Εθνικό Μετσόβιο Πολυτεχνείο

Σχολή Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών
Τομέας Τεχνολογίας Πληροφορικής και Υπολογιστών

Αναγνώριση σύνθετων δραστηριοτήτων με χρήση Πιθανοτικού Λογισμού Γεγονότων σε ροές δεδομένων

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

του

ΠΕΡΙΚΛΗ ΜΑΝΤΕΝΟΓΛΟΥ

Επιβλέποντες: Γεώργιος Στάμου Αναπλ. Καθηγητής Ε.Μ.Π.

29 Οκτωβρίου 2019

Εθνικό Μετσόβιο Πολυτεχνείο

Σχολή Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών
Τομέας Τεχνολογίας Πληροφορικής και Υπολογιστών

Αναγνώριση σύνθετων δραστηριοτήτων με χρήση Πιθανοτικού Λογισμού Γεγονότων σε ροές δεδομένων

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

του

ΠΕΡΙΚΛΗ ΜΑΝΤΕΝΟΓΛΟΥ

Επιβλέποντες: Γεώργιος Στάμου, Αναπλ. Καθηγητής Ε.Μ.Π

Εγκρίθηκε από την τριμελή εξεταστική επιτροπή την 29η Οκτωβρίου 2019.

.....	.....	.....
Γεώργιος Στάμου	Νικόλαος Παπασπύρου	Ανδρέας Γεώργιος Σταφυλοπάτης
Αναπλ. Καθηγητής Ε.Μ.Π.	Καθηγητής Ε.Μ.Π.	Καθηγητής Ε.Μ.Π.

Περίληψη

Ο στόχος αυτής της εργασίας είναι η επέκταση ενός πιθανοτικού αλγορίθμου, βασισμένου σε χρονικά διαστήματα, που χρησιμοποιείται για την αναγνώριση σύνθετων γεγονότων Αυτή η επέκταση οδηγεί σε ένα σύστημα το οποίο είναι σε θέση να εξάγει πιθανοτικές αναγνωρίσεις σύνθετων γεγονότων σε χρονικά διαστήματα από μία ροή απλών γεγονότων χαμηλού επιπέδου. Συγκεκριμένα, εισάγουμε τη χρήση χρονικών παραθύρων και μνήμης εργασίας για να τροποποιήσουμε την υπάρχουσα προσέγγιση. Σε αυτή τη διπλωματική εργασία, περιγράφουμε τον προαναφερθέντα αλγόριθμο, αναλύουμε την απόδοση και την ορθότητά του και διερευνούμε τις απαιτήσεις του σε μνήμη. Τέλος, χρησιμοποιούμε ένα σύνολο δεδομένων που σχετίζεται με την αναγνώριση ανθρώπινης δραστηριότητας για να δοκιμάσουμε το προτεινόμενο σύστημα.

Λέξεις Κλειδιά: τεχνητή νοημοσύνη, αναγνώριση σύνθετων γεγονότων, αναγνώριση ανθρώπινης δραστηριότητας, Λογισμός Γεγονότων

Abstract

The goal of this thesis is to extend a probabilistic interval-based algorithm used for complex event recognition. This extension results in a system which is able to derive probabilistic complex event occurrences in temporal intervals from a stream of simple, low level events. More specifically, we introduce the concept of temporal windows and a working memory to build on the existing approach. In this thesis, we describe the aforementioned algorithm, analyse its performance and correctness and explore its memory requirements. Finally, we employ a dataset for human activity recognition to test the proposed system.

Keywords: artificial intelligence, complex event recognition, human activity recognition, Event Calculus

Ευχαριστίες

Σε αυτό το σημείο, θα ήθελα να ευχαριστήσω ορισμένους ανθρώπους που, με τις συμβουλές και την υποστήριξη τους, με βοήθησαν να ολοκληρώσω αυτή τη διπλωματική εργασία. Πρωτίστως, νιώθω πολύ τυχερός για την καθοδήγηση που έλαβα από τους επιβλέποντες μου από το Ε.Κ.Ε.Φ.Ε. "Δημόκριτος", τον Δρ. Αλέξανδρο Αρτίκη και τον Δρ. Γιώργο Παλιούρα. Εκείνοι ήταν πάντα πρόθυμοι να με συμβουλέψουν καθ΄ όλη τη διάρκεια εκπόνησης της εργασίας, τόσο αυτοπροσώπως στις εβδομαδιαίες συναντήσεις μας όσο και μέσω mail. Οι παρατηρήσεις και τα σχόλια τους με βοηθούσαν να ξεπεράσω τα εμπόδια που συναντούσα, ενώ παράλληλα μου έδιναν έναυσμα για περισσότερη μελέτη και εμβάθυνση στον τομέα της Τεχνητής Νοημοσύνης.

Φυσικά, ευχαριστώ θερμά τον Δρ. Γιώργο Στάμου της Σχολής Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών του ΕΜΠ που με εμπιστεύτηκε στην ολοκλήρωση αυτή της εργασίας. Είναι βέβαιο πως οι διαλέξεις του σε μαθήματα όπως η «Τεχνητή Νοημοσύνη» και τα «Συστήματα και Τεχνολογίες Γνώσεις» με ώθησαν προς αυτόν τον ερευνητικό τομέα.

Θα ήθελα να ευχαριστήσω τους φίλους μου για την υποστήριξη καθ΄όλη τη διάρκεια της ενασχόλησης μου με την εργασία αυτή. Πολλές φορές οι συμβουλές και τα θετική τους διάθεση με ενθάρρυναν να συνεχίσω.

Επίσης, θα ήθελα να ευχαριστήσω όλους τους συνεργάτες μου από το Ε.Κ.Ε.Φ.Ε. "Δημόκριτος" για τις πολύτιμες συμβουλές τους. Ιδιαίτερα ευχαριστώ τον Χρήστο Βλασσόπουλο που με βοήθησε παρέχοντας μου διευκρίνησεις και χρήσιμο κώδικα στην αρχή της διαδικασίας.

Κλείνοντας, ευχαριστώ μέσα από την καρδιά μου τους γονείς μου για την συνεχή υποστήριξη τους κατά τη διάρκεια ολόκληρης της σχολικής και ακαδημαϊκής μου ζωής. Ήταν πάντα εκεί για να με ενθαρρύνουν και να μου συμπαρασταθούν σε κάθε δυσκολία που συναντούσα.

Περιεχόμενα

Κατάλογος Σχημάτων

Κατάλογος Πινάκων

Κεφάλαιο 1

Εισαγωγή

Τα συστήματα Αναγνώρισης Σύνθετων Γεγονότων (Complex Event Recognition – CER) χρησιμοποιούν αλγόριθμους που βασίζονται σε αναγνώριση προτύπων με σκοπό να συμπεράνουν σύνθετα γεγονότα από μια ροή εισόδου που αποτελείται από «απλά, παραγόμενα γεγονότα» [1]. Αυτή η είσοδος προέρχεται από την επεξεργασία ενδείξεων και μετρήσεων διαφόρων αισθητήρων, αναλόγως με τον τομέα που αφορά η εκάστοτε εφαρμογή. Η αυτόματη αναγνώριση σύνθετων γεγονότων έχει χρησιμοποιηθεί σε πολλές πραγματικές εφαρμογές, που κυμαίνονται από την αναγνώριση ανθρώπινης δραστηριότητας [8], την διαχείριση αστικών μεταφορών και τη θαλάσσια επιτήρηση [3] έως την αναγνώριση επιθέσεων σε κόμβους δικτύων υπολογιστών [34] και την ανίχνευση απάτης με πιστωτικές κάρτες [2]. Για παράδειγμα, στην αναγνώριση ανθρώπινης δραστηριότητας, ένα σύστημα CER συλλέγει «απλά, παραγόμενα γεγονότα» (simple derived events – SDEs) σχετικά με τη θέση των ανθρώπων και των αντικειμένων στο χώρο, τον προσανατολισμό τους και την συμπεριφορά τους (π.χ. «περπάτημα», «αδρανής», κ.λπ.) και συμπεραίνει σύνθετα γεγονότα (complex events – CEs) που σχετίζονται με τις αλληλεπιδράσεις τους, όπως τον καβγά μεταξύ δύο ατόμων ή την εγκατάληψη ενός αντικειμένου χωρίς επίβλεψη από ένα άτομο [9].

Ωστόσο, σε διάφορες εφαρμογές CER, μερικά SDEs μπορεί να είναι λανθασμένα λόγω δυσλειτουργίας κάποιου αισθητήρα εισόδου ή κάποια καθυστερήσης στη μετάδοση των γεγονότων. Ως εκ τούτου, το σύστημα CER θα πρέπει να εμπιστεύεται τα γεγονότα εισόδου μόνο σε κάποιο βαθμό. Επομένως, εισάγεται αβεβαιότητα στην εισερχόμενη ροή γεγονότων, η οποία μεταφέρεται στο σύστημα CER αφού έχει επισυναπτεί μία τιμή πιθανότητας και μία χρονοσφραγίδα σε κάθε SDE. Επιπλέον, ο τομέας που εξετάζεται μπορεί να είναι δύσκολο να μοντελοποιηθεί με ακρίβεια, οδηγώντας έτσι σε αβεβαιότητα στη διαδικασία συμπερασμού σύνθετων γεγονότων του συστήματος. Αυτές οι παρατηρήσεις υποδηλώνουν την ανάγκη για συστήματα Πιθανοτικής Αναγνώρισης Σύνθετων Γεγονότων (Probilistic Complex Event Recognition), τα οποία είναι ανθεκτικά στους τύπους αβεβαιότητας που αναφέρονται παραπάνω. Μια εκτεταμένη ανάλυση τέτοιων συστημάτων μπορεί να βρεθεί εδώ [5].

Πολλά Πιθανοτικά συστήματα CER [4] [63] [64] χρησιμοποιούν αναγνώριση γεγονότων που βασίζεται σε χρονικές στιγμές. Αυτό σημαίνει πως η έξοδος τους αποτελείται από όλα τα σύνθετα γεγονότα που συνέβησαν, συνοδευμένα από τις χρονικές στιγμές της αναγνώρισης τους. Εντούτοις, μια στιγμαία ένδειξη ενός συμβάντος μπορεί να οδηγήσει σε λανθασμένη ανίχνευση καθώς είναι ευαίσθητη σε βλάβες αισθητήρων που εμφανίζονται σε μικρά χρονικά παράθυρα, σε λανθασμένο ταίριασμα κάποιου χρονικού προτύπου ενός σύνθετου γεγονότος και σε θορυβώδεις χώρους

ανίχνευσης γεγονότων. Αυτές οι ανακρίβειες υποδεικνύουν την ανάγκη επέκτασης τέτοιων Πιθανοτικών συστημάτων CER προκειμένου να υποστηρίζουν την αναγνώριση σύνθετων γεγονότων που βασίζεται σε Χρονικά Διαστήματα (Temporal Intervals). Στην εργασία [10], παρουσιάζεται ο αλγόριθμος PIEC, ο οποίος λαμβάνει μια σειρά από πιθανολογικές ενδείξεις ενός CE που βασίζονται σε χρονικές στιγμές και καταρτίζει έναν κατάλογο χρονικών διαστημάτων για τα οποία συμβαίνει το συγκεκριμένο CE. Έτσι προκύπτει ένα σύστημα που κατευθύνεται προς την Πιθανοτική Αναγνώριση Σύνθετων Γεγονότων βασισμένη σε Χρονικά Διαστήματα (Inteval-based Probabilistic CER). Αυτή η μετάβαση επιτυγχάνεται με τον ορισμό της έννοιας του μέγιστου πιθανοτικού διαστήματος (probabilistic maximal interval) και τη χρήση ενός αλγόριθμου γραμμικού χρόνου για την εξαγωγή αυτών των διαστημάτων από δεδομένα αναγνώρισης συμβάντων ανά χρονική στιγμή.

1.1 Κίνητρο

Σκοπός αυτής της εργασίας είναι η επέκταση του αλγορίθμου PIEC ώστε να λειτουργεί σε περιβάλλον ροής δεδομένων. Μια σημαντική προδιαγραφή του PIEC είναι το γεγονός ότι λειτουργεί σε μια στατική ομάδα από πιθανότητες εμφάνισης ενός CE για την οποία συμπεραίνει μια λίστα μέγιστων πιθανοτικών διαστημάτων. Ωστόσο, μια πραγματική εφαρμογή απαιτεί προσαρμοστικούς αλγόριθμους οι οποίοι μπορούν να λειτουργήσουν σε μεγάλες ροές δεδομένων και είναι ανθεκτικοί στην εμφάνιση επιπλέον δεδομένων στην είσοδο μετά τη διαδικασία παραγωγής διαστημάτων. Επομένως, είναι σημαντικό να επεκταθεί ο αλγόριθμος PIEC ώστε να προσομοιώνει ένα online σύστημα το οποίο δεν αποθηκεύει ολόκληρη την είσοδο αλλά λειτουργεί με περιορισμένη μνήμη εργασίας.
Σε αυτό το πλαίσιο, παρουσιάζουμε τον αλγόριθμο PIEC Ροής Δεδομένων (Streaming PIEC ή sPIEC), ο οποίος είναι σε θέση να αντιμετωπίζει ροές από πιθανότητες εμφάνισης CE αντί για στατικές ομάδες δεδομένων. Αυτός ο αλγόριθμος χρησιμοποιεί χρονικά παράθυρα για τα οποία εκτελείται χωριστά και συγκολλά τα προκύπτοντα διαστήματα κάθε εκτέλεσης. Αυτή η μέθοδος υποδηλώνει ανθεκτικότητα σε πρόσθετα δεδομένα τα οποία θα μπορούσαν να διευθετηθούν σε ένα χρονικό παράθυρο και να υποστούν επεξεργασία ανεξάρτητα. Αντιθέτως, ο PIEC δεν παρουσιάζει αυτή την ευελιξία και θα πρέπει να ξαναεκτελεστεί για ολόκληρη την στατική ομάδα δεδομένων μαζί με τις νέες αναγνωρίσεις για να εξάγει τα σωστά διαστήματα.

1.2 Συνεισφορά

Συνοψίζουμε τη συμβολή αυτής της εργασίας ως εξής:

- Επεκτείνουμε τον αλγόριθμο PIEC εισάγοντας τις απαραίτητες λειτουργίες και δομές δεδομένων που επιτρέπουν τη λειτουργία του σε περιβάλλον ροής δεδομένων.
- Περιγράφουμε λεπτομερώς τις λειτουργίες του sPIEC και παρέχουμε αποδείξεις για την πολυπλοκότητα και την ορθότητα του.
- Εξετάζουμε θεωρητικά την απόδοση του sPIEC σε ένα πλαίσιο οριοθετημένης μνήμης και προτείνουμε μεθόδους για την αποτελεσματική συντήρηση της μνήμης αυτής.
- Αξιολογούμε τον αλγόριθμο sPIEC σε ένα σύνολο δεδομένων που χρησιμοποιείται για την αναγνώριση ανθρώπινης δραστηριότητας. Παρέχουμε πειραματικά αποτελέσματα τα οποία αντικατοπτρίζουν την απόδοση του αλγορίθμου σε σύγκριση με τον αρχικό αλγόριθμο PIEC και το ground truth του dataset.

1.3 Διάρθρωση εργασίας

Το υπόλοιπο της εργασίας οργανώνεται ως εξής. Στο Κεφάλαιο 2 παρουσιάζεται το Επιστημονικό Υπόβαθρο της παρούσας εργασίας, συζητώντας για τη διάλεκτο «Λογισμός Γεγονότων» (Event Calculus – EC), μία πιθανολογική επέκταση της (Prob-EC) και τον αρχικό αλγόριθμο PIEC που παρουσιάζεται στην [10]. Το κεφάλαιο 3 περιγράφει Σχετικές Εργασίες που αφορούν φορμαλισμούς CER και σχετικά Πιθανοτικά συστήματα. Τα κεφάλαια 4 και 5 εισάγουν τον αλγόριθμο Streaming PIEC και την ανάλυση του σε περιβάλλον περιορισμένης μνήμης, αντίστοιχα. Τέλος, το κεφάλαιο 6 περιλαμβάνει την πειραματική αξιολόγηση του sPIEC και το κεφάλαιο 7 συνοψίζει το έργο μας και συζητά πιθανές επεκτάσεις του αλγορίθμου.

Κεφάλαιο 2

Επιστημονικό Υπόβαθρο

2.1 Λογισμός Γεγονότων

Ο Λογισμός Γεγονότων είναι μία λογική γλώσσα κατηγορημάτων πρώτης τάξης για την αναπαράσταση και τον συμπερασμό που σχετίζεται με γεγονότα και με τα αποτελέσματά τους. Πιο συγκεκριμένα, αυτά τα γεγονότα μπορεί να προκαλέσουν αλλαγές στην τιμή ορισμένων ποσοτήτων που ονομάζουνται fluents. Ένα fluent F είναι μια ποσότητα που επιτρέπεται να έχει διαφορετικές τιμές σε διαφορετικές χρονικές στιγμές. Ο όρος F = V σηματοδοτεί ότι το fluent F έχει την τιμή V. Ανεπίσημα, το F = V ισχύει σε μια συγκεκριμένη χρονική στιγμή εάν ο F = V έχει «εκκινήσει» από ένα συμβάν σε κάποιο προγενέστερο χρονικό σημείο και δεν έχει «τερματιστεί» από ένα άλλο συμβάν εν τω μεταξύ. Αυτή η συνέπεια της τιμής ενός fluent ανάμεσα σε ένα κατάλληλο γεγονός έναρξης και το αντίστοιχο γεγονός τερματισμού εκφράζει τον νόμο της αδράνειας.
Ο Λογισμός Γεγονότων έχει χρησιμοποιηθεί στο παρελθόν για την αναγνώριση γεγονότων, εκφράζοντας τους κανόνες σύμφωνα με τους οποίους μπορούν να συναχθούν γεγονότα υψηλού επιπέδου από γεγονότα χαμηλού επιπέδου. Επειδή το σύνολο δεδομένων που χρησιμοποιείται σε αυτή την εργασία αφορά την αναγνώριση ανθρώπινης δραστηριότητας, τα γεγονότα χαμηλού επιπέδου αντιστοιχούν σε ένα σύνολο δραστηριοτήτων που λαμβάνουν χώρα σε σύντομο χρονικό διάστημα, όπως «περπάτημα», «τρέξιμο» και «αδρανής», ενώ τα γεγονότα υψηλού επιπέδου αντιστοιχούν σε δραστηριότητες μεγαλύτερης διάρκειας και πολυπλοκότητας, όπως ο «καβγάς» και η «συνάντηση». Θα αναφέρουμε τις πρώτες ως «βραχυπρόθεσμες δραστηριότητες» (Short Term Activities – STA) και τις τελευταίες ως «μακροπρόθεσμες δραστηριότητες» (Long Term Activities – LTA).
Μία διάλεκτος του Λογισμού Γεγονότων - η Crisp-EC - έχει χρησιμοποιηθεί για την ανίχνευση LTA από STA. Η Crisp-EC περιλαμβάνει γεγονότα, fluents, ένα γραμμικό μοντέλο χρόνου που αποτελείται από ακέραιους αριθμούς και ένα σύνολο βασικών αξιωμάτων, ένα από τα οποία είναι ο προαναφερόμενος νόμος της αδράνειας. Μια περιγραφή συμβάντος στην Crisp-EC περιλαμβάνει κανόνες που καθορίζουν, μεταξύ άλλων, τις πραγματοποιήσεις γεγονότων (με χρήση του κατηγορήματος happensAt), τις συνέπειες των γεγονότων (με τη χρήση των κατηγορημάτων initiatedAt και terminatedAt) και τις τιμές των fluents (με τη χρήση των κατηγορημάτων initially και holdsAt). Ορισμένα από αυτά τα κατηγορήματα παρουσιάζονται στον πίνακα 2.1. Οι μεταβλητές είναι ολικά ποσοτικοποιημένες και ξεκινάνε με κεφαλαίο γράμμα, εκτός αν ορίζεται διαφορετικά. Τα κατηγορήματα, οι συναρτήσεις και οι σταθερές αρχίζουν με μικρό γράμμα.
Μερικά από αυτά τα κατηγορήματα ορίζονται από ένα σύνολο κανόνων ανεξάρτητων από τον τομέα

Κατηγόρημα	**Σημασία**
happensAt$(E,\ T)$	Το γεγονός E συμβαίνει τη χρονική στιγμή T
initially$(F = V)$	Η τιμή του fluent F είναι V τη χρονική στιγμή 0
holdsAt$(F = V,\ T)$	Η τιμή του fluent F είναι V τη χρονική στιγμή T
initiatedAt$(F = V,\ T)$	Τη χρονική στιγμή T μια χρονική περίοδος όπου $F = V$ εκκινεί.
terminatedAt$(F = V,\ T)$	Τη χρονική στιγμή T μία χρονική περίοδος κατά την οποία $F = V$ τερματίζει

Πίνακας 2.1: Τα κυριότερα κατηγορήματα του Λογισμού Γεγονότων.

της εκάστοτε εφαρμογής, οι οποίοι περιλαμβάνονται στα αξιώματα της διαλέκτου EC. Οι κανόνες που είναι ανεξάρτητοι από τον τομέα εφαρμογής για το κατηγόρημα holdsAt μπορούν να γραφτούν με την ακόλουθη μορφή:

$$\begin{aligned}&\text{holdsAt}(F = V,\ T) \leftarrow \\ &\quad \text{initially}(F = V), \\ &\quad \text{not broken}(F = V,\ 0,\ T).\end{aligned} \tag{2.1}$$

$$\begin{aligned}&\text{holdsAt}(F = V,\ T) \leftarrow \\ &\quad \text{initiatedAt}(F = V,\ T_s),\ T_s < T, \\ &\quad \text{not broken}(F = V,\ T_s,\ T).\end{aligned} \tag{2.2}$$

$$\begin{aligned}&\text{broken}(F = V,\ T_s,\ T) \leftarrow \\ &\quad \text{terminatedAt}(F = V,\ T_f),\ T_s < T_f < T.\end{aligned} \tag{2.3}$$

$$\begin{aligned}&\text{broken}(F = V,\ T_s,\ T) \leftarrow \\ &\quad \text{initiatedAt}(F = V',\ T_f),\ V \neq V',\ T_s < T_f < T.\end{aligned} \tag{2.4}$$

Οι παραπάνω κανόνες ορίζουν επακριβώς τις συνθήκες σύμφωνα με τις οποίες ένα fluent F έχει μια τιμή V σε μία χρονική τιμή T. Πιο συγκεκριμένα, οι κανόνες 2.1 και 2.2 υποδηλώνουν ότι το fluent F έχει την τιμή V την στιγμή T αν του είχε ανατεθεί αυτή η τιμή σε μία προηγούμενη χρονική στιγμή και αυτή η ανάθεση δεν έχει γίνει broken σε ένα ενδιάμεσο χρονικό σημείο. Οι κανόνες 2.3 και 2.4 είναι συμπληρωματικοί και ορίζουν το κατηγόρημα broken. Ένα fluent μπορεί να «σπάσει» μεταξύ δύο χρονικών στιγμών είτε ρητά με ένα κατηγόρημα terminatedAt (Κανόνας 2.3) είτε σιωπηρώς εκκινώντας το ίδιο fluent με μια διαφορετική τιμή με το κατηγόρημα initiatedAt (Κανόνας 2.4).

Είναι σημαντικό να σημειώσουμε ότι οι κανόνες για τα κατηγορήματα initially, initiatedAt και terminatedAt δεν εμφανίζοναι στα παραπάνω αξιώματα. Αυτό οφείλεται στο γεγονός ότι τα αξιώματα που αντιστοιχούν στα κατηγορήματα έναρξης και τερματισμού εξαρτώνται από τον τομέα της εφαρμογής. Έτσι λοιπόν, τα γεγονότα και οι συνθήκες που πρέπει να πληρούνται για την εκκίνηση ή τον τερματισμό ενός LTA εξαρτώνται από το πλαίσιο της υλοποίησης.

Μία συνηθισμένη μορφή ενός κανόνα initiatedAt είναι η ακόλουθη:

$$\begin{aligned}&\text{initiatedAt}(F = V,\ T) \leftarrow \\ &\quad \text{happensAt}(E, T), \\ &\quad \text{Conditions}[T].\end{aligned} \tag{2.5}$$

Ο παραπάνω κανόνας δηλώνει ότι ένα fluent F εκκινεί τη χρονική στιγμή T αν προκύψει ένα συγκεκριμένο γεγονός E την ίδια χρονική στιγμή και ικανοποιούνται κάποιες πρόσθετες χρονικές

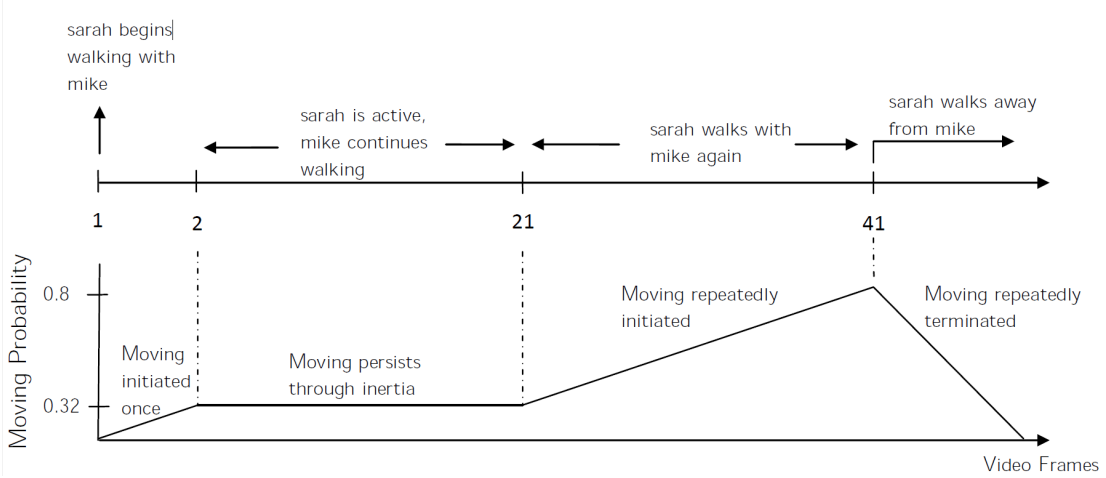
συνθήκες. Η επιλογή του συμβάντος και των συνθηκών είναι προσαρμοσμένες για το συγκεκριμένο τομέα.
Προκειμένου να αποφευχθεί η σύγχυση σε μελλοντικά παραδείγματα, είναι σημαντικό να σημειωθεί ότι το κατηγόρημα initiatedAt($F = V$, T) δεν υποδηλώνει απαραίτητα ότι $F \neq V$ τη χρονική στιγμή T. Ομοίως, το κατηγόρημα terminatedAt($F = V$, T) δεν υποδηλώνει απαραίτητα ότι $F = V$ τη στιγμή T.

