candidates. Plus les contraintes seront nombreuses et plus l'algorithme deviendra praticable en temps d'exécution. C'est pourquoi nous imaginons un cycle interactif d'extraction, dans lequel l'utilisateur lance une première fois le processus de découverte sans aucune contrainte et spécifie ensuite des nouvelles contraintes au fur et à mesure qu'il découvre les résultats renvoyés, puis relance l'algorithme, et ainsi de suite. Ce cycle interactif donne à chaque itération accès des des chroniques de taille plus grande du fait que les contraintes spécifiées par l'utilisateur sont plus nombreuses et plus strictes. C'est pourquoi, même si la complexité du processus de découverte complète peut s'évaluer en $O(a^{n^2})$, nous ne désespérons pas de permettre le traitement pour des n grands par ce cycle interactif.

Ceci constituerait une «pierre deux coups» puisqu'il s'agit également de notre motivation originelle que de permettre à l'utilisateur d'avoir la main sur l'algorithme et de l'intégrer beaucoup plus dans le processus de découverte que les processus d'extraction classiques. C'est dans cette piste-là que nous pourrons

également essayer d'appliquer des travaux proches existants. En effet, à chaque itération, le moment où l'utilisateur spécifie des contraintes en fonction des résultats retournés par l'algorithme peut être opportunément mis à profit pour acquérir des connaissances de l'utilisateur en interaction avec lui (Cordier et al. 2007). Cette base de connaissances peut ensuite elle-même être utilisée pour mesurer de manière plus pertinente l'intérêt de chaque chronique candidate et ainsi orienter le processus de fouille avec une heuristique dite «subjective», c'est-à-dire qui prend en compte l'opinion personnelle de l'utilisateur (Fauré 2007).

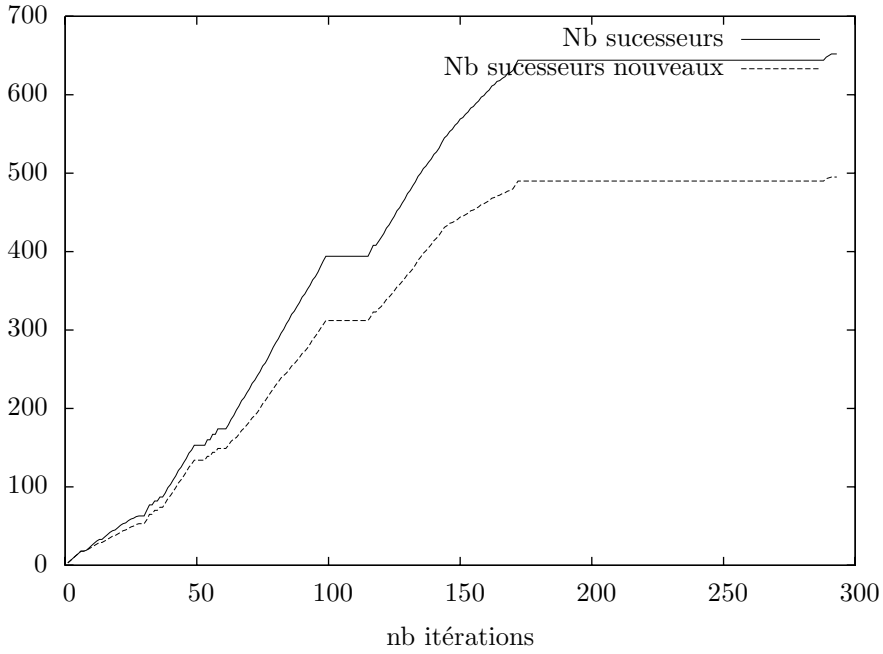


FIG. 3.10 – Rapport entre le nombre de successeurs générés et le nombre de successeurs réellement nouveaux.