2.2 Πιθανοτικός Λογισμός Γεγονότων

Ο Prob-EC είναι μια πιθανολογική εκδοχή του Λογισμού Γεγονότων που έχει υλοποιηθεί σε ProbLog [7]. Είναι μια επέκταση της διάλεκτος Crisp-EC, η οποία προσθέτει πιθανότητες στις αναγνωρίσεις των LTA [4]. Ο κύριος σκοπός της δημιουργίας της ήταν η αντιμετώπιση των διαφόρων τύπων αβεβαιότητας που υπάρχουν στην αναγνώριση δραστηριοτήτων, όπως η εσφαλμένη ανίχνευση STA [8]. Έτσι, ο Prob-EC είναι σε θέση να αναθέτει τιμές πιθανότητας στα κατηγορήματα holdsAt των fluents (LTA) δεδομένων πιθανοτικών κατηγορημάτων happensAt διαφόρων συμβάντων (STA).
Η ProbLog επιτρέπει πιθανοτικές εκφράσεις $:p :: f$ που υποδηλώνουν ότι το κατηγόρημα f, το οποίο μπορεί να εξαρτάται από ελεύθερες μεταβλητές, έχει πιθανότητα p για οποιαδήποτε από τις πιθανές αναθέσεις του. Όλα αυτά τα κατηγορήματα αντιπροσωπεύουν ανεξάρτητες τυχαίες μεταβλητές. Έτσι, ένας κανόνας που ορίζεται ως ένας συνδυασμός k πιθανοτικών προτάσεων, έχει πιθανότητα ίση με το γινόμενο των πιθανοτήτων των προτάσεων αυτών. Επιπλέον, για τα κατηγορήματα που εμφανίζονται στην κεφαλή περισσότερων του ενός κανόνα, η πιθανότητα τους είναι ίση με την πιθανότητα της διάζευξης των κανόνων αυτών.
Έτσι, η πιθανότητα ότι το ερώτημα q πραγματοποιείται σε ένα πρόγραμμα ProbLog υπολογίζεται ως εξής:

$$P_s(q) = P(\bigvee_{e \in Proofs(q)} \bigwedge_{f_i \in e} f_i \ ) \tag{2.6}$$

Έτσι, για να υπολογίσει την πιθανότητα ενός κατηγορήματος, η ProbLog υπολογίζει την πιθανότητα του κάθε κανόνα που έχει το συγκεκριμένο κατηγόρημα στην κεφαλή του και, κατά συνέπεια, τις πιθανότητες όλων των προτάσεων που εμφανίζονται σε αυτούς τους κανόνες. Η αβεβαιότητα αυτών των προτάσεων πηγάζει από τα προκαθορισμένα happensAt(E, T) τα οποία είναι πιθανοτικά λόγω του θορύβου των STA στην είσοδο. Πιο συγκεκριμένα, η ProbLog υπολογίζει αποτελεσματικά αυτή την πιθανότητα χρησιμοποιώντας Διαγράμματα Δυαδικής Απόφασης [6].



Σχήμα 2.1: Πιθανοτική Αναγνώριση Γεγονότων με τον Prob-EC, [4]

Το σχήμα 2.1 παρουσιάζει ένα παράδειγμα της συλλογιστικής διαδικασίας του Prob-EC. Ας υποθέσουμε ότι υπάρχουν δύο άνθρωποι, ο Mike και η Sarah, και αρχίζουν να κινούνται μαζί τη χρονική στιγμή 1. Μία στιγμή αργότερα, η Sarah σταματάει να περπατάει και γίνεται «ενεργή». Η Sarah παραμένει «ενεργή» μέχρι τη χρονική στιγμή 21, όπου ξεκινάει να περπατάει και πάλι με τον Mike. Στη συνέχεια, τη χρονική στιγμή 41 η Sarah ξεκινάει να απομακρύνεται από τον Mike. Για να αναλύσουμε αυτό το παράδειγμα, παρουσιάζουμε πρώτα τον (μερικό) ορισμό της δραστηριότητας συμβάδισης (moving) ως εξής:

$$\begin{aligned}&\mathsf{initiatedAt}(moving(P_1,P_2)=\mathsf{true},\ T)\leftarrow\\&\quad\mathsf{happensAt}(walking(P_1),\ T),\\&\quad\mathsf{happensAt}(walking(P_2),\ T),\\&\quad\mathsf{holdsAt}(close(P_1,P_2)=\mathsf{true},\ T),\\&\quad\mathsf{holdsAt}(similarOrientation(P_1,P_2)=\mathsf{true},\ T).\end{aligned}\tag{2.7}$$

$$\begin{aligned}&\mathsf{terminatedAt}(moving(P_1,P_2)=\mathsf{true},\ T)\leftarrow\\&\quad\mathsf{happensAt}(walking(P_1),\ T),\\&\quad\mathsf{holdsAt}(close(P_1,P_2)=\mathsf{false},\ T).\end{aligned}\tag{2.8}$$

$$\begin{aligned}&\mathsf{terminatedAt}(moving(P_1,P_2)=\mathsf{true},\ T)\leftarrow\\&\quad\mathsf{happensAt}(active(P_1),\ T),\\&\quad\mathsf{happensAt}(active(P_2),\ T).\end{aligned}\tag{2.9}$$

$$\begin{aligned}&\mathsf{terminatedAt}(moving(P_1,P_2)=\mathsf{true},\ T)\leftarrow\\&\quad\mathsf{happensAt}(running(P_1),\ T).\end{aligned}\tag{2.10}$$

Ο κανόνας 2.7 υποδηλώνει ότι γεγονός $moving(P_1, P_2)$ εκκινεί τη χρονική στιγμή T αν την ίδια χρονική στιγμή εμφανίζεται το συμβάν walking και για τα δύο άτομα, είναι αρκετά κοντά και έχουν τον ίδιο προσανατολισμό. Σημειώνουμε ότι οι κανόνες initiatedAt ενεργοποιούνται σε κάθε χρονική στιγμή όπου όλα τα υποκείμενα γεγονότα ικανοποιούνται με μια θετική πιθανότητα. Επομένως, είναι σύνηθες τέτοιοι κανόνες να ικανοποιούνται σε συνεχόμενες χρονικές στιγμές, αυξάνοντας την

πιθανότητα του υποκείμενου LTA κάθε φορά.
Οι κανόνες 2.8, 2.9 και 2.10 διατυπώνουν από ένα σύνολο γεγονότων των οποίων η εμφάνιση με θετική πιθανότητα πυροδοτεί τον τερματισμό της δραστηριότητας $moving(P_1, P_2)$. Για παράδειγμα, ο κανόνας 2.10 υποδεικνύει ότι δύο άτομα σταματούν να βαδίζουν μαζί όταν ένα από αυτά αρχίζει να τρέχει. Κατά συνέπεια, στους κανόνες τερματισμού αποδίδεται μια τιμή πιθανότητας βασισμένη στις πιθανότητες των γεγονότων που τους πυροδότησαν. Πολλαπλοί τερματισμοί ενός γεγονότος για συνεχόμενες χρονικές στιγμές προκαλούν την πτώση της πιθανότητάς του.
Επιστρέφοντας στο σχήμα 2.1, η γραφική παράσταση της πιθανότητας του γεγονότος συμβάδισης αυξάνεται κατά τη διάρκεια διαστημάτων για τα οποία συμβαίνουν διαδοχικές εκκινήσεις της δραστηριότητας, μειώνεται ενώ εμφανίζονται διαδοχικοί τερματισμοί και παραμένει σταθερή όσο δεν εμφανίζονται ούτε εκκινήσεις ούτε τερματισμοί. Στην τελευταία περίπτωση, η πιθανότητα της δραστηριότητας παραμένει σταθερή λόγω του νόμου της αδράνειας.
Θεωρούμε τη ροή εισόδου αποτελούμενη από STA που φαίνεται παρακάτω:

$$
\begin{aligned}
&0.70 :: \mathsf{happensAt}(walking(mike),\ 1).\\
&0.46 :: \mathsf{happensAt}(walking(sarah),\ 1).\\
&\ldots\\
&0.73 :: \mathsf{happensAt}(walking(mike),\ 2).\\
&0.55 :: \mathsf{happensAt}(active(sarah),\ 2).\\
&\ldots\\
&0.69 :: \mathsf{happensAt}(walking(mike),\ 21).\\
&0.58 :: \mathsf{happensAt}(walking(sarah),\ 21).\\
&\ldots\\
&0.18 :: \mathsf{happensAt}(inactive(mike),\ 41).\\
&0.32 :: \mathsf{happensAt}(walking(sarah),\ 41).
\end{aligned}
$$

Υποθέτοντας ότι ο προσανατολισμός και οι συντεταγμένες των αντικειμένων υπολογίζονται με βεβαιότητα, διαπιστώνουμε ότι μια εκκίνηση για τη δραστηριότητα *moving*(mike, sarah) πραγματοποιείται τη χρονική στιγμή *1*, και αντιστοιχεί στον κανόνα 2.7. Επειδή τα υποκείμενα γεγονότα έχουν πιθανότητες *0.70* και *0.46* αντίστοιχα, η πιθανότητα του $\mathsf{initiatedAt}(moving(\text{mike, sarah}), 1)$, που υπολογίζεται ως το γινόμενο των πιθανοτήτων κάθε γεγονότος που βρίσκεται στο σώμα του κανόνα, είναι ίση με *0.322*. Έτσι, παρατηρούμε μία αύξηση στην πιθανότητα *moving*(mike, sarah) από *0* έως *0.322* από την χρονική στιγμή *1* έως τη στιγμή *2*.
Επειδή δεν πληρούνται κανόνες εκκίνησης ή τερματισμού για την δραστηριότητα *moving*(mike, sarah) μέχρι τη χρονική στιγμή *21*, η πιθανότητα της δραστηριότητας συνεχίζει να έχει την τιμή *0.322*. Όμως, τη στιγμή *21* η Sarah ξεκινάει πάλι να βαδίζει. Επομένως, ο κανόνας 2.7 ικανοποιείται και αυξάνεται περαιτέρω η πιθανότητα της δραστηριότητας *moving*(mike, sarah). Εφόσον έχουν πραγματοποιηθεί μέχρι στιγμής περισσότερες από μία εκκινήσης αυτής της δραστηριότητας, ο Prob-EC τις λαμβάνει όλες υπόψη κατά τον υπολογισμό της πιθανότητας της *moving*(mike, sarah). Πιο συγκεκριμένα, ο Prob-EC εκφράζει την πιθανότητα της *moving*(mike, sarah) ως τη διάζευξη των γεγονότων εκκίνησης της δραστηριότητας τις χρονικές στιγμές *1* και *21*. Έτσι, η πιθανότητα πραγματοποίησης της *moving*(mike, sarah) την επακόλουθη χρονική στιγμή υπολογίζεται ως εξής:

$$
\begin{aligned}
P(\mathsf{holdsAt}_{22}) &= P(\mathsf{initiatedAt}_1 \vee \mathsf{initiatedAt}_{21}) \\
&= P(\mathsf{initiatedAt}_1) \vee P(\mathsf{initiatedAt}_{21}) \\
&= P(\mathsf{initiatedAt}_1) + P(\mathsf{initiatedAt}_{21}) - P(\mathsf{initiatedAt}_1) \times P(\mathsf{initiatedAt}_{21}) \\
&= 0.7 \times 0.46 + 0.69 \times 0.58 - 0.7 \times 0.46 \times 0.69 \times 0.58 = 0.593
\end{aligned} \tag{2.11}
$$

Παρατηρούμε ότι η δραστηριότητα *moving*(mike, sarah) εκκινείται επανειλημμένα μέχρι τη στιγμή *41*, καθώς η πιθανότητά της αυξάνεται συνεχώς και φτάνει στην τιμή *0.80*. Ωστόσο, τη στιγμή *41* η Sarah αρχίζει να απομακρύνεται από τον Mike, αυξάνοντας την απόσταση μεταξύ τους. Δεδομένου ότι το κατηγόρημα close εξαρτάται μόνο από τις συντεταγμένες των ατόμων, μπορούμε να το συμπεράνουμε με βεβαιότητα. Επομένως, ο τερματισμός της *moving*(mike, sarah) που πραγματοποιείται τη στιγμή *41* και αντιστοιχεί στον κανόνα 2.8 έχει πιθανότητα ίση με το γεγονός της απομάκρυνση της Sarah. Επομένως:

$$
P(\mathsf{terminatedAt}_{41}) = P(\mathsf{happensAt}(walking(sarah), 41)) = 0.32 \tag{2.12}
$$

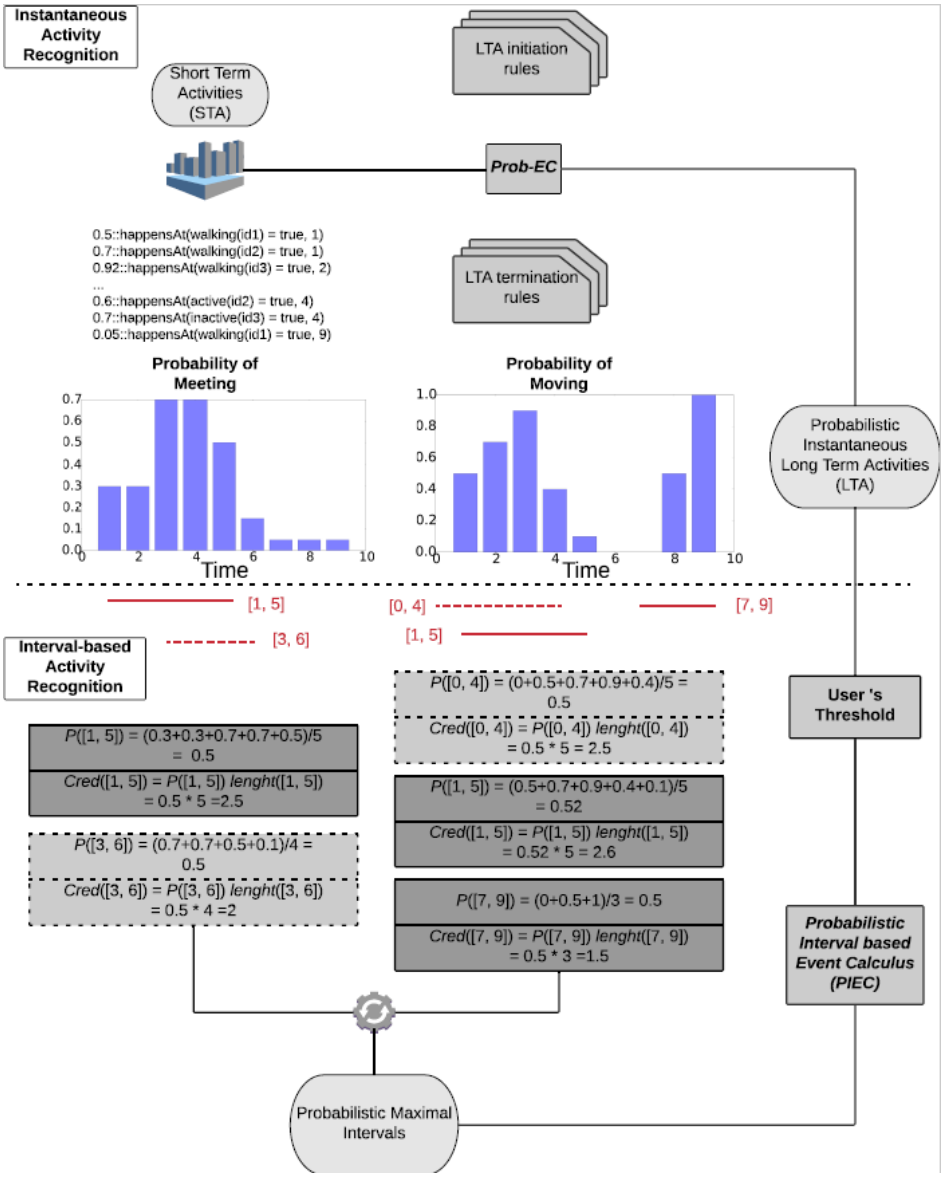
Είναι λογικό η δραστηριότητα *moving*(mike, sarah) να πραγματοποιείται εάν είχε εκκινηθεί σε κάποια προγούμενη χρονική στιγμή και δεν έχει τερματιστεί από τότε. Έτσι, ο Prob-EC υπολογίζει την πιθανότητα του κατηγορήματος $\mathsf{holdsAt}(moving(\text{mike, sarah}) = \text{true}, 42)$ ως εξής:

$$
\begin{aligned}
P(\mathsf{holdsAt}_{42}) &= P(\mathsf{holdsAt}_{41}) * (1 - P(\mathsf{terminatedAt}_{41}) \\
&= 0.8 * (1 - 0.32) \\
&= 0.544
\end{aligned} \tag{2.13}
$$

Παρομοία με την περίπτωση των πολλαπλών εκκινήσεων, οι συνεχείς τερματισμοί της *moving*(mike, sarah) από τη χρονική στιγμή *41* και μετά προκαλούν τη συνεχή μείωση της πιθανότητας αυτής της LTA μέχρι να φτάσει στο *0*.

2.3 Ένας αλγόριθμος Πιθανοτικού Λογισμού Γεγονότων βασισμένος σε χρονικά διαστήματα

Ο Πιθανοτικός Λογισμός Γεγονότων με βάση τα χρονικά διαστήματα (Probabilistic Interval-based Event Calculus – PIEC) περιλαμβάνει έναν αλγόριθμο γραμμικού χρόνου που χρησιμοποιείται για τον υπολογισμό μέγιστων πιθανοτικών διαστημάτων δεδομένου ενός πίνακα από πιθανοτικές στιγμιαίες αναγνωρίσεις γεγονότων [10]. Με βάση τον PIEC, θα παρουσιάσουμε στην συνέχεια τον αλγόριθμο PIEC ροής δεδομένων(Streaming PIEC – sPIEC) που χειρίζεται την είσοδο από στιγμιαίες πιθανότητες σε παράθυρα σταθερού μήκους.
Σε αυτή την ενότητα, εξετάζουμε συνοπτικά τις ιδιότητες του PIEC, επιδεικνύουμε τον ψευδοκώδικα του αλγορίθμου και ένα παράδειγμα της εκτέλεσης του.



Σχήμα 2.2: Πιθανοτική αναγνώριση γεγονότων με χρήση χρονικών διαστημάτων, [10]

Αρχικά (εικόνες πάνω από την διακεκομμένη γραμμή), ο Prob-EC υπολογίζει τις στιγμιαίες πιθανότητες των LTA, όπως συνάντηση (meeting) και συμβάδισμα(moving), δεδομένων των πιθανοτικών δραστηριοτήτων χαμηλού επιπέδου STA, όπως «βάδισμα» (walking), «ενεργή κίνηση των μελών του σώματος» (active) και «ακινησία» (inactive). Στη συνέχεια (κάτω από τη διακεκομμένη γραμμή), η PIEC υπολογίζει μέγιστα πιθανοτικά διαστήματα για τις LTA, υποθέτωντας ένα κατώφλι πιθανότητας ορισμένο από τον χρήστη (0.5 σε αυτό το παράδειγμα). Τα διαστήματα αυτά υποδεικνύονται με τις κόκκινες (διακεκομμένες) γραμμές κάτω από τις κατανομές των στιγμιαίων πιθανοτήτων. Ο υπολογισμός της πιθανότητας αλλά και της «αξιοπιστίας» για κάθε διάστημα παρουσιάζεται αναλυτικά στα πλαίσια κάτω από τις κόκκινες γραμμές.

2.3.1 Επισκόπηση

Η ανάγκη για έναν αλγόριθμο βασισμένο σε διαστήματα προκύπτει από την αναξιόπιστια της αναγνώρισης γεγονότων με βάση χρονικές στιγμές. Η αναγνώριση ενός κατηγορήματος holdsAt(LTA=true, T) μπορεί να οφείλεται σε δυσλειτουργία ενός αισθητήρα ή σε λανθασμένη εμφάνιση ενός μοτίβου αναγνώρισης [10]. Επιπλέον, σ΄ ένα θορυβώδες περιβάλλον η ανίχνευση συμβάντων συνοδεύεται από μια τιμή πιθανότητας, προσθέτοντας μεγαλύτερη αβεβαιότητα στη διαδικασία [5]. Μια προσέγγιση βασισμένη σε χρονικά διαστήματα παρέχει μια πιο σταθερή αναγνώριση, καθώς είναι πιο ανθεκτική σε χρονικά σύντομες δυσλειτουργίες.
Το σχήμα (2.2) παρουσιάζει τη διαδικασία αναγνώρισης μίας λίστας από μέγιστα πιθανοτικά διαστήματα που αντιστοιχούν σε εμφανίσεις των κατάλληλων LTA από πιθανοτικές STA. Καταρχάς, ο Prob-EC χρησιμοποιείται για να υπολογίσει τις πιθανότητες των LTA από την είσοδο στιγμιαία πιθανοτικών STA. Η επόμενη φάση περιλαμβάνει την αναγνώριση γεγονότων με βάση χρονικά διαστήματα. Με βάση ένα πιθανοτικό κατώφλι (probability threshold) που καθορίζεται από το χρήστη, ο PIEC υπολογίζει όλα τα μέγιστα χρονικά διαστήματα, μέσα στα οποια είναι πιθανό να πραγματοποιείται μία δραστηριότητα. Στη συνέχεια, τα αλληλοεπικαλυπτόμενα χρονικά διαστήματα που εντοπίστηκαν ξεδιαλύνονται εφαρμόζοντας ένα κριτήριο αξιοπιστίας.
Η είσοδος του PIEC περιλαμβάνει έναν πίνακα από στιγμιαίες χρονικές πιθανότητες (λίστα A) μεγέθους n και ενός πιθανοτικού ορίου (T). Βασισμένος σε αυτά τα δεδομένα, ο PIEC κατασκευάζει τις ακόλουθες λίστες:

- Η λίστα L περιλαμβάνει κάθε στοιχείο της λίστας A μειωμένο κατά το πιθανοτικό κατώφλι T.
- Η λίστα *prefix* περιέχει τα συσσωρευτικά αθροίσματα της λίστα L.
- Η λίστα *dp* κατασκευάζεται διασχίζοντας την λίστα *prefix* σε αντίστροφη σειρά. Πιο συγκεκριμένα, $dp[i] = \max_{i \leq j \leq n}(prefix[j])$.

Για λόγους πληρότητας, παρουσιάζουμε ορισμένους ορισμούς[10] σχετικά με τα διαστήματα που παράγονται από τον PIEC πριν εξετάσουμε περαιτέρω τον αλγόριθμο.

Ορισμός 1. *Η πιθανότητα του διαστήματος $I_{LTA}=[i,j]$ ενός LTA με $length(I_{LTA})=j-i+1$ χρονικές στιγμές ορίζεται ως*

$$P(I_{LTA}) = \frac{\sum_{k=i}^{j} P(\mathit{holdsAt}(LTA = \mathit{true},\ k))}{length(I_{LTA})} \ .$$

Με άλλα λόγια, η πιθανότητα ενός διαστήματος είναι ίση με τον μέσο όρο των πιθανοτήτων των χρονικών σημείων που περιέχει.
Μια βασική έννοια του PIEC είναι αυτή του μέγιστου πιθανοτικού διαστήματος:

Ορισμός 2. *Ένα μέγιστο πιθανοτικό διάστημα $I_{LTA}=[i,j]$ ενός LTA είναι ένα διάστημα τέτοιο ώστε, δεδομένου κάποιου πιθανοτικού κατωφλιού $\mathcal{T} \in [0,1]$, $P(I_{LTA}) \geq \mathcal{T}$, και δεν υπάρχει άλλο διάστημα I'_{LTA} τέτοιο ώστε $P(I'_{LTA}) \geq \mathcal{T}$ και το I_{LTA} να είναι υποδιάστημα του I'_{LTA}.*

Έχει αποδειχθεί στο [12] ότι ένα διάστημα I ικανοποιεί την συνθήκη $P(I) \geq T$ αν και μόνο αν το άθροισμα των αντίστοιχων στοιχείων της λίστας L είναι μεγαλύτερο ή ίσο του μηδενός.
Ο PIEC χρησιμοποιεί δύο δείκτες, τον *start* και τον *end*, για να υποδείξει την έναρξη και το

πέρας ενός πιθανού μέγιστου πιθανοτικού διαστήματος, αντίστοιχα. Επιπλέον, χρησιμοποιεί την μεταβλητή *dprange* η οποία ορίζεται ως εξής:

$$dprange[start, end] = \begin{cases} dp[end] - prefix[start-1] & start > 0 \\ dp[end] & start = 0 \end{cases} \quad (2.14)$$

Αυτή η μεταβλητή υποδεικνύει το μέγιστο άθροισμα που μπορεί να υπολογιστεί προσθέτοντας τα στοιχεία της λίστας L από τον δείκτη $start$ έως κάποιο δείκτη $end^* \geq end$ ή:

$$\max_{end \leq end* \leq n}(L[start] + ... + L[end^*]).$$

Έτσι, ένα μη αρνητικό *dprange* για κάποιες τιμές των δεικτών $start$ και end υποδηλώνει ότι υπάρχει κάποια τιμή $end^* \geq end$ για την οποία $\sum_{start \leq i \leq end^*} L[i] \geq 0$ και το $I = [start, end^*]$ είναι, επομένως, ένα πιθανό μέγιστο πιθανοτικό διάστημα. Σε αυτή την περίπτωση, ο PIEC αυξάνει τον δείκτη end μέχρι το *dprange* να μην είναι πλέον μη αρνητικό. Η τιμή του δείκτη $start$ και η τιμή του προτελευταίου δείκτη end αντιστοιχούν σε ένα μέγιστο πιθανοτικό διάστημα.
Εφόσον το *dprange* είναι τώρα αρνητικό, δεν υπάρχει πιθανό μέγιστο πιθανοτικό διάστημα που να εκκινεί από τον δείκτη $start$ και να τερματίζει σε οποιοδήποτε χρονικό σημείο end^*: $end \leq end^* \leq n$. Γι΄ αυτό, ο PIEC αυξάνει τον δείκτη $start$ και να ξαναϋπολογίζει τη μεταβλητή *dprange*.
Επαναλαμβάνοντας τη διαδικασία που περιγράψαμε παραπάνω μέχρις ότου όλα τα στοιχεία της λίστας A να έχουν προσπελαστεί από οποιονδήποτε δείκτη, ο PIEC εξάγει ένα σύνολο που περιέχει όλα τα μέγιστα πιθανοτικά διαστήματα για τη δεδομένη είσοδο. Σημειώνουμε πως ορισμένα από αυτά τα διαστήματα μπορεί να επικαλύπτονται. Στην εργασία [10] έχει προταθεί ένας μηχανισμός για την εύρεση του πιο «αξιόπιστου» διαστήματος μίας περιοχής αλληλεποκαλυπτόμενων πιθανοτικών μέγιστων διαστημάτων και έχει επισυναπτεί στο τέλος του αλγορίθμου. Διάφοροι ορισμοί της αξιοπιστίας ενός διαστήματος έχουν προταθεί στα [10] και [12].