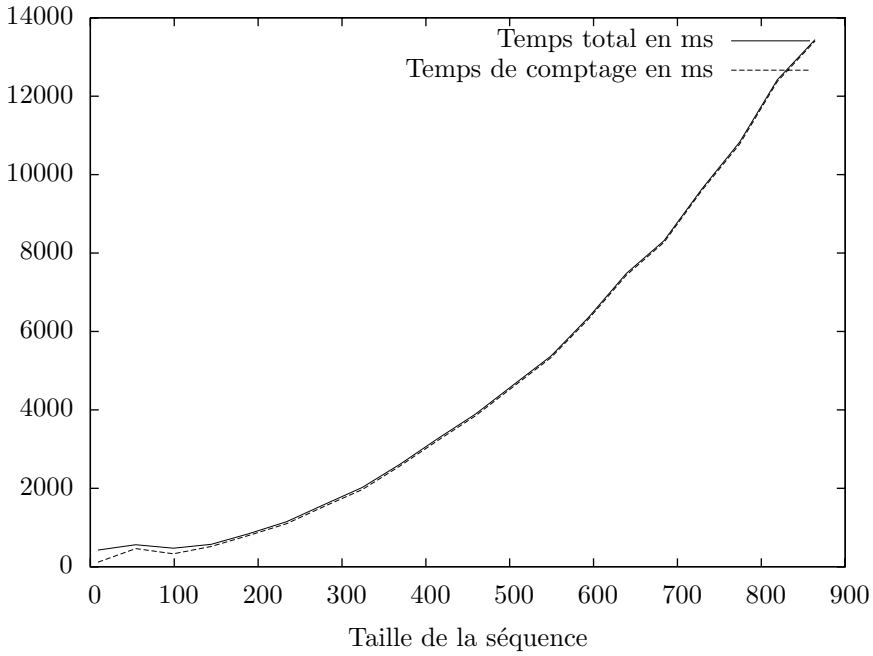


FIG. 3.11 – Courbe d'évolution du temps total d'extraction des chroniques et du temps de comptage en fonction de la taille de $\mathcal{S}$.

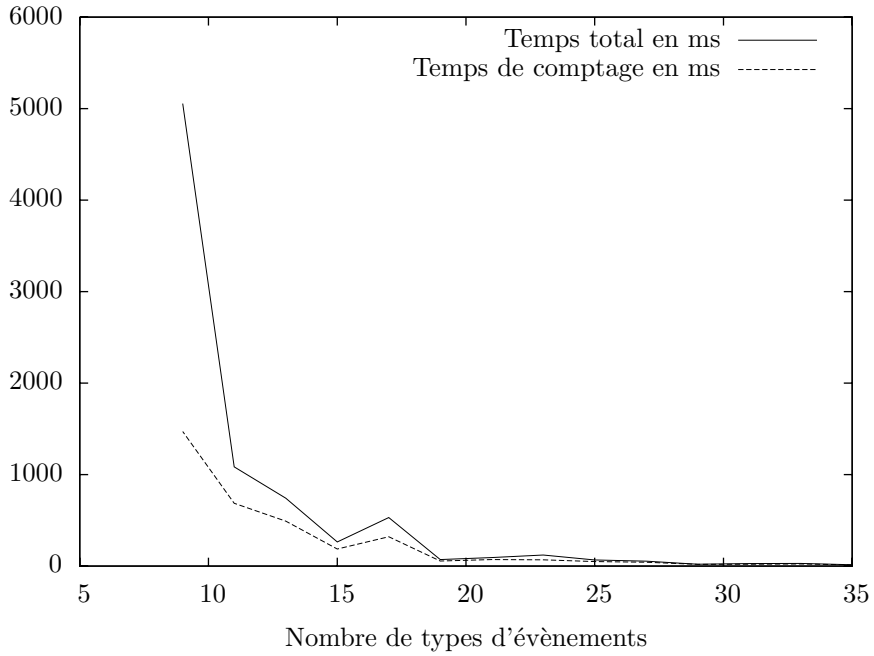


FIG. 3.12 – Courbe d'évolution du temps total d'extraction des chroniques et du temps de comptage en fonction de $|\mathbb{E}|$.

Chapitre 4

Conclusions et travaux à venir

4.1 Conclusion

Afin d'implémenter les approches interactives d'assistance à partir de traces d'interactions expliquées dans nos précédents rapports et publications, nous avons proposé ici une méthode de découverte interactive de chroniques à partir de séquences d'évènements qui servira de base à la réalisation de l'*agent analyseur*. Le choix du formalisme des chroniques de (Dousson et al. 1993) pour exprimer les motifs temporels de la trace d'interaction a été effectué car les autres méthodes de fouilles de données temporelles existantes ne permettent pas une pleine expressivité des relations entre évènements et la quantification de celles-ci. Ce choix est cependant discutable à l'égard de l'existant en matière de fouille de workflows (van der Aalst et al. 2003, van der Aalst et al. 2004). En réalité, l'expressivité des workflows est supérieure à celle des chroniques, car un workflow permet d'exprimer des situations comme «"A", suivi de "B et C dans n'importe quel ordre", puis de "D"» ou comme «"A", suivi de "B ou C", puis de "D"», alors que les chroniques ne le peuvent pas. Si nous avons fait ce choix c'est que les performances des méthodes existantes de fouille de workflows sont encore assez limitées, et le problème posé reste assez loin de nos besoins.

Un algorithme a été proposé pour implémenter notre méthode et des premières expériences ont été réalisées pour observer son comportement en situation réelle. Sa complexité est exponentielle, mais comme l'algorithme, inspiré de $\mathcal{A}^*$, maintient à tout moment un ensemble de chroniques fréquentes trouvées jusque-là, il est possible d'interrompre le processus de découverte avant qu'il ait touché à sa fin en se déclarant satisfait de ces chroniques. Deux aspects d'optimisation sont encore à étudier très prochainement : le choix d'une heuristique permettant d'arriver le plus rapidement possible à un ensemble de chroniques fréquentes satisfaisantes et l'intégration de contraintes spécifiées par l'utilisateur dans le processus de découverte.

D'autres améliorations et extensions sont également à prévoir par la suite afin de rapprocher le formalisme des motifs temporels extraits de celui des traces d'interactions modélisées : la prise en compte des évènements persistants et l'extraction de chroniques avec variables. Les deux sections suivantes en expliquent

brièvement les enjeux et donnent quelques détails sur ces deux perspectives.

4.2 Prise en compte des évènements persistants

Les méthodes de fouille de séquences que nous avons évoquées en introduction, et la méthode que nous avons exposée dans ce rapport ne traitent que des séquences dont chaque évènement est ponctuel. Ceci constitue souvent une grande simplification par rapport à la temporalité réelle des données qui sont fouillées. Plus précisément et pour ce qui nous concerne, le meta-modèle des traces d'interaction modélisées permet de représenter un observé dans le temps notamment sous forme d'intervalle. Dès qu'on commence à traiter les évènements comme non ponctuels, ou *persistants*, le processus de découverte se complexifie grandement. En effet, seulement 3 relations élémentaires peuvent lier deux évènements ponctuels : *avant*, *après* et *simultanément*. Il en résulte qu'en les composant pour l'expression des motifs temporels, seuls 7 types de relations entre évènements sont possibles :*avant*, *après*, *simultanément*, *avant ou simultanément*, *simultanément ou après*, *avant ou après*, *avant ou après ou simultanément*. Dans la pratique, les méthodes de fouille de séquences ne cherchent pas à extraire ces 7 relations. Dans le cas de la recherche d'épisodes en série, elles n'en cherchent que de deux types : *avant* et *après*. Dans le cas de la recherche d'épisodes parallèles, elles n'en cherchent que d'un type : *avant ou après ou simultanément*. Les 4 autres types de relations sont amalgamées à ces 3 relations, ce qui n'est généralement pas trop limitatif en terme d'expressivité des motifs puisqu'il s'agit à chaque fois d'une approximation à une égalité près.

Il en est tout autre chose lorsqu'il s'agit d'intervalles. (Allen 1983) identifie 13 types de relations élémentaires possibles entre deux intervalles, ce qui fait 8191 relations possibles entre deux intervalles pour l'expression d'un motif. Les méthodes de fouille de séquences d'évènements persistants comme (Wu & Chen 2007) sont donc généralement incomplètes par souci d'efficacité et de simplicité. Seuls (Patel et al. 2008) proposent une méthode complète d'extraction de motifs temporels, implémentée avec l'algorithme `IEMiner`. Dans (Chen & yi Wu 2006), les auteurs se simplifient la tâche en transformant la séquence d'évènements persistants en séquence d'évènements ponctuels en remplaçant chaque intervalle par deux dates ponctuelles : la borne inférieure de l'intervalle et la borne supérieure. Le problème de toutes ces méthodes est qu'elles ne permettent pas de quantifier les relations entre intervalles. Seul (Laxman et al. 2007*b*) propose de quantifier la durée des intervalles, mais de manière très incomplète et très imparfaite, puisque les évènements sont préalablement transformés en évènements ponctuels auxquels on ajoute une valeur de durée parmi 3 ou 4 valeurs de durées possibles.