2.3.2 Ψευδοκώδικας

Algorithm 1 Αλγόριθμος PIEC

Είσοδος: Η λίστα A με τις στιγμιαίες πιθανότητες ενός συμβάντος και το πιθανοτικό κατώφλι T
Έξοδος: Μία λίστα από μέγιστα πιθανοτικά διαστήματα.
Απαιτεί: Τις λίστες L, *prefix* και *dp*

```
1:  procedure PIEC(List A, threshold T)
2:      start, end ← 0
3:      flag ← false
4:      output ← ∅
5:      dprange[start, end] ← 0
6:      while start < n && end < n do
7:          if start == 0 then
8:              dprange[start, end] ← dp[end]
9:          else
10:             dprange[start, end] ← dp[end] − prefix[start − 1]
11:         if dprange[start, end] ≥ 0 then
12:             if end == n − 1 && start < end then
13:                 add(start, end) to output
14:             if end == start == n − 1 then
15:                 add(start, end) to output
16:             flag ← true
17:             end + +
18:         else
19:             if start < end && flag == true then
20:                 add(start, end) to output
21:             flag ← false
22:             start + +
        return output
```

2.3.3 Παράδειγμα

Σε αυτή την ενότητα, παρουσιάζουμε το παράδειγμα εκτέλεσης του PIEC από την εργασία [10]. Αρχικά περιγράφουμε τις δομές που χρησιμοποιούνται και, έπειτα, την εκτέλεση του αλγορίθμου.

Εισοδος

$$A = [0, 0.5, 0.7, 0.9, 0.4, 0.1, 0, 0, 0.5, 1]$$
$$T = 0.5$$

Λιστες

$Time$	0	1	2	3	4	5	6	7	8	9
A	0	0.5	0.7	0.9	0.4	0.1	0	0	0.5	1
L	-0.5	0	0.2	0.4	-0.1	-0.4	-0.5	-0.5	0	0.5
$prefix$	-0.5	-0.5	-0.3	0.1	0	-0.4	-0.9	-1.4	-1.4	-0.9
dp	0.1	0.1	0.1	0.1	0	-0.4	-0.9	-0.9	-0.9	-0.9

Εκτελεση

Βήμα	**Δείκτες**	**Υπολογισμοί**	**Σχόλια**
1	start=0 end=0	$dprange[0,0] \leftarrow dp[0] = 0.1 \geq 0$ $flag \leftarrow true$ $end++$	Ένα υποψήφιο μέγιστο πιθανοτικό διάστημα βρέθηκε. Ο PIEC θα προσπαθήσει να το επεκτείνει αυξάνοντας τον δείκτη end ώσπου η μεταβλητή $dprange$ να γίνει αρνητική.
2	start=0 end=1	$dprange[0,1] \leftarrow dp[1] = 0.1 \geq 0$ $flag \leftarrow true$ $end++$	-
3	start=0 end=2	$dprange[0,2] \leftarrow dp[2] = 0.1 \geq 0$ $flag \leftarrow true$ $end++$	-
4	start=0 end=3	$dprange[0,3] \leftarrow dp[3] = 0.1 \geq 0$ $flag \leftarrow true$ $end++$	-
5	start=0 end=4	$dprange[0,4] \leftarrow dp[4] = 0.1 \geq 0$ $flag \leftarrow true$ $end++$	-
6	start=0 end=5	$dprange[0,5] \leftarrow dp[5] = -0.4 < 0$ Το διάστημα [0,4] προστίθεται στα αποτελέσματα $flag \leftarrow false$ $start++$	Επειδή η flag αλλάξε σε false,ο PIEC συμπεραίνει ότι το διάστημα που βρέθηκε στην προηγούμενη επανάληψη είναι ένα μέγιστο πιθανοτικό διάστημα και το προσθέτει στη λίστα αποτελεσμάτων του. Επειδή η dprange είναι αρνητική, δεν υπάρχει $end^* > end$ για το οποίο το $[start, end^*]$ να είναι ένα μέγιστο πιθανοτικό διάστημα. Επομένως, ο δείκτης start αυξάνεται.
7	start=1 end=5	$dprange[1,5] \leftarrow dp[5] - prefix[0]$ $= 0.1 \geq 0$ $flag \leftarrow true$ $end++$	-

8	start=1 end=6	$dprange[1,6] \leftarrow dp[6] - prefix[0]$ $= -0.4 < 0$ Το διάστημα [1,5] προστίθεται στα αποτελέσματα $flag \leftarrow false$ $start++$	-
9	start=2 end=6	$dprange[2,6] \leftarrow dp[6] - prefix[1]$ $= -0.4 < 0$ $flag \leftarrow false$ $start++$	-
10	start=3 end=6	$dprange[3,6] \leftarrow dp[6] - prefix[2]$ $= -0.6 < 0$ $flag \leftarrow false$ $start++$	-
11	start=4 end=6	$dprange[4,6] \leftarrow dp[6] - prefix[3]$ $= -1 < 0$ $flag \leftarrow false$ $start++$	-
12	start=5 end=6	$dprange[5,6] \leftarrow dp[6] - prefix[4]$ $= -0.9 < 0$ $flag \leftarrow false$ $start++$	-
13	start=6 end=6	$dprange[6,6] \leftarrow dp[6] - prefix[5]$ $= -0.5 < 0$ $flag \leftarrow false$ $start++$	-
14	start=7 end=6	$dprange[7,6] \leftarrow dp[6] - prefix[6]$ $= 0 \geq 0$ $flag \leftarrow true$ $end++$	-
15	start=7 end=7	$dprange[7,7] \leftarrow dp[7] - prefix[6]$ $= 0 \geq 0$ $flag \leftarrow true$ $end++$	-
16	start=7 end=8	$dprange[7,8] \leftarrow dp[8] - prefix[6]$ $= 0 \geq 0$ $flag \leftarrow true$ $end++$	-
17	start=7 end=9	$dprange[7,9] \leftarrow dp[9] - prefix[6]$ $= 0 \geq 0$ Το διάστημα [7,9] προστίθεται στα αποτελέσματα $flag \leftarrow true$ $end++$	Επειδή ο δείκτης end έχει φτάσει στο τελευταίο χρονικό σημείο και η flag είναι true, το διάστημα [7,9] προστίθεται στα αποτελέσματα (γραμμές 12-13).

Κεφάλαιο 3

Σχετικές Εργασίες

3.1 Ο Λογισμός Γεγονότων και άλλοι σχετικοί φορμαλισμοί

Οι Kowalski και Sergot έχουν προτείνει τον Λογισμό Γεγονότων (Event Calculus – EC) [13] ως μία Χρονική Λογική που επεκτείνει τη λογική πρώτης τάξης προσθέτοντας την έννοια του χρόνου. Συσχετίζεται συχνά με τον Λογισμό Καταστάσεων (Situation Calculus) που παρουσιάστηκε για πρώτη φορά στο [14] από τον McCarthy. Ο Λογισμός Καταστάσεων [14] [15] [16] [18] είναι μια γλώσσα δεύτερης τάξης σχεδιασμένη για να αντιπροσωπεύει δυναμικά μεταβαλλόμενους κόσμους. Αυτές οι αλλαγές προκαλούνται από *γεγονότα*, μια ακολουθία των οποίων συνιστά μια *κατάσταση*. Οι καταστάσεις είναι ολικές στον κόσμο που αφορούν και δημιουργούνται από την αντίστοιχη ακολουθία γεγονότων. Τα γεγονότα μετατρέπουν μια παγκόσμια κατάσταση σε μία άλλη. Αντιθέτως, η EC χειρίζεται τα γεγονότα τοπικά αποδίδοντας σε καθένα από αυτά το κατάλληλο αρχικό και τελικό σημείο. Όπως και στην περίπτωση της EC, ο Λογισμός Καταστάσεων μπορεί να υλοποιηθεί με τις προτάσεις Horn που έχουν συμπληρωθεί χρησιμοποιώντας «άρνηση ως αποτυχία», με αποτέλεσμα ένα λογικό πρόγραμμα [17]. Ωστόσο, η απόφαση για την επίδραση ενός γεγονότος σε κάθε σχέση μιας κατάστασης μπορεί να επιβάλει ένα μεγάλο υπολογιστικό κόστος. Αντιθέτως, η EC χρησιμοποιεί την έννοια της αδράνεια και, κατά συνέπεια, συλλογίζεται μόνο για τα fluents που συσχετίζονται με το τρέχον γεγονός. Πρόσθετες εργασίες που αναδεικνύουν τον τρόπο με τον οποίο η EC αντιστοιχεί στον Λογισμό Καταστάσεων παραθέτονται εδώ [19] [20] [21] [22].
Στην [23], οι Miller και Shanahan έχουν προτείνει ένα σύνολο αξιωμάτων για την EC, που ορίζονται εφαρμόζοντας κλασσική λογική. Έπειτα, παρουσίασαν εναλλακτικές αξιωματοποιήσεις της EC μεταβάλλοντας ελαφρώς διαφορετικά υποσύνολα των προτεινόμενων αξιωμάτων. Κάποιες ενδιαφέρουσες περιπτώσεις ήταν ο Ντετερμινιστικός Λογισμός Γεγονότων, ο οποίος απαγορεύει την ταυτόχρονη εκκίνηση και τερματισμό ενός fluent, η Κατηγοριοποίηση των Fluents στην EC, η οποία χωρίζει τα fluents σε *frame* fluents και *non-frame*, διευκρινίζοντας ότι μόνο τα frame fluents που συμμορφώνονται στο νόμο της αδράνειας. Στην [24], οι Miller και Shanahan επεκτείνουν την προηγούμενη προσέγγισή τους συζητώντας τη δυνατότητα αναδιατύπωσης της EC κλασσικής λογικής σε «γλώσσες περιγραφής ενεργειών» για την περιγραφή του τομέα εφαρμογής χρησιμοποιώντας τη γλώσσα $\mathcal{E}$ [25] [26].
Αυτές οι «γλώσσες περιγραφής ενεργειών» είναι σε θέση να αναπαριστούν και να επεξεργάζονται

τις επιδράσεις των γεγονότων με έναν πιο εξειδικευμένο ανά τομέα τρόπο [27] [28]. Η γλώσσα $\mathcal{A}$ είναι μια γλώσσα περιγραφής ενεργειών που παρουσιάζει ομοιότητες με τον Λογισμό Καταστάσεων. Στην [28], η γλώσσα $\mathcal{A}$ έχει εφαρμοστεί για την επεξεργασία εκτεταμένων λογικών προγραμμάτων, τα οποία διαχωρίζουν την «άρνηση ως αποτυχία» από την κλασσική άρνηση [29]. Παρόλο που οι περισσότερες γλώσσες περιγραφής ενεργειών επηρεάστηκαν από τον Λογισμό Καταστάσεων, η γλώσσα $\mathcal{E}$ κληρονομεί τις οντολογίες της από τον EC. Στην [24], έχει παρουσιαστεί μια μετάφραση μεταξύ του EC και της γλώσσας $\mathcal{E}$. Αυτή δείχνει ότι η EC κληρονομεί τις αποδεδειγμένα ορθές αυτοματοποιημένες διαδικασίες λογικής της $\mathcal{E}$. Στην [30], οι Allen και Ferguson επικρίνουν τις γλώσσες περιγραφής ενεργειών και τον Λογισμό Καταστάσεων σχετικά με την ικανότητά τους να εκτελούν θεμελιώδεις εργασίες συλλογιστικής, ενώ προτείνουν μια χρονική λογική δράσεων [31] [32] ως λύση. Αρχικά, περιγράφουν τη χρονική δομή της λογικής ως ένα γραμμικό μοντέλο χρόνου με ένα πρωτεύον αντικείμενο, την χρονική περίοδο, και μια πρωταρχική σχέση που ονομάζεται *Meets*. Στη συνέχεια, καθιερώνουν μια χρονική λογική διαστήματος με βάση αυτή τη δομή του χρόνου. Παρομοίως με αυτήν την προσέγγιση, η EC εμφανίζει μια σαφή συσχέτιση μεταξύ του χρόνου, των γεγονότων και των τιμών των fluents, αυξάνοντας την εκφραστικότητα και των δύο φορμαλισμών σε σύγκριση με τον στατικό Λογισμό Καταστάσεων και τις μη χρονικές επεκτάσεις του. Όμως, η EC έχει μια ασθενέστερη αναπαράσταση των γεγονότων, καθώς οι επιπτώσεις τους μπορούν να συμβούν μόνο αμέσως μετά την εμφάνισή τους και δεν είναι σαφές πώς θα μπορούσαν να επεκταθούν τα γεγονότα στο χρόνο.

Διάφορες διάλεκτοι του Λογισμού Γεγονότων έχουν προταθεί στην σχετική βιβλιογραφία. Μερικές από αυτές παρουσιάζονται εδώ. Ο Cached EC (Chittaro και Montanari [38]) επεκτείνει τον EC με ένα μηχανισμό προσωρινής αποθήκευσης που λαμβάνει τις πραγματοποιήσεις γεγονότων ως είσοδο και ενημερώνει αναλόγως την προσωρινά αποθηκευμένη ομάδα διαστημάτων μέγιστης εγκυρότητας (maximal validity intervals – MVIs). Ένα MVI για ένα δεδομένο fluent αντιπροσωπεύει το μέγιστο χρονικό διάστημα μέσα στο οποίο ισχύει το fluent, δεδομένης μιας γνωστής ροής γεγονότων. Ο κύριος στόχος του Cached EC είναι να ξεπεραστούν τα προβλήματα αποδοτικότητας της EC εκτελώντας την διαδικασία συλλογισμού μόνο για το ελάχιστο δυνατό σύνολο από MVIs κάθε φορά που συμβαίνει ένα νέο γεγονός. Με βάση αυτή την εργασία των Chittaro και Montanari και την έννοια των διαστημάτων μέγιστης εγκυρότητας, έχουν προταθεί περισσότερες επεκτάσεις της EC. Ο Macro-Event Calculus [37] επεκτείνει την αρχική προσέγγιση επιτρέποντας την ταυτόχρονη εμφάνιση συσχετισμένων συμβάντων και την αποδοχή γεγονότων που συμβαίνουν σε χρονικά διαστήματα. Ο Reactive Event Calculus [39] εισάγει την έννοια της *αμετάκλητικότητας* στην αρχική προσέγγιση της EC και παρουσιάζει αντιδραστικές λειτουργίες που της επιτρέπουν να συλλογίζεται σε ένα δυναμικό περιβάλλον.

3.2 Μη-πιθανοτικά συστήματα αναγνώρισης σύνθετων γεγονότων βασισμένα στη λογική

Στην [33], ο Dousson παρουσιάζει ένα σύστημα αναγνώρισης «χρονικών» βασισμένο σε ένα προτασιακό τεκμηριωμένο φορμαλισμό Χρονικής Λογικής [35]. Ένα «χρονικό» είναι ένα σύνολο γεγονότων, που συνδέονται μεταξύ τους με χρονικούς περιορισμούς, με τη μορφή ενός κατευθυνόμενου γράφου που αντιπροσωπεύει ένα χρονικό πρότυπο LTA. Οι αλγόριθμοι αναγνώρισης «χρονικών» επεξεργάζονται τη ροή γεγονότων εισόδου προσδιορίζοντας όλα τα παρατηρούμενα σύνολα γεγονότων που ταιριάζουν με ένα πρότυπο «χρονικού» συμβάντος σε σχέση με τους χρονικούς περιορισμούς. Η [33] επεκτείνει την αναπαράσταση «χρονικών» εισάγοντας μετρητές γεγονότων με το

κατηγόρημα *occurs*. Μια περαιτέρω επέκταση που περιλαμβάνει χρονική εστίαση σε ασυνήθιστα γεγονότα και ιεράρχηση έχει προταθεί και δοκιμαστεί στην εργασία [34].
Η ταξινόμηση γεγονότων με βάση την πολυπλοκότητα και τη συχνότητά τους είναι μια κοινή προσέγγιση στα πλαίσια αναγνώρισης συμβάντων. Μια μέθοδος για την εκμάθηση εξαρτήσεων υπό συνθήκες μεταξύ γεγονότων και την κωδικοποίησή τους σε ένα γράφημα έχει προταθεί στην [36]. Αυτά τα *γραφήματα εξάρτησης υπο-γεγονότων* χρησιμοποιούνται για την αναγνώριση γεγονότων σε πρωτότυπα βίντεο. Στη συνέχεια περιγράφεται μία εκτεταμένη αναπαράσταση για τη συσχέτιση του μοντέλου γεγονότων με την ανθρώπινη κατανόηση. Η καινοτομία της εφαρμοζόμενης μεθόδου είναι η δυνατότητα ανίχνευσης πολλαπλών δραστών που εκτελούν πολλαπλές ενέργειες, χωρίς προηγούμενης γνώσης σχετικά με τον αριθμό των δραστών ή τη διάρκεια των συμβάντων.

3.3 Συστήματα πιθανοτικής αναγνώρισης σύνθετων γεγονότων

Η διαδικασία αναγνώρισης σύνθετων γεγονότων από μία ροή απλών ενεργειών γίνεται πιο δυσνόητη όταν θεωρούμε ότι τα χρονικά σηματοδοτημένα γεγονότα της ροής εισόδου μπορεί να είναι λανθασμένα [5]. Αυτή η αβεβαιότητα των συμβάντων εισόδου μπορεί να οφείλεται σε δυσλειτουργία κάποιου αισθητήρα ή σε βλάβη μετάδοσης. Ορισμένες μετρήσεις ενδέχεται να φτάσουν στη ροή με καθυστέρηση [11], να είναι ελλιπείς ή να λείπουν εξ ολοκλήρου [40]. Πρόσθετοι παράγοντες που προκαλούν αβεβαιότητα στην ανίχνευση απλών συμβάντων είναι τα φραγμένα αντικείμενα και τα θορυβώδη ακουστικά σήματα. Επομένως, τα ανιχνευθέντα απλά γεγονότα έχουν μία τιμή πιθανότητας συνδεδεμένη με αυτά. Σε ορισμένες περιπτώσεις, η αβεβαιότητα προέρχεται από τους καθορισμένους κανόνες λογικής σύνθετων γεγονότων εξαιτίας της ανεπαρκούς γνώσης του τομέα της εφαρμογής. Για τον λόγο αυτό, οι Skarlatidis et al. χρησιμοποιήσαν Μαρκοβιανά Λογικά Δίκτυα (Markov Logic Networks – MLN) [41] για να χαλαρώσουν τους περιορισμούς που επιβάλλονται από αυτούς τους κανόνες, προσθέτοντας μια τιμή βάρους στον καθένα [42].

3.3.1 Συστήματα βασισμένα σε Μαρκοβιανά Λογικά Δίκτυα

Μια έρευνα σχετική με την αβεβαιότητα στην αναγνώριση σύνθετων γεγονότων [43] περιγράφει μεθόδους για την προσάρτηση τιμών πιθανότητας σε γεγονότα και κανόνες και για την εξαγωγή σύνθετων γεγονότων σε ένα τέτοιο πλαίσιο. Η έρευνα επικεντρώνεται στα πιθανοτικά γραφικά μοντέλα - με έμφαση στα Μαρκοβιανά Λογικά Δίκτυα. Όπως αναφέρθηκε, τα MLN είναι σε θέση να μετατρέψουν αυστηρούς λογικούς κανόνες σε χαλαρούς περιορισμούς (F) με βάρος (w). Μια μεγάλη τιμή βάρους αντιστοιχεί σε ένα σφιχτό περιορισμό, καθιστώντας τους κόσμους που τον παραβιάζουν λιγότερο πιθανούς. Αντίθετα, μια μικρή τιμή βάρους χαλαρώνει τον αντίστοιχο περιορισμό, αυξάνοντας την πιθανότητα των κόσμων που τον παραβιάζουν. Μια εκτενής εργασία για την εφαρμογή MLN στην αναγνώριση γεγονότων παρουσιάζεται στις [4] [44].
Στην [45] επεξεργάζονται απλά πιθανοτικά γεγονότα από πολλαπλά σημεία - ανίχνευση πόζας, ανίχνευση αντικειμένων και μετακίνηση - για να εξάγουν πολύπλοκα γεγονότα με τη χρήση Δυναμικών Μαρκοβιανών Λογικών Δικτύων (Dynamic Markov Logic Networks – DMLN) [46]. Τα Δυναμικά MLN επεκτείνουν την κλασσική προσέγγιση των MLN για να διευκολύνουν τη συλλογιστική σε δυναμικά πεδία εφαρμογών όπου ένα βασικό κατηγόρημα μπορεί να είναι true ή false ανάλογα με το βήμα χρόνου (t). Πιο συγκεκριμένα, τα DMLN αντικαθιστούν πρότυπα πρώτης

τάξης με τα αντίστοιχα fluents προσθέτοντας μια χρονική παράμετρο, χρησιμοποιώντας την συνάρτηση διαδόχου *succ(t)* και επιβάλλοντας χρονικούς περιορισμούς με αποτέλεσμα ένα MLN που περιλαμβάνει ένα σύνολο σταθμισμένων τύπων που ορίζονται σε fluents. Μια πρόσθετη εφαρμογή των MLN για την αναγνώριση δραστηριοτήτων [47] επιτρέπει την αναγνώριση των παρεμβαλλομένων και ταυτόχρονων συμβάντων διαχωρίζοντας τις δραστηριότητες του τομέα της εφαρμογής σε δραστηριότητες *υποβάθρου* και *προσκηνίου*, επιτρέποντας στους δράστες να εμπλέκονται σε περισσότερες από μία σύνθετες δραστηριότητες. Η [48] περιγράφει μια λεπτομερή εφαρμογή MLN, ανθεκτική σε ελλιπή ή απόντα δεδομένα εισόδου, αβέβαιους λογικούς κανόνες και αβεβαιότητα εκτεταμένης αξιολόγησης, για οπτική παρακολούθηση σε χώρο στάθμευσης όπου οι δραστηριότητες αντιστοιχούν σε σύνθετες αλληλεπιδράσεις ανθρώπων και οχημάτων. Στην [52], οι σχεσιακές δραστηριότητες πολλαπλών δραστών - είτε επιτυχείς, σκοπευμένες ή αποτυχημένες - συνάγονται από θορυβώδη δεδομένα GPS που απεικονίζουν ανθρώπους που αλληλεπιδρούν σε ένα παιχνίδι σύλληψης της σημαίας. Η μηχανή συλλογισμού τους χρησιμοποιεί υβριδικά-MLN [51], μια επέκταση των MLN στην οποία τα χαρακτηριστικά μπορεί να είναι συνεχείς ιδιότητες. Αυτή η προσέγγιση επιτρέπει τη μοντελοποίηση χαλαρών περιορισμών (π.χ. όσον αφορά την απόσταση παικτών) που παίρνουν πραγματικές τιμές. Τέλος, παρουσιάζουμε δύο προσπάθειες αξιοποίησης των MLN για αναγνώριση γεγονότων στον τομέα ενός παιχνιδιού μπάσκετ [49] [50].

Είναι σαφές ότι τα MLN είναι μια αποτελεσματική λύση για τον χειρισμό της αβεβαιότητας στην αναγνώριση γεγονότων, ιδιαίτερα της αβεβαιότητας στις παρατήρησεις χαμηλού επιπέδου που προέρχονται από τους αισθητήρες εισόδου (θολούρα, θόρυβος σε ακουστικά σήματα, παρεμβαλλόμενα αντικείμενα, μεταβαλλόμενο παρασκήνιο κλπ.) [49] [48]. Ομοίως με την βασισμένη στον Prob-EC προσέγγισή μας, τα MLN χρησιμοποιούν κανόνες βασισμένους σε γνώσεις ειδικών του αντίστοιχου τομέα για να συμπεράνουν πολύπλοκα γεγονότα. Και οι δύο προσεγγίσεις βασίζονται στη γνώση καθώς βασίζονται σε λογικούς κανόνες για την αναγνώριση γεγονότων σε κάποιο βαθμό. Σε μια μεταγενέστερη ενότητα περιγράφουμε άλλες πιθανοτικές γραφικές μεθόδους που υιοθετούν προσέγγισεις βασισμένες στα δεδομένα. Παρόλο που η προσέγγιση αυτή είναι ανθεκτική στην αβεβαιότητα, στερείται εκφραστικότητα και δεν φαίνεται γνώριμη στους ειδικούς του εκάστοτε τομέα. Ο Prob-EC αποφεύγει τη χρήση MLN λόγω της εγγενούς δυσκολίας έκφρασης των χρονικών περιορισμών στα MLN. Αυτή η παρατήρηση υποστηρίζεται στην εργασία [42] όπου χρησιμοποιείται μια διακριτή εκδοχή του Λογισμού Γεγονότων για να αποφευχθεί η συνδυαστική έκρηξη του αριθμού των προτάσεων που παράγονται από αξιώματα που ποσοτικοποιούνται σε μεγάλο βαθμό από χρονικές μεταβλητές. Κάθε ποσοτικοποίηση κάθε πρότασης θα αντιπροσωπευόταν από έναν κόμβο στο αντίστοιχο Μαρκοβιανό Δίκτυο, καθιστώντας τον υπολογιστικά και χωρικά ακατόρθωτο.

3.3.2 Συστήματα βασισμένα σε άλλα πιθανοτικά γραφικά μοντέλα

Σε αυτήν την ενότητα, παρουσιάζουμε περαιτέρω εργασίες σχετικά με την αναγνώριση σύνθετων γεγονότων με χρήση διαφορετικών πιθανοτικών γραφικών μοντέλων, όπως τα Κρυφά Μαρκοβιανά Μοντέλα (Hidden Markov Models – HMM) [53] [54], τα Δυναμικά Μπεϋζιανά Μοντέλα (Dynamic Bayesian Networks – DBN) [58] [59] και τα Υπο Συνθήκη Τυχαία Πεδία (Conditional Random Fields – CRF)[61]. Αυτές οι προσεγγίσεις δεν χρησιμοποιούν λογικούς κανόνες και βασίζονται κυρίως σε δεδομένα.

Τα Κρυφά Μαρκοβιανά Μοντέλα είναι μια επέκταση των κλασικών Μοντέλων Markov όπου η παρατήρηση δεν είναι η ίδια η κατάσταση, αλλά μια πιθανοτική συνάρτηση της κατάστασης. Κατά συνέπεια, τα HMM είναι μια διπλά ενσωματωμένη στοχαστική διαδικασία με μια υποκείμενη στοχαστική διαδικασία που δεν είναι παρατηρήσιμη, αλλά μπορεί να παρατηρηθεί μόνο μέσω μιας άλλης

σειράς στοχαστικών διαδικασιών που παράγουν την ακολουθία των παρατηρημένων συμβόλων. Εκτός από την αναγνώριση ομιλίας [54], τα HMMs έχουν χρησιμοποιηθεί για αναγνώριση σύνθετων γεγονότων σε διάφορους τομείς [55] [56]. Επιπλέον, ένα σύστημα HMM έχει συνδυαστεί με γνώση πεδίου εφαρμογής για να συναγάγει σύνθετα γεγονότα στον τομέα ενός παιχνιδιού τένις [57].
Τα Δυναμικά Μπεϋζιανά Δίκτυα επεκτείνουν τα Bayes Nets για την μοντελοποίηση κατανομών πιθανοτήτων σε ημι-άπειρες συλλογές τυχαίων μεταβλητών, δημιουργώντας ένα δίκτυο που αλλάζει με την πάροδο του χρόνου. Στην [60], έχουν χρησιμοποιηθεί τρία στρώματα DBN για την αναγνώριση θέσεων σώματικων μελών, δραστηριοτήτων ενός ατόμου και αλληλεπιδράσεων πολλαπλών δραστών, αντίστοιχα, από δεδομένα που προέρχονται από βίντεο παρακολούθησης χώρου.
Τα Υπο Συνθήκη Τυχαία Πεδία είναι γραφικά μοντέλα τα οποία περιλαμβάνουν ένα γράφημα του οποίου οι κορυφές αντιστοιχούν σε πιθανές ακολουθίες ετικετών για τις ακολουθίες δεδομένων εισόδου. Η καινοτομία των CRF είναι το γεγονός ότι οι κορυφές τους υπακούουν στην ιδιότητα Markov όσον αφορά τις γειτονικές κορυφές στο προαναφερθέν γράφημα [61]. Η [62] χρησιμοποιεί το CRF για να αναγνωρίσει τις δραστηριότητες που εκτελεί ένας κάτοικος σε ένα έξυπνο οικιακό περιβάλλον και συγκρίνει τα αποτελέσματα της αναγνώρισης CRF με αντίστοιχη εφαρμογή HMM.
Σε σύγκριση με την προσέγγιση μας που βασίζεται στη λογική, αυτές οι μέθοδοι στερούνται μία τυπική γλώσσα αναπαράστασης, περιπλέκοντας την ερμηνεία των αποτελεσμάτων. Επιπλέον, τα HMM και τα DBN δεν διαθέτουν ευελιξία στην αναγνώριση πολύπλοκων δραστηριοτήτων που περιλαμβάνουν πολλούς δράστες, πολλαπλές σχέσεις ή επεκτείνονται σε μεγάλα χρονικά διάστηματα. Αυτή η δυσκολία απορρέει από το μεγάλο ποσό των πόρων που απαιτούνται για να αναπαρασταθούν εξαντλητικά όλα τα πιθανά αντικείμενα, οι σχέσεις και οι χρονικούς περιορισμούς.

Κεφάλαιο 4

Πιθανοτικός Λογισμός Γεγονότων με βάση χρονικά διαστήματα σε ροές δεδομένων

Μέχρι στιγμής, έχουμε εξετάσει τον PIEC, έναν γραμμικό αλγόριθμο χρόνου που είναι σε θέση να αναγνωρίζει LTA σε χρονικά διαστήματα, εξαλείφοντας κάποια από την αβεβαιότητα της αναγνώρισης σε χρονικές στιγμές. Είναι σημαντικό να σημειωθεί ότι ο PIEC εξετάζει μια στατική λίστα στιγμιαίων πιθανοτήτων, που σημαίνει ότι δεν είναι προσαρμοστικός στην εμφάνιση επιπλέον δεδομένων και, επομένως, δεν μπορεί να προσομοιώνει ένα online σύστημα. Για παράδειγμα, σκεφτείτε έναν αισθητήρα που λειτουργεί όλη την ημέρα μέχρι τις 17:00. Μέχρι τότε, θα είχε συγκεντρώσει πολλά δεδομένα που αποτελούνται από STA, τη χρονική στιγμή εμφάνισής τους και την πιθανότητά τους. Ένας pipeline των συστημάτων Prob-EC και PIEC παράγει μια σειρά από διαστήματα στα οποία συμβαίνει μία LTA, όπως για παράδειγμα η δραστηριότητα *moving*(mike, sarah). Σκεφτείτε τώρα ότι περνάνε 5 λεπτά και θέλουμε να αποφασίσουμε τα διαστήματα του ίδιου LTA. Ο PIEC θα πρέπει να συλλογιστεί για όλα τα δεδομένα που έλαβε ο αισθητήρας από τα μεσάνυχτα μέχρι τις 17:05 ώστε να εξάγει τη σωστή λίστα διαστημάτων, ακόμη και αν ο χρήστης ενδιαφέρεται μόνο για τις αλλαγές που προξένησαν τα δεδομένα που λήφθηκαν τα τελευταία 5 λεπτά. Αυτή η αναγκαιότητα οφείλεται στο γεγονός ότι ο PIEC ασχολείται με μέγιστα πιθανοτικά διαστήματα και, γι΄ αυτό το λόγο, πρέπει να διασχίσει ολόκληρη τη λίστα στιγμιαίων πιθανοτήτων ώστε να βεβαιωθεί πως τα διαστήματα που βρέθηκαν μέχρι τις 17:00 δε μπορούν να επεκταθούν.

Ας υποθέσουμε, για παράδειγμα, ότι ο PIEC εντοπίζει μία LTA για το διάστημα $I_1 = [16:45 - 16:55]$ όταν εξετάζει όλα τα δεδομένα που παράγονται μέχρι τις 17:00. Πέντε λεπτά αργότερα φτάνουν τα νέα δεδομένα και εκτελείται ο PIEC μόνο γι΄ αυτά. Αποφασίζει ότι το ίδιο LTA πραγματοποιείται και για το διάστημα $I_2 = [17:05 - 17:30]$. Θεωρώντας όμως τον ορισμό ενός μέγιστου πιθανοτικού διαστήματος (**Ορισμός 2**), δεν είναι ξεκάθαρο ότι το I_2 δεν μπορεί να επεκταθεί προς τα αριστερά, εξετάζοντας μόνο τα νέα δεδομένα. Για παράδειγμα, η πιθανότητα του διαστήματος $I_3 = [16:45 - 17:30]$, υπολογισμένη όπως προτείνεται (**Ορισμός 1**), θα μπορούσε να είναι μεγαλύτερη από το καθορισμένο από το χρήστη πιθανοτικό κατώφλι. Επομένως, δεν μπορούμε να αποφανθούμε με βεβαιότητα ότι είτε το I_1 ή το I_2 είναι μέγιστα πιθανοτικά διαστήματα. Είναι πλέον σαφές ότι η εκτέλεση του PIEC μόνο στα νέα δεδομένα δεν υπόσχεται το επιδιωκόμενο αποτέλεσμα.

Για να ξεπεράσουμε τα εμπόδια της αναγνώρισης σε ροές δεδομένων, προτείνουμε μια αναβαθμισμένη έκδοση του PIEC στην οποία θα αναφερόμαστε ως PIEC **Ροής Δεδομένων** (Streaming PIEC ή sPIEC για συντομία). Κατασκευάσαμε τον αλγόριθμο sPIEC με την ιδιότητα να είναι προσαρμοστικός στα επιπλέον δεδομένα που εμφανίζονται στη ροή, έχοντας τη δυνατότητα να συλλογίζεται σχετικά με τις αλλαγές που αυτά θα προκαλέσουν στην έξοδο χωρίς να χρειάζεται να λάβει υπόψη κάθε προηγούμενο χρονικό σημείο. Θεωρούμε ότι κάθε ομάδα νέων δεδομένων αποτελεί ένα παράθυρο στιγμιαίων πιθανοτήτων το οποίο εξετάζεται ξεχωριστά από τον sPIEC.
Ιδανικά, ο sPIEC θα λειτουργούσε εξέταζοντας μόνο το παράθυρο των στιγμιαίων πιθανοτήτων που συνάγεται από τα νέα δεδομένα και πιθανοτικό κατώφλι. Αλλά αυτό είναι αδύνατο καθώς ο αλγόριθμος πρέπει να έχει κάποια γνώση των πιθανοτήτων σε προηγούμενα χρονικά σημεία για να παράξει το επιδιωκόμενο αποτέλεσμα, όπως εξηγήθηκε στο παραπάνω παράδειγμα. Επομένως, ο στόχος μας σε αυτό το κεφάλαιο είναι να ορίσουμε μια αναβαθμισμένη έκδοση του PIEC που είναι σε θέση να λειτουργεί αποτελεσματικά σε ένα περιβάλλον ροής δεδομένων, εξετάζοντας μόνο τα δεδομένα του νεότερου παράθυρου, ενώ αποθηκεύει το ελάχιστο ποσό δεδομένων που αφορούν προηγούμενες χρονικές στιγμές.
Στις επόμενες ενότητες, θα παρουσιάσουμε μια επισκόπηση του sPIEC, περιγράφοντας τη δομή και τους κύριους μηχανισμούς του. Επίσης, θα ορίσουμε όλες τις δομές δεδομένων που αποθηκεύονται μεταξύ εκτελέσεων διαδοχικών παραθύρων. Θα αναφερόμαστε σε αυτά τα δεδομένα ως **Δεδομένα Υποστήριξης** (Support Data). Στη συνέχεια, παρέχουμε ψευδοκώδικα που περιγράφει τις συναρτήσεις που εκτελούν τις πιο κρίσιμες λειτουργίες του sPIEC, εξετάζουμε την πολυπλοκότητα και την ορθότητα τους και παρουσιάζουμε ένα παράδειγμα εκτέλεσης του.

4.1 Επισκόπηση

4.1.1 Η δομή του sPIEC

Για τη καλύτερη κατανόηση του αλγορίθμου, αποφασίσαμε πρώτα να δώσουμε ένα σκιαγράφημα του sPIEC, χωρίζοντάς τον σε συναρτήσεις τις οποίες θα σχολιάσουμε ανεξάρτητα. Διαχωρίζουμε τον sPIEC σε δύο στρώματα, το εξωτερικό και το εσωτερικό.
Το εξωτερικό στρώμα περιλαμβάνει μια συνάρτηση η οποία λαμβάνει ως είσοδο μία λίστα στιγμιαίων πιθανοτήτων A, πιθανώς προερχόμενη από τις πιο πρόσφατες μετρήσεις ενός αισθητήρα, το πιθανοτικό κατώφλι T και μια καθορισμένη από το χρήστη τιμή για το μέγεθος των παραθύρων win_size. Ο κύριος σκοπός του είναι να διαχωρίσει τη λίστα A σε παράθυρα καθορισμένου μεγέθους και να προωθήσει το καθένα, μαζί με τα απαραίτητα δεδομένα υποστήριξης, στο εσωτερικό στρώμα που θα εκτελέσει τη διαδικασία συλλογισμού. Στη συνέχεια, το εσωτερικό στρώμα θα επιστρέψει τα αναγνωρισμένα διαστήματα στο εξωτερικό, συνοδευόμενα από τροποποιημένα δεδομένα υποστήριξης, τα οποία αργότερα θα τροφοδοτηθούν στο επόμενο παράθυρο. Έπειτα, το εξωτερικό στρώμα εκτελεί μια διαδικασία διαχωρισμού που αποφασίζει ποια από τα διαστήματα που έχουν εντοπιστεί από εκτελέσεις προηγούμενων παραθύρων είναι υποδιαστήματα κάποιου διαστήματος που επιστράφηκε από το εσωτερικό στρώμα. Η διαγραφή αυτών των υποδιαστημάτων, συνοδευόμενη από το γεγονός ότι το εσωτερικό στρώμα επιστρέφει όλα τα προβλεπόμενα μέγιστα πιθανοτικά διαστήματα, εγγυάται ότι το τελικό σύνολο παραγόμενων διαστημάτων θα περιέχει μόνο μέγιστα πιθανοτικά διαστήματα. Η επανάληψη αυτής της διαδικασίας για κάθε παράθυρο της διαιρεμένης ροής εισόδου παράγει ένα τελικό σύνολο διαστημάτων και μια τελική συλλογή δεδομένων υποστήριξης, τα οποία αντιστοιχούν στην εκτέλεση του εσωτερικού στρώματος για το τελευταίο παράθυρο της ροής. Το πρώτο σύνολο περιλαμβάνει όλα τα μέγιστα πιθανοτικά διαστήματα που κυμαίνονται από την αρχή

την αρχή μέχρι την τελευταία χρονική στιγμή που έχει σταλεί στον sPIEC. Στη συνέχεια, θα αποδείξουμε ότι αυτό το σύνολο διαστημάτων είναι πανομοιότυπο με το σύνολο που θα αναγνώριζε ο αρχικός PIEC αν εκτελούνταν στατικά στα ίδια δεδομένα. Τα τελικά δεδομένα υποστήριξης εξάγονται ώστε να προωθηθούν σε ένα μελλοντικό παράθυρο όταν εμφανιστούν νέα δεδομένα στη ροή. Έτσι, ο sPIEC εγγυάται την ορθότητα μελλοντικών εκτελέσεων για νέα παράθυρα δεδομένων. Ένα σκιαγράφημα του εξωτερικού στρώματος του sPIEC παρουσιάζεται με ψευδοκώδικα παρακάτω (Αλγόριθμος 2).

Algorithm 2 Το εξωτερικό στρώμα του sPIEC

Είσοδος: Μία λίστα από στιγμιαίες πιθανότητες A, το πιθανοτικό κατώφλι T και το μέγεθος παραθύρου win_size.
Έξοδος: Μία λίστα με όλα τα μέγιστα πιθανοτικά διαστήματα και μία συλλογή από δεδομένα υποστήριξης.
Απαιτεί: Τα διαστήματα και τα δεδομένα υποστήριξης προηγούμενων εκτελέσεων (αν υπάρχουν).

```
1: procedure sPIEC_OUTER_LAYER(A, T, win_size)
2:     windows ← split(A, win_size)
3:     for each window in windows do
4:         (new_intervals, new_support_data) = sPIEC_inner_layer(window, , support_data)
5:         intervals ← unravel_intervals(intervals, new_intervals)
6:         support_data ← new_support_data
       return (intervals, support_data)
```

Το εσωτερικό στρώμα περιέχει τον αλγόριθμο που χρησιμοποιείται για την εύρεση όλων των μέγιστων πιθανοτικών διαστημάτων που είτε βρίσκονται εξ ολοκλήρου μέσα στο παράθυρο υπό εξέταση είτε ξεκινούν από μία παρελθοντική χρονική στιγμή και καταλήγουν εντός αυτού του παραθύρου. Τα πρώτα ανιχνεύονται με την εκτέλεση του αρχικού αλγόριθμου PIEC με την παράμετρο A να συμπίπτει με το τρέχον παράθυρο και το T με το πιθανοτικό κατώφλι του χρήστη. Τα τελευταία είναι πιο προβληματικά στην ανίχνευση, καθώς απαιτούν γνώσεις σχετικά με τις προγενέστερες χρονικές στιγμές. Για να επιλυθεί αυτό το ζήτημα, τα δεδομένα υποστήριξης του sPIEC περιέχουν μία πολύ κρίσιμη δομή δεδομένων που ονομάζεται **Σύνολο Υποστήριξης** (Support Set). Η δομή αυτή περιέχει κάθε παρελθοντικό χρονικό σημείο που αποτελεί πιθανό σημείο εκκίνησης ενός μέγιστου πιθανοτικού διαστήματος. Κάθε τέτοιο χρονικό σημείο συνοδεύεται από μια πραγματική τιμή, που ονομάζεται *prefix_prev*, η οποία υποδεικνύει την ελάχιστη τιμή που πρέπει να εμφανίζεται στη λίστα dp του παραθύρου ώστε το χρονικό αυτό σημείο να είναι το σημείο εκκίνησης ενός πιθανοτικού διαστήματος που λήγει στο τρέχον παράθυρο. Το σύνολο υποστήριξης θα αναλυθεί σε βάθος σε ακόλουθες ενότητες.
Το εσωτερικό στρώμα μπορεί να περιγραφεί ως 3 χωριστές λειτουργίες οι οποίες είναι ανεξάρτητες και παράγουν διαφορετικά αποτελέσματα. Οι λειτουργίες αυτές παρουσιάζονται παρακάτω:

1. Ανίχνευση όλων των μέγιστων πιθανοτικών διαστημάτων που ξεκινούν από μία χρονική στιγμή που είναι αποθηκευμένη στο σύνολο υποστήριξης και τελειώνουν στο τρέχον παράθυρο. Αυτή η λειτουργία περιγράφεται σε ακόλουθη ενότητα (Αλγόριθμος 4).

2. Εφαρμογή του PIEC για την εύρεση όλων των μέγιστων πιθανοτικών διαστημάτων εντός του τρέχοντος παραθύρου.

3. Ανίχνευση κάθε χρονικού σημείου στο τρέχον παράθυρο, το οποίο αποτελεί πιθανό σημείο εκκίνησης ενός μέγιστου πιθανοτικού διαστήματος και συνεπώς πρέπει να προστεθεί στο σύνολο υποστήριξης. Ο αλγόριθμος που αποφασίζει αυτά τα χρονικά σημεία δίνεται σε μία ακόλουθη ενότητα (Αλγόριθμος 5).

Το σκιαγράφημα που περιγράψαμε για το εσωτερικό στρώμα του sPIEC παρουσιάζεται παρακάτω (Αλγόριθμος 3). Μία αναλυτική περιγραφή των δεδομένων υποστήριξης που περιέχονται σε αυτό δίνονται στην επόμενη ενότητα.

Algorithm 3 Το εσωτερικό στρώμα του sPIEC

Είσοδος: Μία λίστα από στιγμιαίες πιθανότητες A που αντιστοιχούν στα δεδομένα ενός παραθύρου, το πιθανοτικό κατώφλι T και τα δεδομένα υποστήριξης. Αυτά περιλαμβάνουν τα *win_count*, *prefix_prev*, *last_prefix* και *support_set* που θα αναλυθούν στη συνέχεια
Έξοδος: Μία λίστα από μέγιστα πιθανοτικά διαστήματα και μία συλλογή από τροποποιημένες δομές δεδομένων υποστήριξης.
Απαιτεί: Τις λίστες *prefix* και *dp*

1: **procedure** sPIEC_INNER_LAYER(A, T, *support_data*)
2: $results\_from\_support\_set \leftarrow find\_intervals\_from\_support\_set(dp, support\_set)$
3: $results\_inside\_window \leftarrow PIEC(A, T)$
4: $intervals \leftarrow results\_from\_support\_set \cup results\_inside\_window$
5: $new\_elements \leftarrow find\_potential\_starting\_points(prefix, last\_prefix, prefix\_prev)$
6: $support\_set \overset{\text{add}}{\Longleftarrow} new\_elements$
7: **return** $(intervals, support\_data)$

4.1.2 Δεδομένα Υποστήριξης

Ορίζουμε ως δεδομένα υποστήριξης όλα τα δεδομένα που πρέπει να αποθηκευτούν μετά την επεξεργασία ενός παραθύρου για να εξασφαλιστεί ότι η επεξεργασία ενός μελλοντικού παραθύρου θα παράξει τα σωστά αποτελέσματα. Τα δεδομένα υποστήριξης μεταφέρονται μεταξύ χρονικά διαδοχικών παραθύρων. Όλες οι δομές δεδομένων που περιλαμβάνονται στα δεδομένα υποστήριξης του αλγορίθμου παρουσιάζονται παρακάτω:

1. *winId*: Ένας αυξητικός ακέραιος που χρησιμοποιείται ως αναγνωριστικό για κάθε παράθυρο. Χρησιμοποιείται για τον υπολογισμό του πλήθους των χρονικών σημείων που έχουν επεξεργαστεί από την αρχή της εκτέλεσης και για την ορθή ανάθεση ενός σημείου εκκίνησης και ενός σημείου τερματισμού σε κάθε διάστημα. Για παράδειγμα, υποθέστε ότι μετά την επεξεργασία του παραθύρου με $winId = i$ το εσωτερικό στρώμα εντόπισε το διάστημα $I = [s, e]$. Δεδομένου ότι s και e αναφέρονται σε χρονικές συντεταγμένες παραθύρου, μετατρέπονται σε χρονικές συντεταγμένες ροής μέσα στο εσωτερικό στρώμα κατά την αναγνώριση. Έτσι, το διάστημα I μετατρέπεται στο $I' = [s + i * win\_size, e + i * win\_size]$, το οποίο υπακούει στην ολική χρονική αρίθμηση των σημείων της ροής.

2. *last_prefix*: Η τιμή του τελευταίου στοιχείου της λίστας prefix του προηγούμενου παραθύρου. Δεδομένου ότι η λίστα prefix περιέχει τα συσσωρευτικά άθροίσματα της λίστας L, η προσωρινή

αποθήκευση της τελευταίας τιμής της αρχεί για να διατηρηθούν συνεπείς οι τιμές της λίστας στις επόμενες εκτελέσεις.

3. Σύνολο Υποστήριξης (*Support Set*): Μια λίστα πλειάδων με τη μορφή (Int, Double). Το πρώτο στοιχείο κάθε πλειάδας αντιπροσωπεύει ένα παρελθοντικό χρονικό σημείο, ενώ το δεύτερο διατηρεί την τιμή *prefix_prev* του σημείου αυτού. Τα χρονικά σημεία που περιλαμβάνονται στο *support set* είναι όλα τα πιθανά χρονικά σημεία έναρξης μέγιστων πιθανοτικών διαστημάτων που έχουν προσπελαστεί. Η τιμή *prefix_prev* ενός χρονικού σημείου αντιστοιχεί στην ελάχιστη τιμή dp που πρέπει να εμφανιστεί σε ένα μελλοντικό παράθυρο, προκειμένου να διασφαλιστεί η ύπαρξη πιθανοτικού χρονικού διαστήματος που αρχίζει από εκείνο το χρονικό σημείο και τελειώνει σε αυτό το παράθυρο.
Για καλύτερη κατανόηση του τρόπου με τον οποίο ο sPIEC επιλέγει πιθανά σημεία εκκίνησης και αναθέτει τις τιμές *prefix_prev*, παρουσιάζουμε μια σημαντική ιδιότητα του sPIEC, την οποία κληρονόμησε από το αρχικό αλγόριθμο PIEC.

Ιδιότητα 1. *Μία χρονική στιγμή t είναι το σημείο εκκίνησης ενός μέγιστου πιθανοτικού διαστήματος αν και μόνο αν*

$$\exists end \geq t :$$

(α΄) Ικανοποιείται η ακόλουθη ανίσωση:

$$dprange = \begin{cases} dp[end] - prefix[t-1] \geq 0 & t > 0 \\ dp[end] \geq 0 & t = 0 \end{cases} \tag{4.1}$$

(β΄) Δεν υπάρχει χρονικό σημείο $end' > end$ που να ικανοποιεί τη συνθήκη 4.1.

(γ΄) Δεν υπάρχει χρονικό σημείο $t' < t$ που να ικανοποιεί τη συνθήκη 4.1 για οποιοδήποτε πέρας end': $end' \geq end$.

Είναι φανερό από την Ιδιότητα 1 ότι η μόνη τιμή που εξαρτάται από το σημείο εκκίνησης t και είναι σχετική όταν εξετάζεται ένα μέγιστο πιθανοτικό διάστημα είναι η $prefix[t-1]$. Έτσι, ορίζουμε τη μεταβλητή *prefix_prev* ως εξής:

Ορισμός 3. *Θεωρούμε τη χρονική στιγμή t της ροής εισόδου. Ορίζουμε:*

$$prefix\_prev[t] = \begin{cases} prefix[t-1] & t > 0 \\ 0 & t = 0 \end{cases} \tag{4.2}$$

Σημειώνουμε ότι και στην Ιδιότητα 1 και στον Ορισμό 3 ο χρόνος υπολογίζεται αναφορικά με τη ροή δεδομένων. Επομένως, η χρονική στιγμή $t = 0$ είναι η πρώτη χρονική στιγμή ολόκληρης της ροής και το στιγμή εκκίνησης του κάθε παραθύρου.
Ορίζοντας το *prefix_prev* του σημείου t με αυτόν τον τρόπο, γίνεται φανερό ότι η εμφάνιση ενός χρονικού σημείου $t' > t$ για το οποίο το $dp[t'] \geq prefix\_prev[t]$ συνεπάγεται ότι $dprange[t, t'] \geq 0$ (Ισότητα 4.1), υποδεικνύοντας ότι το $I = [t, t']$ είναι ένα πιθανοτικό διάστημα.
Από τη μία πλευρά, η επιλογή των πλειάδων $(t, prefix\_prev[t])$ προς προσθήκη στο *support set* θα πρέπει να γίνει με συντηρητικό τρόπο για να ελαχιστοποιηθεί η ποσότητα μνήμης που απαιτείται για την αποθήκευση του *support set*. Από την άλλη πλευρά, η εξαίρεση

μίας πλειάδας μπορεί να οδηγήσει στην παράβλεψη κάποιου μέγιστου πιθανοτικού διαστήματος. Έτσι, ο στόχος μας είναι να αποκλείσουμε από το *support set* μόνο εκείνα τα χρονικά σημεία τα οποία είναι αδύνατο να είναι σημεία εκκίνησης ενός πιθανοτικού μέγιστου διαστήματος. Θα αναφερόμαστε σε όλα τα υπόλοιπα χρονικά σημεία ως *πιθανά σημεία εκκίνησης*. Η διαδικασία ανίχνευσης πιθανών σημείων εκκίνησης εκτελείται από τη συνάρτηση $find\_potential\_starting\_points$, όπως φαίνεται στο ψευδοκώδικα του εσωτερικού στρώματος (Αλγόριθμος 3, γραμμή 5), η οποία περιγράφεται σε βάθος σε μια επόμενη ενότητα (Αλγόριθμος 5).
Για παράδειγμα, θεωρούμε την ακόλουθη λίστα από στιγμιαίες πιθανότητες:

$Time$	0	1	2	3	4	5	6	7	8	9
A	0	0.5	0.7	0.9	0.4	0.1	0	0	0.5	1

Η είσοδος A είναι πανομοιότυπη με αυτή του παραδείγματος (Ενότητα 2.3.3) για το αλγόριθμο PIEC. Τα διαστήματα που είχαν βρεθεί σε εκείνο το παράδειγμα για το πιθανοτικό κατώφλι $T = 0.5$ ήταν τα $[0, 4], [1, 5]$ και $[7, 9]$.
Ο διαχωρισμός της εισόδου σε 2 παράθυρα και η εκτέλεση του PIEC σε καθένα ξεχωριστά θα είχε το ακόλουθο αποτέλεσμα:

Πρωτο παραθυρο

$Time$	0	1	2	3	4
A	0	0.5	0.7	0.9	0.4
L	-0.5	0	0.2	0.4	-0.1
$prefix$	-0.5	-0.5	-0.3	0.1	0
dp	0.1	0.1	0.1	0.1	0

- Ο PIEC εντοπίζει το διάστημα $[0, 4]$

Δευτερο παραθυρο

$Time$	5	6	7	8	9
A	0.1	0	0	0.5	1
L	-0.4	-0.5	-0.5	0	0.5
$prefix$	-0.4	-0.9	-1.4	-1.4	-0.9
dp	-0.4	-0.9	-0.9	-0.9	-0.9

- Ο PIEC εντοπίζει το διάστημα $[7, 9]$

Επειδή το χρονικό σημείο $t = 1$ ανήκει στο πρώτο παράθυρο, είναι αδύνατο να βρεθεί το διάστημα $[1, 5]$ χωρίς κάποια πληροφορία από την προηγούμενη εκτέλεση. Αυτός είναι ο λόγος για τον οποίο ο sPIEC θα είχε προσθέσει το χρονικό σημείο $= 1$ στο *support set* με τιμή *prefix_prev* ίση με -0.5. Το χρονικό σημείο $t = 5$ είναι το τελευταίο του οποίου η τιμή dp είναι μεγαλύτερη από -0.5. Έτσι, ο sPIEC θα εντόπιζε επιπλέον το διάστημα $[1, 5]$.