Ce que nous proposerons dans la suite de notre travail sera de traiter le cas des séquences d'évènements persistants tout en pouvant quantifier les relations entre intervalles d'un même motif temporel, et sans passer par les relations d'Allen. Pour cela, nous pensons utiliser la même astuce que (Chen & yi Wu 2006) et ne considérer pour chaque intervalle que ses dates de début et de fin. À partir de cela, nous bâtirons et rechercherons dans la séquence des chroniques d'évènements persistants telles qu'au lieu d'avoir une contrainte par couple d'évènements, nous aurons plusieurs contraintes par couple. Par exemple, entre deux évènements persistants A et B, en notant a^- et a^+ les bornes inférieure et

supérieure de A, b^- et b^+ les bornes inférieure et supérieure de B, la chronique d'évènements persistants sur `AB` comportera les 6 contraintes suivantes :

- la contrainte temporelle entre a^- et a^+ (contrainte sur la durée de A),
- la contrainte temporelle entre a^- et b^-,
- la contrainte temporelle entre a^- et b^+,
- la contrainte temporelle entre a^+ et b^-,
- la contrainte temporelle entre a^+ et b^+,
- la contrainte temporelle entre b^- et b^+ (contrainte sur la durée de B).

4.3 Découverte de chroniques avec variables non généralisées

(Duong 2001) propose une extension de son travail pour prendre en compte des informations supplémentaires sur les évènements par l'intermédiaire d'attributs. Ainsi chaque évènement de la séquence peut avoir plusieurs attributs qui lui sont attachés. De plus des relations peuvent exister entre ces attributs. Ces relations servent à représenter des connaissances sur le domaine sur lequel opèrent les évènements de la séquence. Par exemple, une séquence d'évènements avec attributs pourrait être :

$$\mathcal{S} = \langle (A[c_1, p_1], 1)(B[c_1, p_1], 2)(A[c_2, p_1], 6) \rangle$$
$$\mathit{contient}(p_1, c_1)$$
$$\mathit{contient}(p_1, c_2)$$

où A et B sont des types d'évènement et c_1, c_2, et p_1 sont des valeurs d'attributs. Par exemple, supposons que A représente l'action de «cliquer», B l'action de «entrer du texte», c_1 l'objet «champ» d'un formulaire p_1, et c_2 l'objet «bouton de validation» de ce formulaire. Si de plus il existe une relation «contient» entre p_1 et c_1, et une autre entre p_1 et c_1, alors la séquence $\mathcal{S}$ déclare qu'à la date 1, on a observé un «clique» sur le champ c_1 du formulaire p_1 ; qu'à la date 2 on a observé que ce même champ a été «rempli», puis à la date 6 que le bouton c_2 de validation de ce formulaire a été cliqué. En résumé, cette séquence $\mathcal{S}$ décrit le remplissage et l'envoi d'un formulaire.

En plus de cette représentation adaptée à des données plus structurées, (Duong 2001) propose une méthode de découverte de chroniques avec variables. Ce nouveau motif est constitué des mêmes éléments que la chronique au sens où nous l'entendions jusque-là auxquels s'ajoute un «graphe de relations». Dans cette chronique, des variables sont attachées aux évènements, et le graphe de relations exprime des contraintes structurelles sur ces variables. Par exemple la chronique de taille 2 ci dessous :

$$\langle A[?x_1, ?x_2]B[?x_3, ?x_4] : A[?x_1, ?x_2][1, 2]B[?x_3, ?x_4] | \mathit{contient}(?x_2, ?x_1) \rangle$$

exprime que les occurrences doivent être composées de A suivi de B de 1 à 2 unités de temps après, et que le 2^{e} attribut de A doit être en relation `contient` avec son 1^{er} attribut. L'extraction de chroniques avec variables est une bonne idée qui paraît très prometteuse dans l'optique de l'appliquer à des traces d'interaction modélisées, mais la méthode proposée par (Duong 2001) souffre de quelques problèmes.