4. *ignore_value*:: Αυτή η μεταβλητή είναι πάντα ίση με το μικρότερο *prefix_prev* που έχει βρέθει προς το παρόν στη ροή δεδομένων. Χρησιμοποιείται από τον Αλγόριθμο 5 για την ανίχνευση νέων στοιχείων για το *support set*.

4.1.3 Εύρεση διαστημάτων που ξεκινάνε από σημεία εκκίνησης του support set

Σε αυτή την ενότητα θα περιγράψουμε τη λειτουργία που εκτελείται στο εσωτερικό στρώμα του sPIEC για την ανίχνευση διαστημάτων που ξεκινούν από ένα χρονικό σημείο που είναι αποθηκευμένο στο *support set* και τελειώνει σε ένα χρονικό σημείο εντός του τρέχοντος παραθύρου (Αλγόριθμος 3, γραμμή 2). Πρώτον, παρουσιάζουμε τον αλγόριθμο που εκτελεί αυτή τη λειτουργία και παρέχουμε τον απαραίτητο σχολιασμό. Στη συνέχεια, επιδεικνύουμε την πολυπλοκότητα του αλγορίθμου και αποδεικνύουμε την ορθότητα του.

Algorithm 4 Εύρεση μέγιστων πιθανοτικών διαστημάτων με βάση τα στοιχεία του support set

Είσοδος: Η λίστα dp του τρέχοντος παραθύρου και το *support set*
Έξοδος: Μία λίστα που περιέχει όλα τα μέγιστα πιθανοτικά διαστήματα που εκκινούν από ένα χρονικό σημείο του *support set* και τερματίζουν στο τρέχον παράθυρο.
Απαιτεί: win_size και $winId$.

1: **procedure** FIND_FROM_SUPPORT_SET(dp, $support\ set$)
2: $start \leftarrow 0$
3: $end \leftarrow 0$
4: $flag \leftarrow false$
5: $intervals \leftarrow \emptyset$
6: **while** $start < support\_set.length$ **and** $end < dp.length$ **do**
7: **if** $dp[end] \geq support\_set[start].prefix\_prev$ **then**
8: $flag \leftarrow true$
9: $end \mathrel{+}= 1$
10: **else**
11: **if** $flag == true$ **then**
12: $intervals \overset{\text{add}}{\Longleftarrow} (support\_set[start].timepoint, (end-1) + win\_size * winId)$
13: $flag \leftarrow false$
14: $start \mathrel{+}= 1$
15: **if** $flag == true$ **and** $end == dp.length$ **then**
16: $intervals \overset{\text{add}}{\Longleftarrow} (support\_set[start].timepoint, (end-1) + win\_size * winId)$
return intervals

4.1.3.1 Επισκόπηση

Οι κύριες λειτουργίες του Αλγόριθμου 4 εκτελούνται μέσα στο βρόχο *while* (Γραμμή 6). Αυτός ο βρόχος χρησιμοποιεί δύο δείκτες, *start* και *end*, για να διασχίσει το *support set* και τη λίστα dp του τρέχοντος παραθύρου, αντίστοιχα. Είναι σημαντικό να σημειώσουμε ότι τα στοιχεία του *support set* ταξινομούνται με αύξουσα χρονική σειρά. Ελέγχοντας τις λειτουργίες μέσα στον βρόχο *while*, μπορούμε να ανιχνεύσουμε πολλές ομοιότητες με τον αρχικό αλγόριθμο PIEC (Αλγόριθμος

1). Επειδή η τιμή *prefix_prev* ενός στοιχείου του *support set* έχει οριστεί ως η τιμή prefix του προηγούμενου χρονικού σημείου του, είναι σαφές ότι η συνθήκη του *if* στη γραμμή 7 ελέγχει αν η τιμή:

$$\begin{aligned} dp[end] - support\_set[start].prefix\_prev &= dp[end] - prefix[support\_set[start].timepoint - 1] \\ &= dprange[support\_set[start].timepoint, end] \quad (4.3) \end{aligned}$$

είναι αρνητική.

1. Αν η τιμή αυτή είναι μη αρνητική, τότε $dprange[support\_set[start].timepoint, end] \geq 0$ και, επομένως, το διάστημα $I = [support\_set[start].timepoint, end]$ μπορεί να επεκταθεί ώστε να γίνει πιθανοτικό. Στην μεταβλητή $flag$ ανατίθεται η τιμή true (γραμμή 8) και στη συνέχεια το *end* αυξάνεται ως προσπάθεια επέκτασης αυτού του διαστήματος στα δεξιά.

2. Εάν η τιμή είναι αρνητική, τότε $dprange[support\_set[start].timepoint, end] < 0$. Το γεγονός αυτό δείχνει πως όχι μόνο το διάστημα

   $$I = [support\_set[start].timepoint, end + win\_size * winId]$$

   δεν είναι πιθανοτικό, αλλά και ότι

   $$\nexists end' > end : I' = [support\_set[start].timepoint, end' + win\_size * winId]$$

   όπου το I' να είναι πιθανοτικό διάστημα. Πρώτα, ο αλγόριθμος ελέγχει εάν οι iterators έδειχναν σε ένα πιθανοτικό διάστημα στην προηγούμενη επανάληψη. Εάν ναι, ο sPIEC αναγνωρίζει εκείνο το διάστημα ως μέγιστο πιθανοτικό διάστημα, καθώς δεν κατόρθωσε να το επεκτείνει (γραμμές 11–12). Στη συνέχεια, εναλλάσει την τιμή του $flag$ σε false και αυξάνει τον δείκτη *start*, καθώς είναι βέβαιο πως δεν μπορεί να σχηματιστεί κάποιο επιπλέον πιθανοτικό διάστημα ξεκινώντας από το τρέχον στοιχείο του *support set*.

Έτσι, ο Αλγόριθμος 4 ξεκινά από την αρχή του *support set* και του τρέχοντος παραθύρου και ελέγχει εάν το αντίστοιχο διάστημα μπορεί να επεκταθεί σε ένα πιθανοτικό. Αν ναι, προσπαθεί να το επεκτείνει λαμβάνοντας υπόψη το αμέσως επόμενο χρονικό σημείο του παραθύρου ως πέρας του διαστήματος. Εάν όχι, προσθέτει το διάστημα της προηγούμενης επανάληψης στην έξοδο, αν ήταν πιθανοτικό, και επιχειρεί να εντοπίσει επιπλέον διαστήματα, εξετάζοντας το επόμενο στοιχείο του *support set*. Σημειώστε ότι εάν ο βρόχος τερματίσει σε ένα πιθανοτικό διάστημα, αυτό προστίθεται στην έξοδο μετά το τέλος του βρόχου (γραμμή 16).
Οι ομοιότητες του αλγορίθμου 4 και του PIEC είναι πλέον ακόμα πιο εμφανείς. Ο βρόγχος και των δύο αλγορίθμων ελέγχει την ίδια συνθήκη, οι δείκτες αυξάνονται αντιστοίχως και χρησιμοποιείται παρόμοια προϋπόθεση για τον εντοπισμό διαστημάτων. Η κύρια διαφορά τους είναι ότι ενώ ο PIEC ελέγχει για σημεία εκκίνησης και λήξης διαστημάτων σε μια κοινόχρηστη λίστα χρονικών σημείων, ο Αλγόριθμος 4 χρησιμοποιεί δύο ξεχωριστές λίστες. Προσπελαύνει το *support set* αναζητώντας σημεία εκκίνησης και το τρέχον παράθυρο ψάχνοντας για σημεία τερματισμού.

4.1.3.2 Πολυπλοκότητα

Επειδή σε κάθε επανάληψη είτε ο δείκτης *start* είτε ο *end* αυξάνεται, είναι σαφές ότι η πολυπλοκότητα του αλγορίθμου είναι $O(support\_set.length + dp.length)$. Ένας αφελής αλγόριθμος θα

επέστρεφε το ίδιο αποτέλεσμα συγκρίνοντας όλα τα ζεύγη στοιχείων των δύο λιστών εισόδου, επιτυγχάνοντας την πολυπλοκότητα $O(support\_set.length * dp.length)$. Ο Αλγόριθμος 4 πετυχαίνει καλύτερη πολυπλοκότητα επειδή εκμεταλλεύεται το γεγονός ότι τα στοιχεία και στις δύο λιστες βρίσκονται σε φθίνουσα σειρά (λαμβάνοντας υπόψη την τιμή *prefix_prev* κάθε πλειάδας στο *support set*). Πιο συγκεκριμένα, εάν η ανίσωση $support\_set[start].prefix\_prev > dp[end]$ ικανοποιείται, αυτή θα ισχύει και για κάθε ακόλουθο στοιχείο της λίστας dp. Επομένως, αυτές οι συγκρίσεις δεν εκτελούνται ποτέ και μόνο το *start* αυξάνεται. Η ίδια λογική εφαρμόζεται και για τη συνθήκη αύξησης του *end* (γραμμή 9).

4.1.3.3 Ορθότητα

Σε αυτή την ενότητα παρουσιάζουμε μία απόδειξη της ορθότητας του Αλγορίθμου 4. Πιο συγκεκριμένα, θα δείξουμε ότι κάθε διάστημα που προστίθεται στην έξοδο είναι ένα μέγιστο πιθανοτικό διάστημα και ότι δεν υπάρχει μέγιστο πιθανοτικό διάστημα που ξεκινάει από ένα χρονικό σημείο στο *support set* και τελειώνει στο τρέχον παράθυρο που να παραλείπει ο αλγόριθμος.
Παρουσιάζουμε πρώτα ένα θεώρημα που θα μας βοηθήσει να αποδείξουμε την παραπάνω πρόταση.

Θεώρημα 1. *Θεωρούμε:*

- $dp = [dp_0, dp_1, \cdots, dp_n]$
- $support\_set = [(t_0, s_0), (t_1, s_1), \cdots, (t_m, s_m)]$
- *Το διάστημα* $I = [t_{j'}, (i'-1) + win\_size * winId]$ *είναι ένα μέγιστο πιθανοτικό διάστημα, με* $t_{j'} \in support\_set$ *και* $i' \in [0, n]$.

Η εκτέλεση του Αλγορίθμου 4 θα φτάσει σε μια κατάσταση για την οποία $start = j'$ *και* $end = i' - 1$.

Απόδειξη. Επειδή το I είναι ένα μέγιστο πιθανοτικό διάστημα, ισχύει ότι:

$$dp_{i'-1} \geq s_{j'} \tag{4.4}$$

,

$$\forall k < j', l \geq i' - 1 : dp_l < s_k. \tag{4.5}$$

και

$$\forall k \leq j', l > i' - 1 : dp_l < s_k. \tag{4.6}$$

Οι συνθήκες 4.5 και 4.6 εγγυώνται ότι δεν υπάρχει πιθανοτικό διάστημα I' για το οποίο $I \subset I'$ και το I είναι επομένως μέγιστο.
Δεδομένου ότι ο βρόχος *while* του αλγορίθμου τερματίζεται όταν είτε ο *start* ή ο *end* έχει υπερβεί το μήκος της αντίστοιχης λίστας του, είναι βέβαιο ότι η εκτέλεση θα φτάσει σε κατάσταση όπου είτε $start = j'$ ή $end = i' - 1$.

1. Ας υποθέσουμε ότι το $end = i' - 1$ επιτυγχάνεται πρώτο, όταν $start = k < j'$. Λόγω της ανίσωσης 4.5, ισχύει ότι $dp_{i'-1} < s_k$. Επομένως, η συνθήκη της εντολής *if* στη γραμμή 7 αποτυγχάνει, αυξάνοντας το *start* ως αποτέλεσμα. Και πάλι λόγω της ανίσωσης 4.5, το *start* θα συνεχίσει να αυξάνεται μέχρι να φτάσει στην τιμή j'. Επομένως, θα επιτευχθεί η επιθυμητή κατάσταση.

2. Ας υποθέσουμε ότι το $start = j'$ επιτυγχάνεται πρώτο, όταν $end = l < i' - 1$. Λόγω της ανίσωσης 4.4 και του γεγονότος ότι η λίστα dp είναι φθίνουσα, ισχύει ότι $\forall l'$ όπου $l \leq l' \leq i' - 1 : dp_{l'} \geq s_{j'}$. Επομένως, σε κάθε επακόλουθη επανάληψη του βρόγχου, η συνθήκη της εντολής *if* στη γραμμή 7 πραγματοποιείται και ο δείκτης end αυξάνεται συνεχώς, φθάνοντας την τιμή $i' - 1$. Επομένως, θα επιτευχθεί η επιθυμητή κατάσταση.

□

Θεώρημα 2. *Όλα τα διαστήματα που προστίθενται στην έξοδο από τον Αλγόριθμο 4 είναι μέγιστα πιθανοτικά διαστήματα, υποθέτοντας ότι το support set περιέχει αυστηρώς όλα τα πιθανά σημεία εκκίνησης από προηγούμενα παράθυρα (εγγυημένο από τον Αλγόριθμο 5)*

Απόδειξη. Θεωρούμε:

- $dp = [dp_0, dp_1, \cdots, dp_n]$
- $support\_set = [(t_0, s_0), (t_1, s_1), \cdots, (t_m, s_m)]$
- Το διάστημα $I = [t_{j'}, (i' - 1) + win\_size * winId]$ έχει εντοπιστεί από τον Αλγόριθμο 4, με $t_{j'} \in support\_set$ και $i' \in [0, n]$.

Αποδεικνύοντας ότι το I είναι πιθανοτικό:
Για να φτάσει η εκτέλεση είτε στη γραμμή 12 ή στη 16, η μεταβλητή $flag$ πρέπει να είναι ίση με true. Η τιμή του $flag$ εξαρτάται από το αποτέλεσμα της εντολής *if* (γραμμή 7) στην προηγούμενη επανάληψη. Πιο συγκεκριμένα, $flag = true \iff dp_{end-1} \geq s_{start}$ αφού ο μόνος τρόπος το $flag$ να γίνει ίσο με true είναι να ήταν true η συνθήκη του if στην προηγούμενη επανάληψη. Έτσι, αν το I προστεθεί στην έξοδο στην επανάληψη στην οποία $start = j'$ και $end = i'$, ισχύει το εξής:

$$\begin{aligned} dp_{i'-1} &\geq s_{j'} && \iff \\ dp_{i'-1} &\geq prefix[t_{j'-1}] && \iff \\ dprange[t_{j'}, (i' - 1) + win\_size * winId] &\geq 0 && \iff \\ P(I) &\geq threshold \end{aligned}$$

Επομένως, το I είναι ένα πιθανοτικό διάστημα
Αποδεικνύοντας ότι το I είναι μέγιστο:

1. Έστω πως το I προστίθεται στην έξοδο στη γραμμή 12:
   Για να φτάσει η εκτέλεση σε αυτή τη γραμμή κώδικα, πρέπει να ισχύει ότι $dp_{i'} < s_{j'}$ και $flag = true$. Επομένως, $dp_{i'} < s_{j'} \leq dp_{i'-1}$. Επιπλέον, επειδή η λίστα dp είναι φθίνουσα, δεν περιέχει κανένα στοιχείο μετά το $dp_{i'}$ που να είναι μεγαλύτερο από την ποσότητα $s_{j'}$. Άρα, η ανίσωση $dp_k - s_{j'} \geq 0$ ισχύει για $k = i' - 1$ και δεν ισχύει για οποιοδήποτε $k \geq i'$. Άρα, δεν υπάρχει διάστημα $I' = [t_{j'}, k + win\_size * winId]$ με $k > i' - 1$ το οποίο είναι πιθανοτικό. Άρα, το διάστημα I δε μπορεί να επεκταθεί προς τα δεξιά.
   Έστω πως το I μπορεί να επεκταθεί προς τα αριστερά. Αυτό σημαίνει ότι:

$$\exists k < j' : s_k \leq dp_{i'-1} \tag{4.7}$$

$$s_k > dp_{i'} \tag{4.8}$$

Έστω ότι ο k είναι ο πρώτος δείκτης για τον οποίο ισχύουν οι παραπάνω ανισώσεις. Τότε, ισχύει ότι:

$$\forall k' < k, s_{k'} > dp_{i'-1}. \tag{4.9}$$

Επομένως, το $I'' = [t_k, (i'-1) + win\_size * winId]$ είναι ένα μέγιστο πιθανοτικό διάστημα. Εξ αιτίας του Θεωρήματος 1, η εκτέλεση του Αλγορίθμου 4 φτάνει σε μία κατάσταση για την οποία $start = k$ και $end = i' - 1$. Τώρα, λόγω της ανίσωσης 4.7, η συνθήκη του *if* θα πραγματοποιηθεί και ο end θα αυξηθεί. Στην επόμενη επανάληψη, όμως, η συνθήκη του *if* θα είναι ψευδής (Ανίσωση 4.7) και το διάστημα I' θα προστεθεί στην έξοδο. Επειδή η μεταβλητή $flag$ είναι false, είναι πλέον αδύνατο να προστεθεί ένα νέο διάστημα, χωρίς να αλλάξει η τιμή της $flag$ και να αυξηθεί ο end (γραμμές 8 και 9). Επομένως, ο sPIEC δεν μπορεί να προσθέσει άλλο διάστημα που τελειώνει στο δείκτη end στην έξοδο. Άρα, το I δεν θα προστεθεί στην έξοδο. Η πρόταση αυτή μας οδηγεί σε άτοπο λόγω της υπόθεσης. Άρα, $\nexists k < j : s_k \leq dp_{i-1}$, το διάστημα I'' δεν είναι πιθανοτικό και, επομένως, το διάστημα I δεν μπορεί να επεκταθεί προς τα αριστερά.
Άρα, το διάστημα I είναι μέγιστο.

2. Έστω πως το I προστίθεται στην έξοδο στη γραμμή 12:
Η εκτέλεση μπορεί να φτάσει σε αυτή τη γραμμή μόνο αν το $dp_{i'-1}$ είναι το τελευταίο στοιχείο της λίστας dp και $s_{j'} \leq dp_{i'-1}$. Είναι προφανές ότι το I δεν μπορεί να επεκταθεί προς τα δεξιά επειδή το $(i'-1)+win\_size*winId$ είναι το τελευταίο χρονικό σημείο του παραθύρου. Μπορεί να αποδειχθεί, όπως και στην προηγούμενη περίπτωση, ότι το I δεν μπορεί να επεκταθεί προς τα αριστερά.
Έτσι, το διάστημα I είναι μέγιστο.

□

Θεώρημα 3. *Ο Αλγόριθμος 4 δεν παραλείπει κάποιο μέγιστο πιθανοτικό χρονικό διάστημα που ξεκινάει από ένα χρονικό σημείο στο support set και τερματίζει στο τρέχον παράθυρο. Και πάλι, υποθέτουμε ότι το support set περιέχει αυστηρώς όλα τα πιθανά σημεία εκκίνησης από προηγούμενα παράθυρα (εγγυημένο από τον Αλγόριθμο 5).*

Απόδειξη. Θεωρούμε:

- $dp = [dp_0, dp_1, \cdots, dp_n]$
- $support\_set = [(t_0, s_0), (t_1, s_1), \cdots, (t_m, s_m)]$
- Το διάστημα $I = [t_{j'}, (i'-1) + win\_size * winId]$ είναι ένα μέγιστο πιθανοτικό διάστημα, με $t_{j'} \in support\_set$ και $i' \in [0, n]$.

Εφαρμόζοντας το Θεώρημα 1, συμπεραίνουμε ότι η εκτέλεση του αλγορίθμου 4 φτάνει σε μια κατάσταση στην οποία $start = j'$ και $end = i' - 1$. Στη συνέχεια, η συνθήκη της εντολής *if* στη γραμμή 7 ικανοποιείται λόγω ανίσωσης 4.4. Έτσι, ο δείκτης end αυξάνεται και η μεταβλητή $flag$ γίνεται true, υποδεικνύοντας ότι το I είναι ένα μέγιστο πιθανοτικό διάστημα. Στην επόμενη επανάληψη, οι τιμές των δεικτών είναι $start = j'$ και $end = i'$. Τώρα, η συνθήκη του *if* αποτυγχάνει

λόγω της ανίσωσης 4.6. Επομένως, το διάστημα I προστίθεται στην έξοδο. Έτσι, ο Αλγόριθμος 4 εντοπίζει κάθε μέγιστο πιθανοτικό διάστημα.

□

Άρα, συνδυάζοντας τα Θεωρήματα 2 και 3 αποδεικνύεται ότι όλα τα διαστήματα που εντοπίζονται από τον Αλγόριθμο 4 είναι μέγιστα πιθανοτικά διαστήματα και κανένα τέτοιο διάστημα δεν παραλείπεται.

4.1.4 Προσθήκη νέων πιθανών σημείων εκκίνησης στο support set

Όπως αναφέρθηκε προηγουμένως, είναι σημαντικό το *support set* να περιέχει κάθε πιθανό σημείο εκκίνησης, ώστε να διασφαλίζεται η ορθότητα του sPIEC, και να μην περιλαμβάνει κανένα περιττό χρονικό σημείο, για την βελτίωση της απόδοσης του αλγορίθμου και την οικονομία μνήμης. Έτσι, σε αυτή την ενότητα, παρουσιάζουμε έναν αλγόριθμο για την ανίχνευση κάθε πιθανού σημείου εκκίνησης καθώς εξελίσσεται η ροή δεδομένων. Αυτή η λειτουργία εκτελείται στο εσωτερικό στρώμα του sPIEC (Αλγόριθμος 3, γραμμή 5). Παρέχουμε επίσης σχολιασμό, ανάλυση πολυπλοκότητας και απόδειξη ορθότητας.

Algorithm 5 Εύρεση πιθανών σημείων εκκίνησης εντός του τρέχοντος παραθύρου

Είσοδος: Η λίστα prefix του τρέχοντος παραθύρου και οι μεταβλητές $last\_prefix$ και $ignore\_value$.
Έξοδος: Το ανανεωμένο *support set* που περιέχει όλα τα πιθανά σημεία εκκίνησης εντός του παραθύρου.
Απαιτεί: Τις μεταβλητές win_size, $winId$ και το αρχικό *support set*.

```
1: procedure FIND_POTENTIAL_STARTING_POINTS(prefix,last_prefix,ignore_value)
2:     for i ← 0 to win_size − 1 do
3:         if i == 0 then
4:             prefix_prev ← last_prefix
5:         else
6:             prefix_prev ← prefix[i − 1]
7:         if prefix_prev < ignore_value then
8:             support_set ⟸^add (i + winId ∗ win_size, prefix_prev)
9:             ignore_value ← prefix_prev
       return support_set
```

4.1.4.1 Επισκόπηση

Ο Αλγόριθμος 5 περιλαμβάνει έναν βρόχο *for* ο οποίος κάνει μία επανάληψη για κάθε χρονικό σημείο του τρέχοντος παραθύρου. Καταρχάς, ο αλγόριθμος υπολογίζει την τιμή *prefix_prev* του τρέχοντος χρονικού σημείου (γραμμές 3-6). Έπειτα, ελέγχει αν η τιμή αυτή είναι μικρότερη από την *ignore_value* (γραμμή 7). Αν ναι, προσθέτει μια πλειάδα που περιέχει το χρονικό σημείο που εξετάζεται μαζί με την τιμή *prefix_prev* που υπολογίστηκε στο *support set*. Στη συνέχεια, το *ignore_value* ενημερώνεται με την τρέχουσα τιμή *prefix_prev*, αν πραγματοποιήθηκε εισαγωγή.
Η δομή του Αλγορίθμου 5 μας θυμίζει τη δομή ενός αλγορίθμου επιλογής για την εύρεση του ελάχιστου στοιχείου μιας λίστας. Για κάθε νέα τιμή *prefix_prev*, ο αλγόριθμος ελέγχει εάν είναι μικρότερη από το μικρότερο *prefix_prev* που έχει βρεθεί προς το παρόν. Αν ναι, ο αλγόριθμος αποθηκεύει αυτή την τιμή στο *support set* και ενημερώνει το μικρότερο *prefix_prev* στην τιμή του τρέχοντος στοιχείου. Μετά την εκτέλεση του αλγορίθμου, το *ignore_value* θα είναι ίσο

με το ελάχιστο *prefix_prev* που έχει επεξεργαστεί μέχρι στιγμής και το *support set* θα περιέχει κάθε χρονικό σημείο του οποίου η τιμή *prefix_prev* είναι μικρότερη από την τιμή *prefix_prev* κάθε χρονικού σημείου προγενέστερού του.

4.1.4.2 Πολυπλοκότητα

Ο βρόχος του αλγορίθμου κάνει μία επανάληψη για κάθε χρονικό σημείο στο τρέχον παράθυρο. Έτσι, η πολυπλοκότητά του είναι $O(win\_size)$.

4.1.4.3 Ορθότητα

Για να έχει ο Αλγόριθμος 5 το επιθυμητό αποτέλεσμα, το *support set* πρέπει να περιέχει αυστηρώς όλα τα πιθανά σημεία εκκίνησης εντός του παραθύρου μετά την εκτέλεσή του. Έτσι, σε αυτήν την ενότητα, παρουσιάζουμε την απόδειξη ότι όλα τα χρονικά σημεία που περιέχονται στο *support set* είναι πιθανά σημεία εκκίνησης μέγιστων πιθανοτικών διαστημάτων και ότι κανένα τέτοιο χρονικό σημείο δεν παραλείπεται.
Πριν προχωρήσουμε στην απόδειξη της παραπάνω πρότασης, υπενθυμίζουμε ότι ο sPIEC χρησιμοποιεί την τιμή dprange δύο χρονικών σημείων για να προσδιορίσει αν αυτά συγκροτούν ένα πιθανοτικό διάστημα. Αυτή ορίζεται ως

$$dprange[start, end] = dp[end] - prefix[start - 1]. \tag{4.10}$$

Πιο συγκεκριμένα, εάν το $dprange[start, end]$ είναι μη αρνητικό, υπάρχει ένα διάστημα $I = [start, end']$ με $end' \geq end$ που έχει πιθανότητα μεγαλύτερη ή ίση με το πιθανοτικό κατώφλι. Ο sPIEC προσθέτει το διάστημα $[start, end]$ στην έξοδό του αν $dprange[start, end] \geq 0$ και $dprange[start, end + 1] < 0$ Έτσι, εγγυάται ότι ανιχνεύονται μόνο τα μέγιστα διαστήματα.
Ο Αλγόριθμος 5 εκμεταλλεύεται τον ορισμό του dprange για να διακρίνει τα πιθανά σημεία εκκίνησης στο τρέχον παράθυρο. Ειδικότερα, για κάθε χρονικό σημείο t_i, ορίζει την τιμή *prefix_prev* του ως την τιμή $prefix[t_i - 1]$ αφού αυτή είναι το μόνο στοιχείο του ορισμού 4.10 στο οποίο συμμετέχει το σημείο εκκίνησης. Για να αποτελέσει το t_i σημείο εκκίνησης ενός πιθανοτικού διαστήματος, πρέπει να είναι αληθές ότι για κάποιο μελλοντικό χρονικό σημείο t_j, ισχύει ότι $dp[t_j] \geq prefix\_prev[t_i]$ και $dp[t_j + 1] < prefix\_prev[t_i]$. Για να είναι το διάστημα αυτό μέγιστο, όλα τα χρονικά σημεία πριν από το t_i πρέπει να έχουν *prefix_prev* μεγαλύτερο από $dp[t_j]$. Άρα, εάν

$$\exists t_{i'} < t_i : prefix\_prev[t_{i'}] \leq prefix\_prev[t_i]$$

τότε εάν το διάστημα $I = [t_i, t_j]$ είναι πιθανοτικό και το διάστημα $I' = [t_{i'}, t_j]$ είναι υποχρεωτικά πιθανοτικό. Αυτό αποδεικνύει ότι κάθε πιθανοτικό διάστημα που ξεκινά από το t_i μπορεί να επεκταθεί στα αριστερά ώστε να εκκινεί από το $t_{i'}$ ενώ παραμένει πιθανοτικό. Δεδομένου ότι μας ενδιαφέρουν μόνο τα μέγιστα πιθανοτικά διαστήματα, το t_i δεν είναι ένα πιθανό σημείο εκκίνησης και δεν πρέπει να βρίσκεται στο *support set*. Με αυτή τη λογική αποκλείουμε χρονικά σημεία από το να είναι πιθανά σημεία εκκίνησης. Όλα τα υπόλοιπα πρέπει είναι παρόντα στο *support set*. Επομένως, το *support set* πρέπει να περιέχει όλα τα χρονικά σημεία των οποίων το *prefix_prev* είναι μικρότερο από αυτό κάθε προγενέστερου χρονικού σημείου. Πιο συγκεκριμένα, το χρονικό σημείο t είναι ένα πιθανό σημείο εκκίνησης αν και μόνο αν:

$$\forall t' < t : prefix\_prev[t] < prefix\_prev[t']. \tag{4.11}$$

Θα ακολουθήσουν δύο Θεωρήματα. Το πρώτο θα εγγυηθεί ότι το σημείο εκκίνησης κάθε μέγιστου πιθανοτικού διαστήματος ικανοποιεί την προϋπόθεση (4.11). Το επόμενο θα αποδείξει ότι ο Αλγόριθμος 5 προσθέτει μόνο τα χρονικά σημεία που ικανοποιούν τη συνθήκη 4.11 στο *support set*. Ο συνδυασμός αυτών των δύο θεωρημάτων αποδεικνύει ότι το *support set* που δημιουργήθηκε από τον Αλγόριθμο 5 περιέχει τις επιθυμητές πλειάδες.

Θεώρημα 4. *Έστω ότι το $I = [t_i, t_j]$ είναι ένα μέγιστο πιθανοτικό διάστημα. Το χρονικό σημείο t_i ικανοποιεί τη συνθήκη 4.11.*

Απόδειξη. Έστω πως το t_i δεν ικανοποιεί αυτή τη συνθήκη. Τότε,

$$\exists t_{i'} < t_i : prefix\_prev[t_{i'}] \leq prefix\_prev[t_i]. \tag{4.12}$$

Αληθεύει ότι

$$\begin{aligned} dprange[t_{i'}, t_j] &= dp[t_j] - prefix[t_{i'} - 1] \\ &= dp[t_j] - prefix\_prev[t_{i'}] \\ &\overset{4.12}{\geq} dp[t_j] - prefix\_prev[t_i] \\ &= dprange[t_i, t_j] \geq 0. \end{aligned}$$

επείδη το διαστημα $[t_i, t_j]$ είναι πιθανοτικό.
Έτσι, για το διάστημα $I' = [t_{i'}, t_j]$ ισχύει ότι:

- $P(I') \geq threshold$ και
- $I \subset I'$.

Άρα, το I δεν είναι μέγιστο πιθανοτικό διάστημα (Άτοπο).
Άρα, το σημείο t_i πρέπει να ικανοποιεί τη συνθήκη 4.11. □

Θεώρημα 5. *Όλα τα χρονικά σημεία που προσθέτονται στο support set από τον Αλγόριθμο 5 στη γραμμή 8 ικανοποιούν την συνθήκη 4.11. Υποθέτουμε ότι η μεταβλητή ignore_value έχει τη μέγιστη δυνατή τιμή στην αρχή της ροής δεδομένων.*

Απόδειξη. Ένα χρονικό σημείο t προστίθεται στο *support set* εάν και μόνο αν ικανοποιεί την συνθήκη της εντολής *if* στη γραμμή 7. Εάν πληρούται η προϋπόθεση αυτή, το *ignore_value* ενημερώνεται στην τιμή *prefix_prev* του τρέχοντος χρονικού σημείου. Δεδομένου ότι ο βρόχος του Αλγορίθμου 5 επεξεργάζεται όλα τα χρονικά σημεία μέσα στο τρέχον παράθυρο με χρονολογική σειρά, συμπεραίνουμε ότι στην i-οστή του επανάληψη, η μεταβλητή *ignore_value* είναι ίση με την ελάχιστη τιμή *prefix_prev* που έχει βρεθεί μέχρι τη χρονική στιγμή με δείκτη $i - 1$. Επομένως, τη τιμή *prefix_prev* του χρονικού σημείου t είναι μικρότερη από το *ignore_value* αν και μόνο αν ικανοποιεί τη συνθήκη 4.11.
Έτσι, ένα χρονικό σημείο t ικανοποιεί τη συνθήκη της γραμμής 7 αν και μόνο αν ικανοποιεί την συνθήκη 4.11. □

Συνδυάζοντας τα Θεωρήματα 4 και 5, συμπεραίνουμε ότι δεν μπορεί να υπάρξει ένα μέγιστο πιθανοτικό διάστημα στο ρεύμα εισόδου που ξεκινάει από ένα χρονικό σημείο ενός προηγούμενου παραθύρου που ο Αλγόριθμος 5 δεν εντάσσει στο *support set*.

4.2 Παράδειγμα

Σε αυτή την ενότητα παρουσιάζουμε ένα γενικό παράδειγμα της εκτέλεσης του sPIEC. Το πιθανοτικό κατώφλι που χρησιμοποιείται είναι ίσο με το 0.5 και το μέγεθος παραθύρου ισούται με 2 χρονικά σημεία. Η λίστα στιγμιαίων πιθανοτήτων εισόδου παρουσιάζεται παρακάτω:

Στιγμιαιες Πιθανοτητες Εισοδοτ

$Time$	0	1	2	3	4	5	6	7	8	9
A	0	0.5	0.7	0.9	0.4	0.1	0	0	0.5	1

Καταρχάς, ο sPIEC τεμαχίζει τον παραπάνω πίνακα σε παράθυρα μήκους ίσο με 2 χρονικά σημεία και επεξεργάζεται κάθε παράθυρο με χρονολογική σειρά.

Πρωτο Παραθυρο

$Time$	0	1
A	0	0.5
L	-0.5	0
$prefix$	-0.5	-0.5
dp	-0.5	-0.5

support set: Κενό
ignore_value:

1. Εύρεση μέγιστων πιθανοτικών διαστημάτων με βάση τα στοιχεία του *support set*
   - Εφόσον το *support set* είναι κενό, δεν εντοπίζεται κάποιο διάστημα.
2. Ο αρχικός αλγόριθμος PIEC εκτελείται στο παραπάνω παράθυρο.
   - Εντοπίζεται το διάστημα $[1, 1]$.
3. Εύρεση πιθανών σημείων εκκίνησης εντός του τρέχοντος παραθύρου.
   - Δεδομένου ότι δεν έχει επεξεργαστεί κανένα άλλο χρονικό σημείο, το χρονικό σημείο 0 έχει, αυτόματα, το μικρότερο *prefix_prev* και επομένως προστίθεται στο *support set* με το *prefix_prev* $= 0$. Έπειτα, η μεταβλητή *ignore_value* ανανεώνεται στην τιμή 0.
   - Το χρονικό σημείο 1 προστίθεται στο *support set* με πρόθεμα *prefix_prev* $= 0.5$ επειδή είναι το ελάχιστο που έχει βρεθεί προς το παρόν. Έπειτα, η μεταβλητή *ignore_value* ανανεώνεται στην τιμή -0.5.

Δευτερο Παραθυρο

$Time$	2	3
A	0.7	0.9
L	0.2	0.4
$prefix$	-0.3	0.1
dp	0.1	0.1

support set: [(0, 0), (1, -0.5)]
ignore_value: -0.5

1. Εύρεση μέγιστων πιθανοτικών διαστημάτων με βάση τα στοιχεία του *support set*

   - Εντοπίζεται το διάστημα $[0, 3]$.
   - Αν και το *prefix_prev* του χρονικού σημείου 1 είναι μικρότερο από την τιμή dp του χρονικού σημείου 3, το διάστημα $[1, 3]$ δεν βρέθηκε επειδή ο αλγόριθμος ανιχνεύει ότι το χρονικό σημείο 3 έχει ήδη χρησιμοποιηθεί ως τελικό σημείο ενός διαστήματος σε μια προηγούμενη επανάληψη. Δεδομένου ότι τα στοιχεία του *support set* είναι σε αύξουσα χρονική σειρά, είναι βέβαιο το διάστημα αυτό θα είναι μέγιστο και όλα τα πιθανά υποδιαστήματα του παραλείπονται.

2. Ο αρχικός αλγόριθμος PIEC εκτελείται στο παραπάνω παράθυρο.

   - Εντοπίζεται το διάστημα $[2, 3]$.
   - Επειδή $[2, 3] \subset [0, 3]$, το διάστημα $[2, 3]$ θα διαγραφεί αργότερα (Αλγόριθμος 2, γραμμή 5).

3. Εύρεση πιθανών σημείων εκκίνησης εντός του τρέχοντος παραθύρου.

   - Δεδομένου ότι για κάθε χρονικό σημείο σε αυτό το παράθυρο $prefix\_prev \geq ignore\_value$, το *support set* δεν μεταβάλλεται.

Τριτο Παραθυρο

$Time$	4	5
A	0.4	0.1
L	-0.1	-0.4
$prefix$	0	-0.4
dp	0	-0.4

support set: [(0, 0), (1, -0.5)]
ignore_value: -0.5

1. Εύρεση μέγιστων πιθανοτικών διαστημάτων με βάση τα στοιχεία του *support set*

   - Εντοπίζεται το διάστημα $[0, 4]$ επειδή $dp_4 \geq prefix\_prev_0$.
     Αυτό το διάστημα δεν μπορεί να επεκταθεί για να συμπεριλάβει το χρονικό σημείο 5 επειδή η τιμή dp είναι μικρότερη από το *prefix_prev* του χρονικού σημείου 0.
   - Εντοπίζεται το διάστημα $[1, 5]$ επειδή $dp_5 \geq prefix\_prev_1$.

2. Ο αρχικός αλγόριθμος PIEC εκτελείται στο παραπάνω παράθυρο.

   - Δεν εντοπίζεται κάποιο διάστημα.

3. Εύρεση πιθανών σημείων εκκίνησης εντός του τρέχοντος παραθύρου.

   - Δεδομένου ότι για κάθε χρονικό σημείο σε αυτό το παράθυρο $prefix\_prev \geq ignore\_value$, το *support set* δεν μεταβάλλεται.

Τεταρτο Παραθυρο

$Time$	6	7
A	0	0
L	-0.5	-0.5
$prefix$	-0.9	-1.4
dp	-0.9	-1.4

support set: [(0, 0), (1, -0.5)]
$ignore\_value$: -0.5

1. Εύρεση μέγιστων πιθανοτικών διαστημάτων με βάση τα στοιχεία του *support set*.
   - Δεν εντοπίζεται κάποιο διάστημα.
2. Ο αρχικός αλγόριθμος PIEC εκτελείται στο παραπάνω παράθυρο.
   - Δεν εντοπίζεται κάποιο διάστημα.
3. Εύρεση πιθανών σημείων εκκίνησης εντός του τρέχοντος παραθύρου.
   - Το χρονικό σημείο 6 δεν είναι πιθανό σημείο εκκίνησης επειδή $prefix\_prev_6 = prefix_{6-1} = prefix_5 = -0.4 > -0.5 = ignore\_value$.
   - Το χρονικό σημείο 7 είναι πιθανό σημείο εκκίνησης επειδή $prefix\_prev_7 = prefix_{7-1} = prefix_6 = -0.9 < -0.5 = ignore\_value$.
     Έτσι, το χρονικό σημείο 7 εισάγεται στο *support set* με τιμή $prefix\_prev = -0.9$. Η μεταβλητή $ignore\_value$ ανανεώνεται στην τιμή -0.9.

Πεμπτο Παραθυρο

$Time$	8	9
A	0.5	1
L	0	0.5
$prefix$	-1.4	-0.9
dp	-0.9	-0.9

support set: [(0, 0), (1, -0.5), (7, -0.9)]
$ignore\_value$: -0.9

1. Εύρεση μέγιστων πιθανοτικών διαστημάτων με βάση τα στοιχεία του *support set*
   - Εντοπίζεται το διάστημα $[7, 9]$ επειδή $dp_9 \geq prefix\_prev_7$
2. Ο αρχικός αλγόριθμος PIEC εκτελείται στο παραπάνω παράθυρο.
   - Εντοπίζεται το διάστημα $[8, 9]$.
   - Επειδή $[8, 9] \subset [7, 9]$, το διάστημα $[8, 9]$ θα διαγραφεί αργότερα (Αλγόριθμος 2, γραμμή 5).
3. Εύρεση πιθανών σημείων εκκίνησης εντός του τρέχοντος παραθύρου.

- Το χρονικό σημείο 8 είναι ένα πιθανό σημείο εκκίνησης, καθώς η τιμή *prefix_prev* του είναι μικρότερη από την $ignore\_value$. Έτσι, το χρονικό σημείο 8 προστίθεται στο *support set* με τιμή $prefix\_prev = -1.4$. Η μεταβλητή $ignore\_value$ ανανεώνεται στην τιμή -1.4.
- Το χρονικό σημείο 9 δεν προστίθεται στο *support set* επειδή η τιμή $prefix\_prev = -1.4$ δεν είναι μικρότερη από την ανανεωμένη τιμή της $ignore\_value$.
- Παρόλο που αυτό είναι τελευταίο παράθυρο της ροής εισόδου, είναι σημαντικό να συνεχίσουμε να προσθέτουμε χρονικά σημεία στο *support set* για να διασφαλίσουμε την ορθότητα της εκτέλεσης του sPIEC σε περίπτωση εμφάνισης περισσότερων δεδομένων εισόδου.

Κεφάλαιο 5

Αλγόριθμος Περιορισμένης Μνήμης

Γνωρίζουμε ότι στα πραγματικά συστήματα τα δεδομένα που αποθηκεύονται κατά την εκτέλεση ενός αλγορίθμου ροής δεδομένων δεν μπορούν να καταλαμβάνουν άπειρη μνήμη. Το γεγονός αυτό είναι κρίσιμο για τα αποτελέσματα του sPIEC καθώς βασίζεται στην αποθήκευση κάθε πιθανού σημείου εκκίνησης από εκτελέσεις προηγούμενων παραθύρων στο *support set* για να εγγυηθεί την ανίχνευση κάθε μέγιστου πιθανοτικού διαστήματος. Σε αυτό το κεφάλαιο, αναλύουμε τις παρενέργειες στην ακρίβεια του sPIEC που προκαλούνται από τη χρήση ενός *support set* περιορισμένου μεγέθους. Επιπλέον, προτείνουμε τέσσερις μεθόδους για την επιλογή του στοιχείου του *support set* προς διαγραφή σε περίπτωση γεμάτης μνήμης εργασίας. Τέλος, θα αποδείξουμε ότι η εφαρμογή περιορισμένης μνήμης μπορεί να βλάψει μόνο τη μετρική «ανάκλησης» (recall) του αλγορίθμου, σε σύγκριση με τα αποτελέσματα του sPIEC με άπειρη μνήμη ή, ομοίως, με τα μέγιστα πιθανοτικά διαστήματα του PIEC.

5.1 Επισκόπηση Ακρίβειας Αλγορίθμου

Σε αυτή την ενότητα, περιγράφουμε τα αποτελέσματα της χρήσης *support set* περιορισμένης μνήμης στην έξοδο του αλγορίθμου. Συλλογιζόμαστε, επίσης, την επίπτωση της στις μετρικές «ακρίβειας» (precision) και «ανάκλησης» (recall) των διαστημάτων του sPIEC σε σύγκριση με τα πραγματικά μέγιστα πιθανοτικά διαστήματα της εισόδου.
Πρώτα, παρουσιάζουμε μια ανάλυση του μεγέθους του *support set* με δεδομένη τη λίστα εισόδου στιγμιαίων πιθανοτήτων A και του πιθανοτικού κατωφλιού T. Όπως περιγράφηκε στο προηγούμενο κεφάλαιο (Ενότητα 4.1.4.3), το *support set* περιέχει ακριβώς εκείνα τα χρονικά σημεία των οποίων οι τιμές *prefix_prev* είναι μικρότερες από οποιαδήποτε τιμή *prefix_prev* που είναι χρονικά προγενέστερή της. Λόγω του ορισμού του *prefix_prev* (Ορισμός 3), είναι φανερό ότι το μέγεθος του *support set* εξαρτάται από το πλήθος των τρέχουσων ελάχιστων τιμών της λίστας prefix που παράγεται από η λίστα εισόδου A και το κατώφλι T και είναι ανεξάρτητη από το μέγεθος του παραθύρου. Αυτή η τιμή ταύτιζεται με το πλήθων των στοιχείων της λίστας prefix που έχουν μικρότερη τιμή από κάθε στοιχείο που τα προηγείται. Επομένως, εάν ο πίνακας prefix είναι σε φθίνουσα σειρά, το *support set* περιέχει κάθε χρονικό σημείο και έχει μέγεθος ίσο με την είσοδο. Από την άλλη πλευρά, αν ο

πίνακας prefix είναι σε αύξουσα σειρά, *support set* θα συγκρατήσει μόνο το πρώτο χρονικό σημείο. Επομένως, το μέγεθος του *support set* είναι ίσο με το πλήθος των τρέχοντων ελάχιστων τιμών στον πίνακα prefix και είναι γενικά ανεξάρτητο από το μέγεθος της εισόδου.
Τώρα, υποθέτουμε ότι το *support set* έχει περιορισμένο μέγεθος. Σε αυτή την περίπτωση, τα αποτελέσματα του sPIEC περιορισμένης μνήμης διαφέρουν από αυτά του ίδιου αλγορίθμου με άπειρη μνήμη. Για λόγους διαύγειας, εφεξής θα αναφερόμαστε στον αλγόριθμο περιορισμένης μνήμης ως sPIEC$^{\mathsf{b}}$.
Όπως θα αποδείξουμε αυστηρά στην ενότητα 5.3, ο sPIEC$^{\mathsf{b}}$ μπορεί να παράξει μόνο δύο είδη σφαλμάτων λόγω έλλειψης καταχωρήσεων στο *support set*:

1. Εμφανίζονται εσφαλμένα χρονικά διαστήματα στην έξοδο του sPIEC$^{\mathsf{b}}$. Αυτά τα διαστήματα έχουν σωστά σημεία τερματισμού αλλά λανθασμένα σημεία εκκίνησης και είναι πάντοτε υποδιαστήματα ενός μέγιστου πιθανοτικού διαστήματος. Αυτό συμβαίνει λόγω της απουσίας αυτών των σημείων εκκίνησης από το *support set*.
   Έστω ότι το $I = [t_i, t_j]$ είναι ένα μέγιστο πιθανοτικό διάστημα και ότι τα t_i και t_j εμφανίζονται σε διαφορετικά παράθυρα στην εκτέλεση του sPIEC$^{\mathsf{b}}$. Ας υποθέσουμε επίσης ότι το t_i διαγράφηκε από το *support set* πριν προλάβει ο sPIEC$^{\mathsf{b}}$ να επεξεργαστεί το t_j. Εάν υπάρχει ένα χρονικό σημείο t μετά το t_i στο *support set* του οποίου η τιμή *prefix_prev* είναι μικρότερη ή ίση με την τιμή dp του t_j, τότε ο sPIEC$^{\mathsf{b}}$ εντοπίζει διάστημα $I' = [t, t_j]$, χάνοντας τα χρονικά σημεία στο εύρος $[t_i, t)$. Στην περίπτωση που:

   $$\exists t' > t_j : \text{το διάστημα } I'' = [t, t'] \text{ να είναι ένα μέγιστο πιθανοτικό διάστημα,}$$

   ο sPIEC$^{\mathsf{b}}$ επεκτείνει το I' αναλόγως. Διαφορετικά, προσθέτει το I' στην έξοδο.

2. Μερικά μέγιστα πιθανοτικά διαστήματα λείπουν εξ ολοκλήρου από τα αποτελέσματα του sPIEC$^{\mathsf{b}}$. Υποθέστε το μέγιστο πιθανοτικό διάστημα $I = [t_i, t_j]$ του προηγούμενου παραδείγματος και ότι το t_i διαγράφηκε από το *support set* πριν προλάβει ο sPIEC$^{\mathsf{b}}$ να επεξεργαστεί το t_j. Εάν δεν υπάρχει χρονικό σημείο t μετά από t_i στο *support set* του οποίου η τιμή *prefix_prev* είναι μικρότερη ή ίση με την τιμή dp του t_j, τότε κανένα διάστημα που λήγει στο t_j δεν εμφανίζεται στην έξοδο.

Λαμβάνοντας υπόψη τα παραπάνω, η σύγκριση των διαστημάτων που εντοπίστηκαν από τον αλγόριθμο sPIEC$^{\mathsf{b}}$ με το σύνολο όλων των μέγιστων πιθανοτικών διαστημάτων της εισόδου, μας κατευθύνει στο συμπέρασμα ότι ο sPIEC$^{\mathsf{b}}$ προσθέτει στην έξοδο μόνο χρονικά σημεία που είναι μέρος ενός μέγιστου πιθανοτικού διαστήματος. Έτσι, η «ακρίβειά» του είναι τέλεια. Αντίθετα, μερικές φορές χάνει χρονικά σημεία που ανήκουν σε κάποιο μέγιστο πιθανοτικό διάστημα ή αγνοεί κάποια μέγιστα πιθανοτικά διαστήματα ολοκληρωτικά. Αυτό επηρεάζει τη μετρική «ανάκλησης» της σύγκρισης.

5.2 Στρατηγικές Συντήρησης Μνήμης

Σε αυτή την ενότητα περιγράφουμε τις μεθόδους που εφαρμόσαμε για να αποφασίσουμε ποιο στοιχείο του *support set* να διαγράφουμε κάθε φορά που γεμίζει η μνήμη εργασίας. Σε έναν τέλειο κόσμο, θα μπορούσαμε να «μαντέψουμε» ποια σημεία εκκίνησης θα χρησιμοποιηθούν από μεταγενέστερα παράθυρα και να τα διατηρούσαμε στο *support set*. Η τέλεια στρατηγική συντήτησης

μνήμης θα διαγράφει πάντα το στοιχείο του οποίου η αφαίρεση θα προκαλούσε τη μικρότερη ζημειά στην μετρική ανάκλησης. Φυσικά, δεν μπορούμε να «μαντεύουμε» στην πράξη. Αντ΄ αυτού, εφαρμόζουμε ορισμένες στρατηγικές διαγραφής και δοκιμάζουμε πειραματικά την απόδοσή τους.

Στρατηγικές Διαγραφής:

1. Αφελης Τακτικη: Διαγραφή του παλαιότερου στοιχείου του *support set* (το πρώτο στοίχειο της λίστας).
2. Τακτικη Χρονου: Διαγραφή του στοιχείου του οποίου η χρονική διαφορά με το επόμενο στη λίστα είναι ελάχιστη. Αυτή η τακτική θα ελαχιστοποιήσει τη χρονική διαφορά μεταξύ των λανθασμένα αρχικοποιημένων διαστημάτων και των αντίστοιχων σωστών. Η μέθοδος αυτή οδηγεί σε αυξημένη ανάκληση, δεδομένου ότι αυτή υπολογίζεται ανά χρονικό σημείο.
3. Τακτικη Σκορ: Διαγραφή του στοιχείου του οποίου το *prefix_prev* έχει τη μικρότερη διαφορά με το *prefix_prev* του αμέσως προγενέστερου στοιχείου του *support set*. Αυτό μπορεί να μεταφραστεί ως τη διαγραφή του στοιχείου με το μικρότερο διάστημα τιμών dp που χρειάζεται να εμφανιστούν σε μελλοντικά παράθυρα για να πυροδοτηθεί η δημιουργία ενός πιθανοτικού διαστήματος που ξεκινάει από το συγκεκριμένο σημείο εκκίνησης. Αναμένουμε ότι αυτή η τακτική θα αυξήσει την ανάκληση, καθώς τα σημεία εκκίνησης με μικρότερα διαστήματα αποδοχής dp είναι λιγότερο πιθανό να εμφανιστούν ως σημεία εκκίνησης μέγιστων πιθανοτικών διαστημάτων.
4. Ένας σταθμισμένος συνδυασμός της Τακτικής Χρόνου και της Τακτικής Σκορ. Αυτή η τακτική περιλαμβάνει την εύρεση χρονικών διαφορών και διαστημάτων αποδοχής dp κάθε στοιχείου στο *support set*, την εξομάλυνση αυτών των τιμών και τον υπολογισμό του σταθμισμένου αθροίσματος των κανονικοποιημένων πινάκων. Το στοιχείο που αντιστοιχεί στην ελάχιστη τιμή της προκύπτουσας μήτρας διαγράφεται. Τα βάρη που χρησιμοποιούνται σε αυτή την τακτική δίδονται στον αλγόριθμο ως είσοδοι και θα ήταν καλό να βελτιστοποιούνται.