– Chaque attribut d'un type d'évènement ne peut avoir qu'une seule variable dans tout le processus d'extraction. Par exemple, le 1$^{\text{er}}$ attribut de l'évènement A est toujours représenté par la seule variable $?x_1$, ayant ici pour domaine $\{c_1, c_2\}$. Ceci est dû au fait que les variables sont construites une et une seule fois pour chaque attribut de chaque type d'évènement, et à partir de la séquence. Ainsi, il ne sera par exemple pas possible de créer deux chroniques basées sur `AB`, telle que la 1$^{\text{re}}$ variable de la première chronique ait pour domaine $\{c_1, c_2\}$ et la première variable de l'autre chronique ait pour domaine $\{c_3\}$. De plus, si dans une chronique il apparaît deux fois le même type d'évènement, disons A, alors la première occurrence de A dans la chronique et la deuxième occurrence auront exactement les mêmes $?x_1$ et $?x_2$ (les variables de l'une et de l'autre occurrence sont en quelques sortes liées).
– Le graphe de relations qui compose la chronique est construit à partir des évènements présents dans la chronique uniquement, et non à partir de la séquence. Ce problème est en fait une conséquence du premier. En conséquence, le graphe de relation d'une chronique ne pourra pas être pris comme étant représentatif d'une situation bien particulière puisqu'il aura été élaboré complètement à part, à l'initialisation du processus de découverte. Ce graphe de relations représente en réalité plutôt un comportement général de la séquence (les relations entre les valeurs de chaque attribut de chaque type d'évènement de la séquence), et il paraît alors absurde de l'introduire au sein de la chronique qui vise à décrire un comportement local.

À l'avenir, nous souhaitons intégrer les contraintes structurelles dans le mécanisme de découverte de chroniques, mais nous tenons à ce que les chroniques avec variables élaborées puissent être représentatives de situations particulières, et non d'un état général de la séquence. Pour cela, nous pourrons nous inspirer du travail qui a été réalisé par (Duong 2001) mais en apportant de nombreuses améliorations dans le processus de découverte. Il est malheureusement à attendre que ces améliorations soient coûteuses en complexité.

Bibliographie

Agrawal, R. & Srikant, R. (1995), Mining sequential patterns, *in* P. S. Yu & A. S. P. Chen, eds, 'Eleventh International Conference on Data Engineering', IEEE Computer Society Press, Taipei, Taiwan, pp. 3–14.

Allen, J. F. (1983), 'Maintaining knowledge about temporal intervals', *Commun. ACM* **26**(11), 832–843.

Antunes, C. & Oliveira, A. L. (2004), Constraint relaxations for discovering unknown sequential patterns, *in* 'KDID', pp. 11–32.

Casas-Garriga, G. (2003), Discovering unbounded episodes in sequential data, *in* 'PKDD', pp. 83–94.

Chen, Y.-L. & yi Wu, S. (2006), Mining temporal patterns from sequence database of interval-based events, *in* 'FSKD', pp. 586–595.

Cordier, A., Fuchs, B., Lieber, J. & Mille, A. (2007), Interactive Knowledge Acquisition in Case Based Reasoning, *in* D. K. David Wilson, ed., 'Workshop "Knowledge Discovery and Similarity in Case-Based Reasoning", ICCBR'07', pp. 85–94.

Cram, D. (2007*a*), 'Raisonner à partir de l'expérience tracée : Application à un environnement collaboratif', Accessible sur le site Web du projet PROCOGEC : http ://www.procogec.com/ ?download=procogec_livrable_T3.1_partie1.pdf.

Cram, D. (2007*b*), 'Techniques d'extraction de connaissances pour la facilitation des tâches à base de traces d'interactions', Accessible sur le site Web du projet PROCOGEC : http ://www.procogec.com/ ?download=procogec_livrable_T3.1_partie2.pdf.

Cram, D., Fuchs, B., Prié, Y. & Mille, A. (2008), An approach to User-Centric Context-Aware Assistance based on Interaction Traces, *in* 'MRC 2008 : Proceedings of the 5th Workshop on Modelling and Reasoning in Context'.

de Amo, S. & Furtado, D. A. (2007), 'First-order temporal pattern mining with regular expression constraints', *Data Knowl. Eng.* **62**(3), 401–420.

Dechter, R., Meiri, I. & Pearl, J. (1991), 'Temporal constraint networks', *Artif. Intell.* **49**(1-3), 61–95.