Για να εξηγήσουμε τη λογική πίσω από τους ορισμούς των Τακτικών Χρόνου και Σκορ, παρουσιάζουμε δύο παραδείγματα εκτελέσεων του sPIEC$^{\text{b}}$. Κάθε παράδειγμα παρουσιάζει μια περίπτωση στην οποία ευδοκιμεί η αντίστοιχη τακτική. Και οι δύο εκτελέσεις χρησιμοποιούν τις ακόλουθες παραμέτρους:

- πιθανοτικό κατώφλι = 0.5
- μέγεθος παραθύρου = 2 στοιχεία
- μέγεθος ενεργής μνήμης = 2 στοιχεία

5.2.1 Τακτική Χρόνου – Παράδειγμα

Στιγμιαιες Πιθανοτητες Εισοδου και Δομες Δεδομενων

$Time$	**0**	**1**	2	**3**	**4**	5	6	7
A	0.3	0.5	0.3	0.3	0.6	0.7	0.7	0.7
L	−0.2	0	−0.2	−0.2	0.1	0.2	0.2	0.2
$prefix$	−0.2	−0.2	−0.4	−0.6	−0.5	−0.3	−0.1	0.1
dp	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1
$prefix\_prev$	**0**	**-0.2**	−0.2	**-0.4**	**-0.6**	−0.5	−0.3	−0.1

- Τα δυνητικά σημεία εκκίνησης των μέγιστων πιθανοτικών διαστημάτων καθορίζονται με τον υπολογισμό των χρονικών σημείων με τις τρέχουσες ελάχιστες τιμές *prefix_prev*. Αυτά τα χρονικά σημεία έχουν επισημανθεί με έντονη σκίαση στον πίνακα εισόδου και είναι τα μόνα χρονικά σημεία που προστίθενται στο *support set* κατά την εκτέλεση.

- Λαμβάνοντας υπόψη την είσοδο και το κατώφλι, συμπεραίνουμε ότι το μοναδικό μέγιστο πιθανοτικό διάστημα είναι το $[0, 7]$.

Πρωτο Παραθυρο

$Time$	**0**	**1**
$prefix\_prev$	**0**	**-0.2**

Το *support set* μετά την επεξεργασία του παραθύρου:
$[(0, 0), (1, -0.2)]$

Δεδομένου ότι το *support set* μπορεί να κρατήσει δύο στοιχεία, δεν υπάρχει υπερχείλιση μετά την επεξεργασία του πρώτου παραθύρου.

Δευτερο Παραθυρο

$Time$	2	**3**
$prefix\_prev$	-0.2	**-0.4**

Το *support set* μετά την επεξεργασία του παραθύρου:
Αφελης Τακτικη: $[(1, -0.2), (3, -0.4)]$
Τακτικη Χρονου: $[(1, -0.2), (3, -0.4)]$

- Η Αφελής Τακτική χειρίζεται την υπερχείλιση μνήμης διαγράφοντας το χρονικό σημείο 0 από το *support set* και αντικαθιστώντας το με το πιο πρόσφατο χρονικό σημείο 3.

- Η Τακτική Χρόνου υπολογίζει τη χρονική διαφορά κάθε στοιχείου με το ακόλουθο στοιχείο στο *support set*. Στη συνέχεια, διαγράφει το στοιχείο με την ελάχιστη τέτοια διαφορά.
Σε αυτή την περίπτωση, έχουμε το *support set* $[(0, 0), (1, -0, 2)]$ και το νέο στοιχείο $(3, -0, 4)$. Οι χρονικές διαφορές των χρονικών σημείων 0 και 1 είναι 1 και 2 χρονικά σημεία, αντίστοιχα. Αυτή η μέτρηση υποδεικνύει ότι η αφαίρεση του στοιχείου $(0, 0)$ θα προκαλούσε την απουσία ενός χρονικού σημείου από τα προκύπτοντα διαστήματα που θα ξεκινούσαν από το 0. Ομοίως, η αφαίρεση του $(1, -0, 2)$ θα κοστίσει 2 χρονικά σημεία από τα διαστήματα που θα ξεκινούσαν από το χρονικό σημείο 1. Επομένως, η Τακτική Χρόνου διαγράφει το στοιχείο $(0, 0)$ για να ελαχιστοποιήσει αυτό το σφάλμα.

Τριτο Παραθυρο

$Time$	**4**	5
$prefix\_prev$	**-0.6**	-0.5

Το *support set* μετά την επεξεργασία του παραθύρου:
Αφελης Τακτικη: $[(3, -0.4), (4, -0.6)]$
Τακτικη Χρονου: $[(1, -0.2), (4, -0.6)]$

- Η Αφελής Τακτική διαγράφει το στοιχείο $(1, -0.2)$ και προσθέτει το $(4, -0.6)$ στο *support set*.
- Η Τακτική Χρόνου διαγράφει το στοιχείο $(3, -0.4)$ επειδή η χρονική διαφορά του με $(4, -0.6)$ είναι 1 χρονικό σημείο. Στη συνέχεια, προσθέτει $(4, -0.6)$ στο *support set*.

Τεταρτο Παραθυρο

$Time$	6	7
dp	0.1	0.1

Διαστηματα της Αφελης Τακτικης: $[(3, 7)]$
Διαστηματα της Τακτικης Χρονο: $[(1, 7)]$

Επειδή $dp[7] = 0.1$, ο sPIEC$^{\text{b}}$ κατασκευάζει ένα διάστημα που αρχίζει από το πρώτο στοιχείο του *support set*, έστω $(t, prefix\_prev[t])$, για το οποίο $prefix\_prev[t] \leq 0.1$ και τελειώνει στο χρονικό σημείο 7.

- Ο sPIEC$^{\text{b}}$ με την Αφελή Τακτική έχει κρατήσει το *support set*: $[(3, -0.4), (4, -0.6)]$. Έτσι, πυροδοτείται το στοιχείο $(3, -0.4)$ και εντοπίζεται το διάστημα $[3, 7]$.
- Ο sPIEC$^{\text{b}}$ με την Τακτική Χρόνου έχει κρατήσει το *support set*: $[(1, -0.2), (4, -0.6)]$. Έτσι, πυροδοτείται το στοιχείο $(1, -0.2)$ και εντοπίζεται το διάστημα $[1, 7]$.

Συμπέρασμα

Το βέλτιστο αποτέλεσμα θα ήταν ο sPIEC$^{\text{b}}$ να ανιχνεύσει το μέγιστο πιθανοτικό διάστημα $[0, 7]$. Ωστόσο, καμία από τις τακτική δεν διατήρησε το χρονικό σημείο 0 στο *support set*. Έτσι, δεν βρέθηκε το σωστό διάστημα.
Η εκτέλεση του sPIEC$^{\text{b}}$ με την Τακτική Χρόνου ήταν ελαφρώς ανακριβής, χάνοντας μόνο ένα χρονικό σημείο. Αυτό επιτεύχθηκε χάρις την παρακράτηση του χρονικού σημείου 1 στο *support set*. Η συγκριτικά μεγαλύτερη τιμή του στην μετρική χρονικής διαφοράς που χρησιμοποιούμε υποδεικνύει ότι η διαγραφή του θα είχε μεγαλύτερο αντίκτυπο στην ακρίβεια της εκτέλεσης και συνεπώς κρατήθηκε στο *support set*. Αντίθετα, η εκτέλεση του sPIEC$^{\text{b}}$ με την Αφελή Τακτική αντικατέστησε αυτό το στοιχείο και, συνεπώς, χάθηκαν 3 χρονικά σημεία του σωστού διαστήματος.

5.2.2 Τακτική Σκορ – Παράδειγμα

Στιγμιαιες Πιθανοτητες Εισοδου και Δομες Δεδομενων

$Time$	**0**	**1**	**2**	**3**	4	5
A	0	0.3	0.3	0.6	0.4	1
L	−0.5	−0.2	−0.2	0.1	−0.1	0.5
$prefix$	−0.5	−0.7	−0.9	−0.8	−0.9	−0.4
dp	−0.4	−0.4	−0.4	−0.4	−0.4	−0.4
$prefix\_prev$	**0**	**-0.5**	**-0.7**	**-0.9**	−0.8	−0.9

- Τα δυνητικά σημεία εκκίνησης των μέγιστων πιθανοτικών διαστημάτων καθορίζονται με τον υπολογισμό των χρονικών σημείων με τις τρέχουσες ελάχιστες τιμές *prefix_prev*. Αυτά τα χρονικά σημεία έχουν επισημανθεί με έντονη σκίαση στον πίνακα εισόδου και είναι τα μόνα χρονικά σημεία που προστίθενται στο *support set* κατά την εκτέλεση.

- Λαμβάνοντας υπόψη την είσοδο και το κατώφλι, συμπεραίνουμε ότι το μοναδικό μέγιστο πιθανοτικό διάστημα είναι το $[1, 5]$.

Πρωτο Παραθυρο

$Time$	**0**	**1**
$prefix\_prev$	**0**	**-0.5**

Το *support set* μετά την επεξεργασία του παραθύρου:
$[(0, 0), (1, -0.5)]$

Δεδομένου ότι το *support set* μπορεί να κρατήσει δύο στοιχεία, δεν υπάρχει υπερχείλιση μετά την επεξεργασία του πρώτου παραθύρου.

Δευτερο Παραθυρο

$Time$	**2**	**3**
$prefix\_prev$	**-0.7**	**-0.9**

Το *support set* μετά την επεξεργασία του παραθύρου:
Αφελης Τακτικη: $[(2, -0.7), (3, -0.9)]$
Τακτικη Σκορ: $[(0, 0), (1, -0.5)]$

- Η Αφελής Τακτική διαγράφει διαδοχικά τα χρονικά σημεία 0 και 1 από το *support set* και προσθέτει τα χρονικά σημεία του δεύτερου παραθύρου.

- Η Τακτική Σκορ ορίζει την βαθμολογία κάθε στοιχείου ως την διαφορά *prefix_prev* του με τον αμέσως προγενέστερο του στοιχείο και διαγράφει το στοιχείο με την ελάχιστη βαθμολογία. Σε αυτό το παράδειγμα, το *support set* είναι το $[(0, 0), (1, -0, 5)]$ και το νέο στοιχείο είναι το $(2, -0, 7)$. Οι βαθμολογίες τους είναι $+\infty$, 0.5 και 0.2 αντίστοιχα (το χρονικό σημείο 0 δεν έχει προγενέστερο). Αυτές οι βαθμολογίες αντιστοιχούν στο μέγεθος του εύρους τιμών dp που πυροδοτούν το κάθε στοιχείο. Για παράδειγμα, αν ένα χρονικό σημείο t με $-0.5 \leq dp[t] < 0$ εμφανιστεί στη ροή δεδομένων, τότε πυροδοτείται το χρονικό σημείο 1 επειδή δέχεται το εύρος τιμών dp $[-0.5, 0)$ και δημιουργείται ένα πιθανοτικό διάστημα που εκκινεί από το χρονικό σημείο 1.
  Διαγράφουμε το στοιχείο με την μικρότερη βαθμολογία καθώς έχει μικρότερο εύρος τιμών dp και, επομένως, είναι λιγότερο πιθανό να πυροδοτηθεί. Έτσι, το στοιχείο $(2, -0.7)$ αφαιρείται από το *support set*. Στη συνέχεια, το στοιχείο $(3, -0.9)$ προκαλεί υπερχείλιση μνήμης. Η βαθμολογία του είναι $score[3] = -0, 5 - (-0, 9) = 0, 4$, η χαμηλότερη από οποιοδήποτε στοιχείο. Επομένως, το $(3, -0.9)$ δεν προστίθεται και το *support set* παραμένει ίδιο, όπως στο προηγούμενο παράθυρο.

Τριτο Παραθυρο

$Time$	4	5
dp	-0.4	-0.4

Διαστηματα της Αφελης Τακτικη: $[2, 5]$
Διαστηματα της Τακτικης Σκορ: $[1, 5]$

Επειδή $dp[5] = -0.4$, ο sPIEC$^{\text{b}}$ κατασκευάζει ένα διάστημα που αρχίζει από το πρώτο στοιχείο του *support set*, έστω $(t, prefix\_prev[t])$, για το οποίο $prefix\_prev[t] \leq -0.4$ και τελειώνει στο χρονικό σημείο 5.

- Ο sPIEC$^{\text{b}}$ με την Αφελή Τακτική έχει κρατήσει το *support set*: $[(2, -0.7), (3, -0.9)]$. Έτσι, πυροδοτείται το στοιχείο $(2, -0.7)$ και εντοπίζεται το διάστημα $[2, 5]$.
- Ο sPIEC$^{\text{b}}$ με την Τακτική Σκορ έχει κρατήσει το *support set*: $[(0, 0), (1, -0.5)]$. Έτσι, πυροδοτείται το στοιχείο $(1, -0.5)$ και εντοπίζεται το διάστημα $[1, 5]$.

Συμπέρασμα

Η Τακτική Σκορ διατήρησε το χρονικό σημείο 1 στο *support set* καθώς θα μπορούσε να ενεργοποιηθεί από ένα μεγαλύτερο εύρος τιμών dp, $[-0.5, 0)$, συγκριτικά με τα χρονικά σημεία 2 και 3. Η απόφασή του δικαιολογήθηκε από τα δεδομένα του τρίτου παραθύρου, αφού μία τιμή dp ίση με -0.4 εμφανίστηκε στη ροή δεδομένων. Ως εκ τούτου, ο sPIEC$^{\text{b}}$ με την Τακτική Σκορ ήταν σε θέση να συμπεράνει το σωστό μέγιστο πιθανοτικό διάστημα $[1, 5]$ σε αντίθεση με την εκτέλεση Αφελής Τακτικής.

5.3 Απόδειξη της Ακρίβειας του sPIEC$^{\text{b}}$

Θεώρημα 6. *Όλα τα χρονικά σημεία για τα οποία ο sPIEC$^{\text{b}}$ υποθέτει ότι πραγματοποιείται ένα γεγονός περιλαμβάνονται σε κάποιο μέγιστο πιθανοτικό διάστημα. Με άλλα λόγια, τα αποτελέσματα του sPIEC$^{\text{b}}$ μπορεί να περιέχουν μόνο ψευδή αρνητικά (false negatives) αλλά όχι ψευδή θετικά (false positives).*

Απόδειξη. Έστω το διάστημα $I = [t_i, t_j]$ που εντοπίζεται από τον sPIEC$^{\text{b}}$. Για να αποδείξουμε την παραπάνω δήλωση, αρκεί να αποδείξουμε ότι:

$$\exists I' = [t_{i'}, t_{j'}] :$$

$$t_{i'} \leq t_i, \tag{5.1}$$

$$t_{j'} \geq t_j \text{ και} \tag{5.2}$$

$$\text{το διάστημα } I' \text{ είναι ένα μέγιστο πιθανοτικό διάστημα.} \tag{5.3}$$

Αυτό σημαίνει ότι κάθε χρονικό σημείο στο I ανήκει σε ένα μέγιστο πιθανοτικό διάστημα και επομένως δεν αποτελεί ψευδές θετικό.
Είναι σαφές από τον προηγούμενο ισχυρισμό ότι αρκεί να αποδείξουμε ότι το I είναι ένα πιθανοτικό διάστημα, αφού αυτό θα έδειχνε ότι $\exists I' : I \subseteq I'$, όπου το διάστημα I' είναι ένα μέγιστο πιθανοτικό

διάστημα.
Διακρίνουμε δύο δυνατές περιπτώσεις ανάλογα με το αν τα χρονικά σημεία t_i και t_j ανήκουν στο ίδιο παράθυρο κατά την εκτέλεση του sPIEC$^{\text{b}}$:

1. Τα χρονικά σημεία t_i και t_j βρίσκονται στο ίδιο παράθυρο. Επομένως, το I πρέπει να είχε προστεθεί στην έξοδο από την εκτέλεση του PIEC σε αυτό το παράθυρο. Όπως αποδεικνύεται στο [10], ο αλγόριθμος PIEC εντοπίζει μόνο πιθανοτικά διαστήματα.

2. Τα t_i και t_j ανήκουν σε διαφορετικά παράθυρα. Αυτό υποδεικνύει ότι το I προστέθηκε στην έξοδο του sPIEC$^{\text{b}}$ από τον Αλγόριθμο 4. Εξετάζοντας την απόδειξη του Θεωρήματος 2 για την ανάδειξη του διαστήματος I ως πιθανοτικό, καταλήγουμε στο συμπέρασμα ότι τη απόδειξη αυτή γίνεται ανεξάρτητα από την ορθότητα του *support set*. Ειδικότερα, σε κανένα από τα βήματα της απόδειξης δεν θεωρήθηκε ότι το *support set* περιέχει αυστηρώς όλα τα πιθανά σημεία εκκίνησης της λίστας εισόδου. Η απόδειξη βασίζεται αποκλειστικά στο γεγονός ότι η συνθήκη της εντολής *if* στη γραμμή 7 ελέγχει αν $dp[i] \geq support.set[j].prefix\_prev$. Η σύγκριση αυτή είναι ισοδύναμη με τον έλεγχο της πιθανότητας του αντίστοιχου διαστήματος. Δεδομένου ότι η μεταβλητή $flag$ που χρησιμοποιείται από τον αλγόριθμο 4 υποδεικνύει εάν το διάστημα που ελέγχεται είναι πιθανοτικό, η εντολή *if* στη γραμμή 11 πιστοποιεί ότι ο αλγόριθμος εντοπίζει μόνο πιθανοτικά διαστήματα.

Επομένως, το I είναι ένα πιθανοτικό διάστημα. Έτσι, κάθε χρονική στιγμή στο I είναι επίσης ένα χρονικό σημείο ενός μέγιστου πιθανοτικού διαστήματος. Αυτό αποδεικνύει ότι κανένα χρονικό σημείο στο I δεν είναι ψευδές θετικό. □

Κεφάλαιο 6

Πειραματική Αξιολόγηση

6.1 Το CAVIAR dataset

Για την πειραματική αξιολόγηση του αλγορίθμου και των μεθόδων που παρουσιάζονται σε αυτή την εργασία, χρησιμοποιήσαμε το CAVIAR dataset[1]. Το CAVIAR αποτελείται από δύο σύνολα δεδομένων που περιέχουν στημένα βίντεο παρακολούθησης δημόσιου χώρου που μπορούν να χρησιμοποιηθούν για αναγνώριση ανθρώπινης δραστηριότητας. Για τα πειραματικά αποτελέσματα που παρουσιάζουμε σε αυτή την εργασία, χρησιμοποιήθηκε το πρώτο σύνολο δεδομένων του CAVIAR που περιλαμβάνει 28 βίντεο, με συνολικά 26419 καρέ[2]. Οι ηθοποιοί που συμμετέχουν σε αυτά τα βίντεο εκτελούν δραστηριότητες όπως «βάδιση», «τρέξιμο», «συνάντηση» και «καβγάς». Το dataset σχολιάστηκε από τους δημιουργούς του για να είναι διαθέσιμες οι συντεταγμένες, ο προσανατολισμό και οι ground truth δραστηριότητες κάθε ατόμου ή ομάδας ατόμων που λαμβάνουν χώρα σε κάθε καρέ των βίντεο.
Οι δραστηριότητες που παρουσιάζονται στο CAVIAR έχουν διαχωριστεί σε Βραχυπρόθεσμες (STA) και σε Μακροπρόθεσμες (LTA). Αυτός ο διαχωρισμός αντιστοιχεί στο διαχωρισμό των γεγονότων σε γεγονότα χαμηλού επιπέδου και υψηλού επιπέδου στο Λογαριασμό Γεγονότων, αντίστοιχα. Όπως έχει ήδη αναφερθεί, τα STA είναι απλές ενέργειες που λαμβάνουν χώρα σε ένα μικρό χρονικό διάστημα και περιλαμβάνουν τη «βάδιση», το «τρέξιμο» και την «αδράνεια», μεταξύ άλλων. Τα LTA είναι σύνθετες δραστηριότητες που διαρκούν περισσότερο. Μερικά παραδείγματα LTA είναι οι δραστηριότητες «συμβάδιση», «συνάντηση» και «καβγάς».
Το CAVIAR έχει χρησιμοποιηθεί στο παρελθόν για την αναγνώριση LTA με τον Prob-EC, τα αποτελέσματα του οποίου έχουν συγκριθεί με αιτιοκρατικές εκδόσεις του Λογισμού Γεγονότων[4]. Για το σκοπό αυτό, το CAVIAR χωρίστηκε σε τρεις εκδόσεις που αντιστοιχούν σε τρία διαφορετικά επίπεδα θορύβου. Σε αυτή την εργασία, χρησιμοποιούμε τη διαμέριση για πειραματικές αξιολογήσεις, εκτελώντας κάθε δοκιμή και στις τρεις εκδόσεις θορύβου του CAVIAR.

Αυτές οι εκδοχές παρουσιάζονται παρακάτω:

- Στην πρώτη έκδοση του συνόλου δεδομένων —*ομαλός θόρυβος*— ένα υποσύνολο των STA έχει συνημμένες πιθανότητες, που παράγονται από μια κατανομή Gamma με διαφορετικό

[1] `http://homepages.inf.ed.ac.uk/rbf/CAVIAR/`
[2] Section "Clips From INRIA" of `http://groups.inf.ed.ac.uk/vision/CAVIAR/CAVIARDATA1/`

μέσο. Όλα τα υπόλοιπα STA δεν έχουν τέτοιες πιθανότητες, όπως στην αρχική έκδοση του δατασετ.

- Στη δεύτερη έκδοση—*ενδιάμεσος θόρυβος*— πιθανότητες έχουν επισυναφθεί και στα κατηγορίματα συντεταγμένων και προσανατολισμού χρησιμοποιώντας την ίδια κατανομή Gamma.
- Στην τελευταία έκδοση —*ισχυρός θόρυβος*— προστέθηκαν ψευδή STA που δεν ανήκουν στο αρχικό σύνολο δεδομένων χρησιμοποιώντας μια ομοιόμορφη κατανομή.

Αυτές οι εκδόσεις του CAVIAR, μαζί με τα πρότυπα των LTA στον Λογισμό Γεγονότων, είναι διαθέσιμες στο κοινό[3]. Τροφοδοτούμε αυτά τα δεδομένα στον Prob-EC για να επιτύχουμε αναγνώριση δραστηριότητας βάσει χρονικών στιγμών, υπολογίζοντας μια σειρά αναγνωρίσεων του τύπου $Prob :: \mathsf{holdsAt}(LTA = \mathsf{true}, T)$. Αυτή η σειρά δεδομένων περιλαμβάνει τη ροή στιγμιαίων πιθανοτήτων η οποία χρησιμοποιείται ως είσοδος για τους αλγορίθμους διαστημάτων που παρουσιάστηκαν σε αυτή την εργασία.

6.2 Πειραματικά Αποτελέσματα

Ο αλγόριθμος sPIEC$^{\mathsf{b}}$ εκτελέστηκε στο σύνολο δεδομένων CAVIAR για διαφορετικές τιμές μεγέθους ενεργούς μνήμης, πιθανοτικού κατωφλιού και διαφορετικές στρατηγικές συντήρησης μνήμης. Σε αυτή την ενότητα παρουσιάζονται τα αποτελέσματα αυτών των πειραμάτων σε σύγκριση με τα μέγιστα πιθανοτικά διαστήματα του PIEC. Η τιμή f1-score που εμφανίζεται στα τελικά αποτελέσματα είναι ίση με τη μέση τιμή f1-score των εντοπισμένων διαστημάτων για κάθε γεγονός στο σύνολο δεδομένων.
Χρησιμοποιούμε μέγεθος παραθύρου ίσο με ένα στοιχείο για κάθε πειραματικό αποτέλεσμα που παρουσιάζεται. Επιπλέον, επειδή ο sPIEC$^{\mathsf{b}}$ επεξεργάζεται στιγμιαίες πιθανότητες που παράγονται από ένα Πιθανοτικό σύστημα CER που συλλογίζεται ανά χρονικό σημείο, διαχωρίζουμε τα αποτελέσματά μας ανάλογα με το σύστημα που χρησιμοποιείται για τη δημιουργία της εισόδου του αλγορίθμου. Τέλος, διαχωρίζουμε τα αποτελέσματά μας ανά μακροπρόθεσμη δραστηριότητα.
Μερικά βίντεο του CAVIAR με σχολιασμό που επιδεικνύει τις διαφορές της αναγνώρισης του sPIEC$^{\mathsf{b}}$ με του PIEC μπορούν να βρεθούν εδώ[4].

6.2.1 Σύγκριση των αποτελεσμάτων του αλγορίθμου ροής δεδομένων με αυτά του αρχικού αλγορίθμου

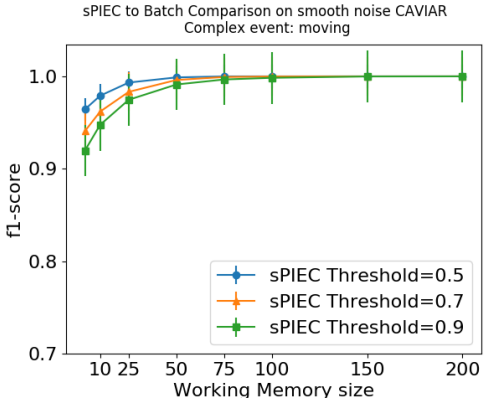
6.2.1.1 Πειράματα με χρήση του πιθανοτικού συστήματος Prob-EC

Ο Prob-EC είναι το πιο διεξοδικά δοκιμασμένο πιθανοτικό σύστημα στιγμιαίας αναγνώρισης στο CAVIAR. Ως εκ τούτου, οι δοκιμές μας βασίζονται στα αποτελέσματα αξιολόγησης του Prob-EC στα δεδομένα του CAVIAR για διάφορα επίπεδα θορύβου. Αυτά τα αποτελέσματα μπορούν να βρεθούν στο [3]
Συγκριτικές μετρικές της απόδοσης των αλγορίθμων sPIEC$^{\mathsf{b}}$ και PIEC με αυτά τα αποτελέσματα ως είσοδο έχουν συναχθεί για κάθε επίπεδο θορύβου. Τα επίπεδα θορύβου που χρησιμοποιούνται

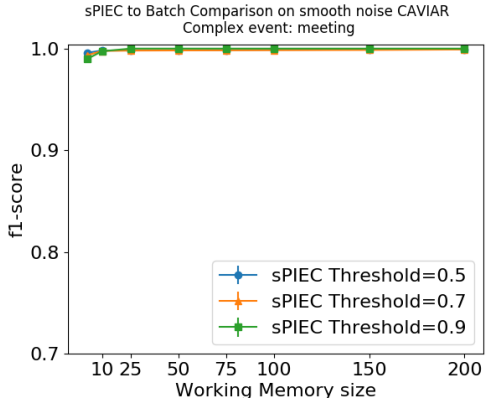
[3] `https://anskarl.github.io/publications/TPLP15/`
[4] `https://gitlab.skel.iit.demokritos.gr/periklismant/caviar_video_annotator`

είναι: καθαρός, ομαλός, ενδιάμεσος και ισχυρός θόρυβος. Λόγω της ομοιότητας μεταξύ των αποτελεσμάτων, παρουσιάζουμε εδώ μόνο τα συγκριτικά διαγράμματα για το CAVIAR ομαλού και ισχυρού θορύβου. Το πλήρες σύνολο διαγραμμάτων μπορεί να βρεθεί στο Παράρτημα Α΄.