Dousson, C., Gaborit, P. & Ghallab, M. (1993), Situation recognition : Representation and algorithms, *in* 'IJCAI', pp. 166–174.

Dousson, C. & Duong, T. V. (1999), Discovering chronicles with numerical time constraints from alarm logs for monitoring dynamic systems, *in* 'IJCAI '99 : Proceedings of the Sixteenth International Joint Conference on Artificial

Intelligence', Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, pp. 620–626.

Dousson, C., Maigat, P. L. & R&d, F. T. (2007), Chronicle recognition improvement using temporal focusing and hierarchization, *in* 'In IJCAI', pp. 324–329.

Duong, M. T. V. (2001), Découverte de chroniques à partir de journaux d'alarmes. Application à la supervision de réseaux de télécommunications, PhD thesis, Institut National Polytechnique de Toulouse.

Fauré, C. (2007), Découverte de réseau pertinents par l'implémentation d'un réseau bayésien : application à l'industrie aéronautique, PhD thesis, Institut National des Sciences Appliquées de Lyon.

Geng, L. & Hamilton, H. J. (2006), 'Interestingness measures for data mining : A survey', *ACM Comput. Surv.* **38**(3), 9.

Han, J. & Kamber, M. (2006), *Data Mining : Concepts and Techniques, Second Edition*, Morgan Kaufmann.

Huang, K.-Y. & Chang, C.-H. (2008), 'Efficient mining of frequent episodes from complex sequences', *Inf. Syst.* **33**(1), 96–114.

Laxman, S., Sastry, P. S. & Unnikrishnan, K. P. (2007*a*), A fast algorithm for finding frequent episodes in event streams, *in* 'KDD '07 : Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining', ACM, New York, NY, USA, pp. 410–419.

Laxman, S., Sastry, P. & Unnikrishnan, K. (2007*b*), 'Discovering frequent generalized episodes when events persist for different durations', *IEEE Transactions on Knowledge and Data Engineering* **19**(9), 1188–1201.

Mackworth, A. K. & Freuder, E. C. (1985), 'The complexity of some polynomial network consistency algorithms for constraint satisfaction problems', *Artificial Intelligence* **25**(1), 65–74.

Mannila, H., Toivonen, H. & Verkamo, A. I. (1997), 'Discovery of frequent episodes in event sequences', *Data Mining and Knowledge Discovery* **1**(3), 259–289.

Montanari, U. (1974), 'Networks of constraints : fundamental properties and applications to picture processing', *Information Sciences* **7**, 95–132.

Méger, N. (2004), Recherche automatique des fenetres temporelles optimales des motifs séquentiels, PhD thesis, Institut National des Sciences Appliquées de Lyon.

Patel, D., Hsu, W. & Lee, M. L. (2008), Mining relationships among interval-based events for classification, *in* 'SIGMOD '08 : Proceedings of the 2008 ACM SIGMOD international conference on Management of data', ACM, New York, NY, USA, pp. 393–404.

Pei, J., Han, J. & Wang, W. (2002), 'Mining sequential patterns with constraints in large databases', pp. 18–25.

Pei, J., Han, J. & Wang, W. (2007), 'Constraint-based sequential pattern mining : the pattern-growth methods', *J. Intell. Inf. Syst.* **28**(2), 133–160.

van der Aalst, W. M. P., van Dongen, B. F., Herbst, J., Maruster, L., Schimm, G. & Weijters, A. J. M. M. (2003), 'Workflow mining : a survey of issues and approaches', *Data Knowl. Eng.* **47**(2), 237–267.

van der Aalst, W., Weijters, T. & Maruster, L. (2004), 'Workflow mining : Discovering process models from event logs', *IEEE Transactions on Knowledge and Data Engineering* **16**(9), 1128–1142.

Wojciechowski, M. (2001), 'Interactive constraint-based sequential pattern mining', *Lecture Notes in Computer Science* **2151**, 169–? ?

Wu, S.-Y. & Chen, Y.-L. (2007), 'Mining nonambiguous temporal patterns for interval-based events', *IEEE Trans. on Knowl. and Data Eng.* **19**(6), 742–758.

Zaki, M. J. (2001), 'Spade : An efficient algorithm for mining frequent sequences', *Machine Learning* **42**(1/2), 31–60.