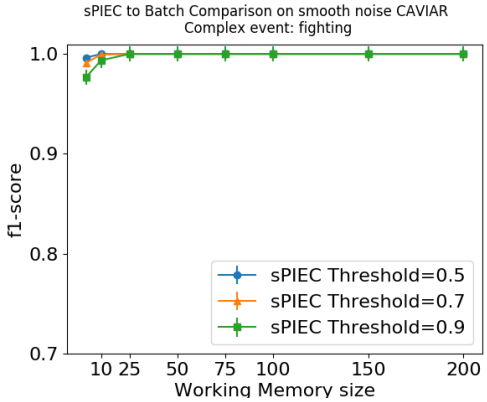
Σχήμα 6.1: Συγκρίσεις με είσοδο τα δεδομένα ομαλού θορύβου



(α΄) Σύγκριση για την LTA: συμβάδιση

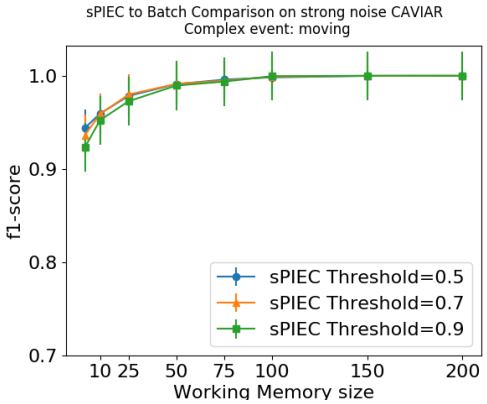


(β΄) Σύγκριση για την LTA: συνάντηση

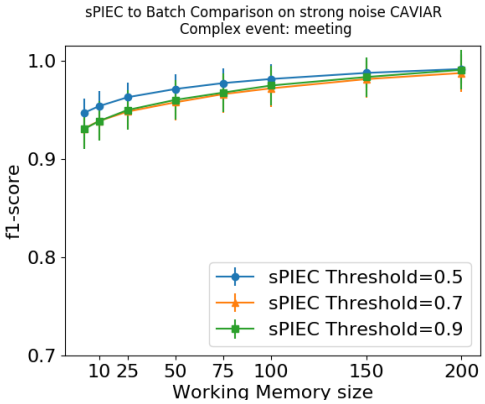


(γ΄) Σύγκριση για την LTA: καβγάς

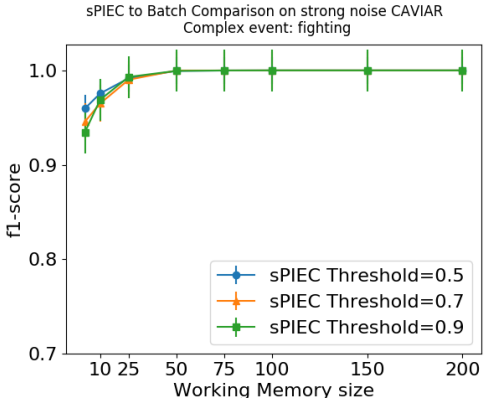
Σχήμα 6.2: Συγκρίσεις με είσοδο τα δεδομένα ισχυρού θορύβου



(α′) Σύγκριση για την LTA: συμβάδιση



(β′) Σύγκριση για την LTA: συνάντηση



(γ′) Σύγκριση για την LTA: καβγάς

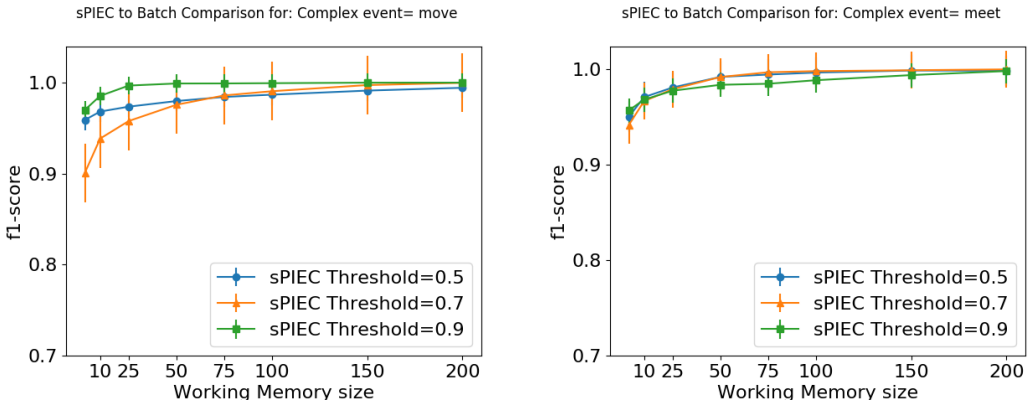
Τα παραπάνω διαγράμματα παρουσιάζουν την ακρίβεια των αποτελεσμάτων του sPIEC^{b} συγκριτικά με τα μέγιστα πιθανοτικά διαστήματα του PIEC για τις τιμές πιθανοτικού κατωφλιού: 0.5, 0.7 και 0.9. Η εκτέλεση του sPIEC^{b} με ενεργή μνήμη επαρκώς μεγάλου μεγέθους πρέπει θεωρητικά να παράγει ακριβώς τα ίδια αποτελέσματα με την αντίστοιχη εκτέλεση του PIEC. Επομένως, είναι λογικό να παρατηρούμε πως η μετρική f1-score αυξάνεται καθώς αυξάνουμε το μέγεθος της ενεργής μνήμης. Τα διαγράμματα 6.1 και 6.2 δείχνουν ότι ο sPIEC^{b} προσεγγίζει το μέγιστα πιθανοτικά διάστηματα ενός σύνθετου γεγονότος πιο άνετα για τα δεδομένα ομαλού θορύβου. Η μόνη εξαίρεση είναι η LTA συμβάδισης για την οποία αυτή η συμπεριφορά δεν είναι τόσο εμφανής.

6.2.1.2 Πειράματα με είσοδο τα αποτελέσματα της πιθανοτικής μεθόδου OSLα

Η μέθοδος OSLα χρησιμοποιεί μια αξιωματοποίηση που συνδυάζει τα Μαρκοβιανά Λογικά Δίκτυα (Markov Logic Networks – MLN) [41] με τον Λογισμό Γεγονότων για την εξαγωγή πιθανοτικών προτύπων σύνθετων γεγονότων [63]. Αυτά τα πρότυπα χρησιμοποιούνται στη συνέχεια για την

πιθανοτική αναγνώριση σύνθετων γεγονότων ανά χρονική στιγμή. Χρησιμοποιούμε αυτές τις αναγνωρίσεις ως είσοδο για τους αλγορίθμους sPIEC^b και PIEC στις δοκιμές μας και παρουσιάζουμε τα αποτελέσματα παρακάτω.

Σχήμα 6.3: Συγκρίσεις με είσοδο τα δεδομένα της μεθόδου OSLα



(α′) Σύγκριση για την LTA: συμβάδιση

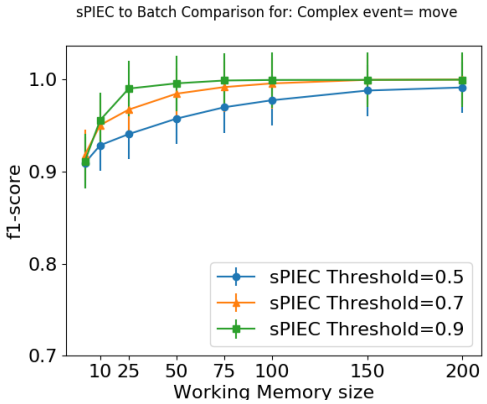
(β′) Σύγκριση για την LTA: συνάντηση

Τα διαγράμματα αυτά (Σχήμα 6.3) παρουσιάζουν σχεδόν τέλεια μετρική f1-score για όλα τα μεγέθη μνήμης. Η μόνο εξαίρεση είναι η εκτέλεση για τη δραστηριότητα συμβάδισης με πιθανοτικό κατώφλι ίσο με 70% (Σχήμα 6.3α′). Εδώ, η βελτίωση της μετρικής f1-score επιτυγχάνεται με μία μικρή αύξηση του μεγέθους της ενεργούς μνήμης.

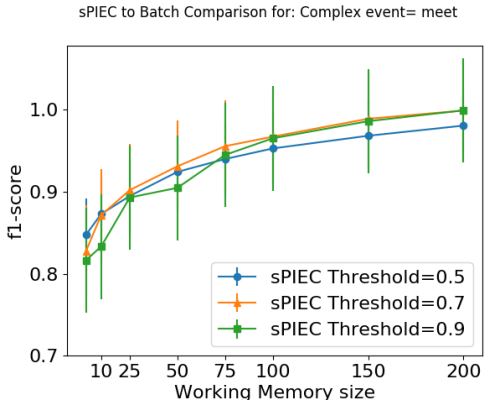
6.2.1.3 Πειράματα με είσοδο τα αποτελέσματα της πιθανοτικής μεθόδου Diagonal Newton

Η μέθοδος Diagonal Newton (DN) βασίζεται σε έναν αλγόριθμο δεύτερης τάξης που περιγράφεται στην εργασία [64]. Το σύστημα που χρησιμοποιήσαμε για τα ακόλουθα πειράματα χρησιμοποιεί τον MLN-EC σε συνδυασμό με τη μέθοδο DN για επιβλεπόμενη εκμάθηση βάρων για τα πρότυπα των LTA [44].

Σχήμα 6.4: Συγκρίσεις με είσοδο τα δεδομένα της μεθόδου DN



(α′) Σύγκριση για την LTA: συμβάδιση



(β′) Σύγκριση για την LTA: συνάντηση

Τα αποτελέσματα που παρουσιάζονται στα παραπάνω διαγράμματα (Σχήμα 6.4) διαφέρουν αρκέτα μεταξύ των LTA που εξετάστηκαν.

- Τα αποτελέσματα για το LTA συμβάδισης (Σχήμα 6.4α′) έχουν f1-score περίπου ίσο με 92% για μικρά μεγέθη μνήμης και βελτιώνονται σταδιακά καθώς αυξάνεται η μνήμη. Ο ρυθμός αυτής της αύξησης είναι μεγαλύτερος για εκτελέσεις με υψηλές τιμές πιθανοτικού κατωφλίου.
- Τα αποτελέσματα για το LTA συνάντησης (Σχήμα 6.4β′) είναι λιγότερο ακριβή από τα αποτελέσματα της συμβάδισης για όλα τα μεγέθη μνήμης, αντίστοιχα. Αυτή είναι η μόνη περίπτωση κατά την οποία το f1-score πέφτει στο 80%. Θεωρούμε ότι αυτό οφείλεται στις διακυμάνσεις στις πιθανότητες των αναγνωρίσεων του DN για το LTA συνάντησης.

6.2.1.4 Γενικά Συμπεράσματα

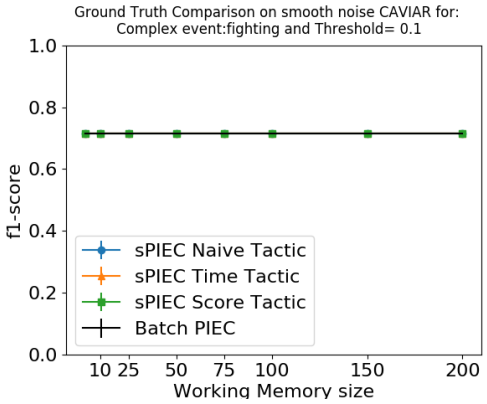
Εξετάζοντας τα παραπάνω διαγράμματα, συμπεραίνουμε ότι χρησιμοποιώντας μια ενεργή μνήμη εργασίας των περίπου 50 στοιχείων, ο αλγόριθμος sPIEC$^{\text{b}}$ είναι σε θέση να ανιχνεύσει κάθε μέγιστο πιθανοτικό διάστημα ενός συγκεκριμένου σύνθετου γεγονότος με σχεδόν τέλεια ακρίβεια και ανάκληση. Επειδή το μέσο βίντεο του CAVIAR περιέχει περίπου 1000 καρέ, αυτό αντιστοιχεί στο 5% των επεξεργασμένων χρονικών σημείων κατά μέσο όρο.
Ωστόσο, ο περιορισμός της μνήμης εργασίας του sPIEC$^{\text{b}}$ περαιτέρω (ακόμη και σε μέγεθος 2 στοιχείων), έχει ως αποτέλεσμα μια τιμή f1-score πάνω από 90 %, κατά μέσο όρο. Επομένως, η χρησιμοποίηση του sPIEC$^{\text{b}}$ σε εφαρμογές που απαιτούν επεξεργασία ενός μεγάλου συνόλου δεδομένων μπορεί να είναι συνετή, αφού ακόμη και οι εκτελέσεις με μικρή ενεργή μνήμη πετυχαίνουν μεγάλη ακρίβεια.

6.2.2 Σύγκριση των αποτελεσμάτων του αλγορίθμου ροής δεδομένων με το ground truth του CAVIAR

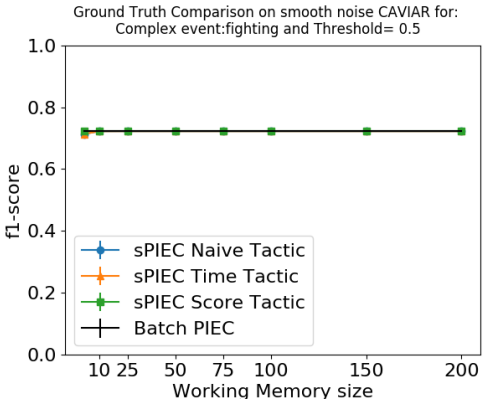
Σε αυτή την ενότητα παρουσιάζουμε μια σύγκριση της κάθε αναγνώρισης γεγονότος από τον αλγόριθμο sPIEC$^{\mathsf{b}}$ στο CAVIAR με τις πραγματικές εμφάνισεις αυτού του γεγονότος, όπως αυτές σημειώθηκαν από τους δημιουργούς του dataset. Τα αποτελέσματα αυτών των συγκρίσεων περιλαμβάνουν τα συνολικά αληθή θετικά, ψευδή αρνητικά και ψευδή θετικά κάθε αναγνώρισης που υπολογίζονται ανά χρονικό σημείο και χρησιμοποιούνται για την μέτρηση του f1-score των εκτελέσεων του sPIEC$^{\mathsf{b}}$ για διάφορα μεγέθη ενεργούς μνήμης.
Παρουσιάζουμε τα αποτελέσματά μας ξεχωριστά για κάθε πιθανοτικό σύστημα εισόδου, σύνθετο γεγονός και πιθανοτικό κατώφλι. Για τα πειράματα με είσοδο τις στιγμιαίες πιθανότητες των συστημάτων OSLα και DN, χρησιμοποιήσαμε τόσο τους σχολιασμούς των δημιουργών του CAVIAR όσο και τις στιγμιαίες πιθανότητες που συνάγονται από κάθε σύστημα ως ground truth.
Λόγω της μεγάλου πλήθους συνδυασμών παραμέτρων και συστημάτων, αποφασίσαμε να απομονώσουμε μόνο τα πιο ενδιαφέροντα διαγράμματα σε αυτή την ενότητα. Το πλήρες σύνολο αυτών των συγκριτικών διαγραμμάτων υπάρχει στο Παράρτημα Α′.

6.2.2.1 Πειράματα με χρήση του πιθανοτικού συστήματος Prob-EC

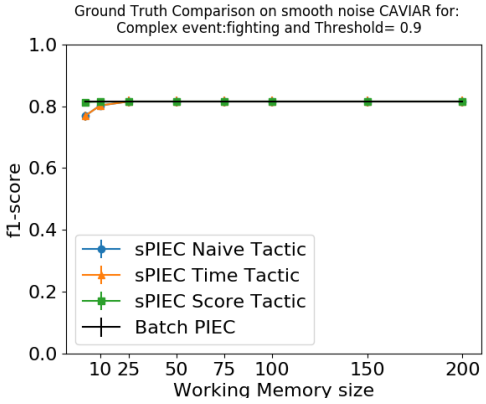
Σχήμα 6.5: Συγκρίσεις με είσοδο τα δεδομένα ομαλού θορύβου για την LTA: καβγάς



(α′) Πιθανοτικό Κατώφλι: 0.1

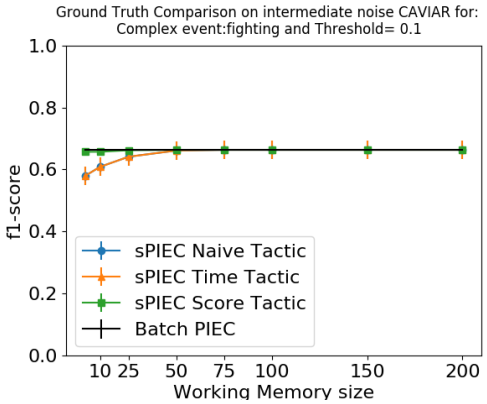


(β′) Πιθανοτικό Κατώφλι: 0.5

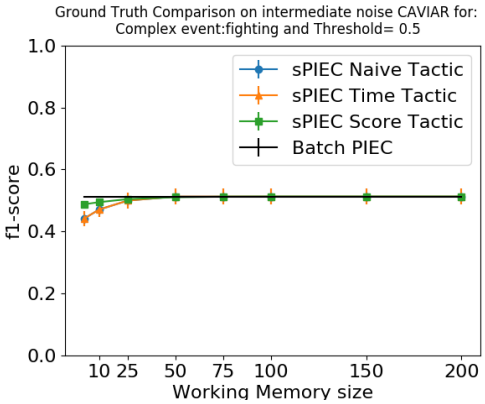


(γ′) Πιθανοτικό Κατώφλι: 0.9

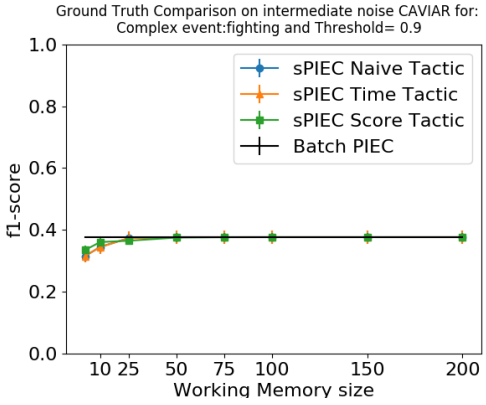
Σχήμα 6.6: Συγκρίσεις με είσοδο τα δεδομένα ενδιάμεσου θορύβου για την LTA: καβγάς



(α′) Πιθανοτικό Κατώφλι: 0.1

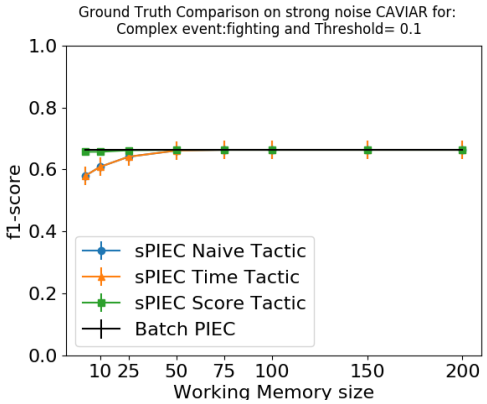


(β′) Πιθανοτικό Κατώφλι: 0.5

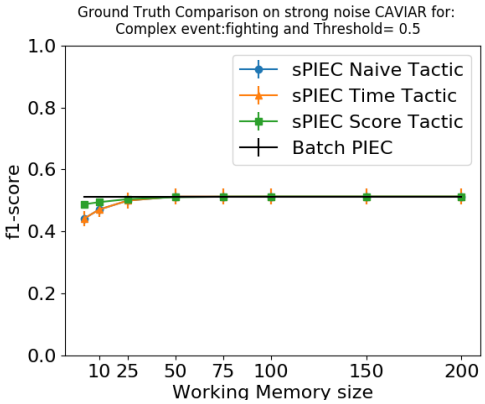


(γ′) Πιθανοτικό Κατώφλι: 0.9

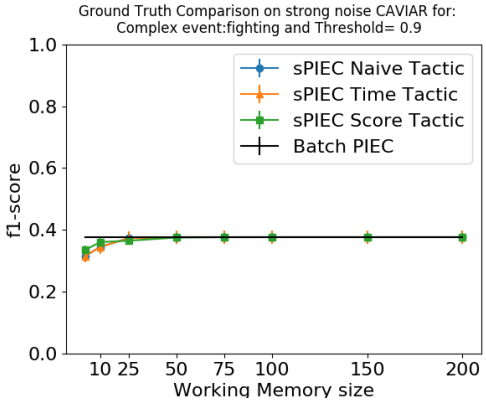
Σχήμα 6.7: Συγκρίσεις με είσοδο τα δεδομένα ισχυρού θορύβου για την LTA: καβγάς



(α′) Πιθανοτικό Κατώφλι: 0.1



(β′) Πιθανοτικό Κατώφλι: 0.5

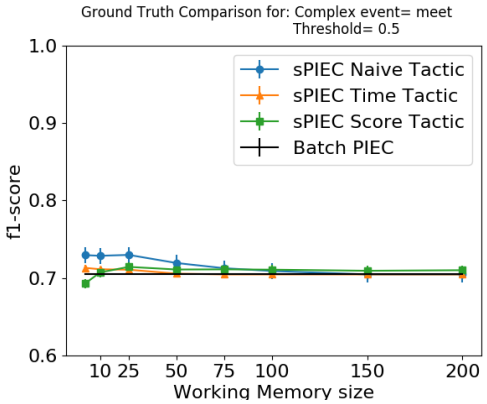


(γ′) Πιθανοτικό Κατώφλι: 0.9

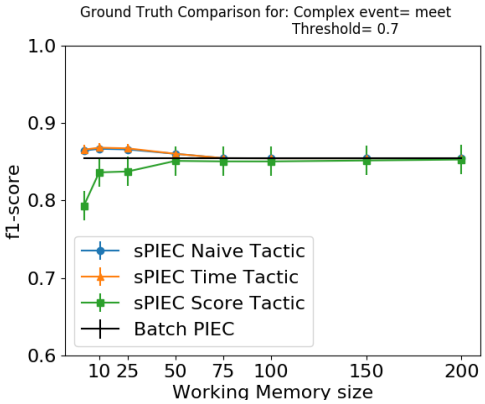
Παρατηρούμε διαφορές ως προς το πώς η επιλογή για την κατωφλίου επηρεάζει την απόδοση του αλγορίθμου ανάλογα με το επίπεδο θορύβου. Για παράδειγμα, μια αύξηση στην τιμή κατωφλίου προκαλεί μείωση της τιμής f1-score για τις πιθανότητες εισόδου ενδιάμεσου και ισχυρού θορύβου (Σχήματα 6.6 και 6.7). Αντιθέτως, ο sPIEC$^{\text{b}}$ αποδίδει καλύτερα για μια τιμή κατωφλίου ίση με 90% σε ένα περιβάλλον ομαλού θορύβου (Σχήμα 6.5).

6.2.2.2 Πειράματα με είσοδο τα αποτελέσματα της πιθανοτικής μεθόδου OSLα

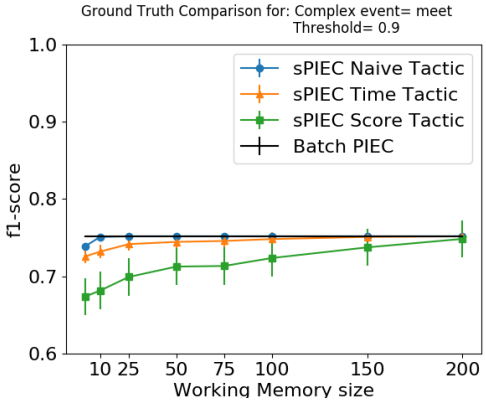
Σχήμα 6.8: Συγκρίσεις με είσοδο τα δεδομένα της μεθόδου OSLα με το ground truth για την LTA: συνάντηση



(α′) Πιθανοτικό Κατώφλι: 0.5



(β′) Πιθανοτικό Κατώφλι: 0.7



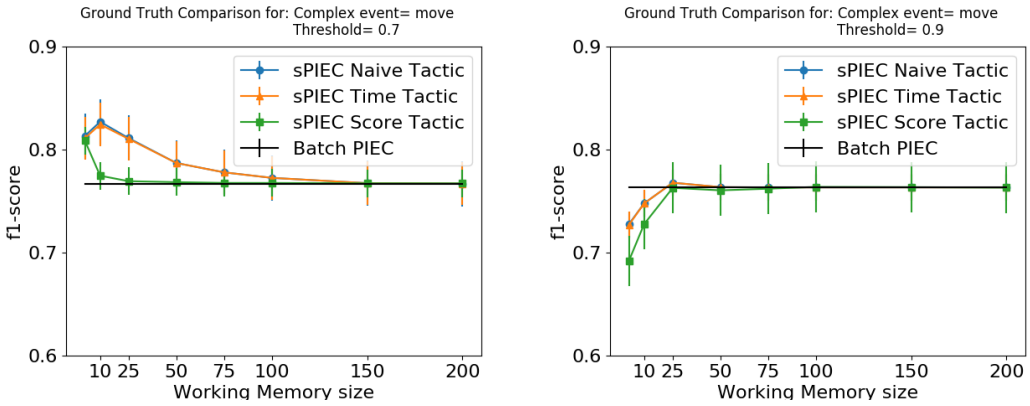
(γ′) Πιθανοτικό Κατώφλι: 0.9

Τα μέγιστα πιθανοτικά διαστήματα που ανιχνεύονται από τον sPIEC$^{\text{b}}$ έχουν μικρότερο μέσο μέγεθος καθώς η τιμή του πιθανοτικού κατωφλιού αυξάνεται. Επομένως, τα διαστήματα που βρέθηκαν από τον sPIEC$^{\text{b}}$ με $t = 0.5$ περιέχουν τα περισσότερα χρονικά σημεία μεταξύ των παραπάνω εκτελέσεων. Η ένταξη πάρα πολλών χρονικών σημείων στην έξοδο αυξάνει στατιστικά το ποσό των ψευδώς θετικών και βλάπτει τη μετρική ακρίβειας. Αυτό το γεγονός μπορεί να εξηγήσει τη διαφορά f1-score μεταξύ των διαγραμμάτων 6.8α′ και 6.8β′.
Τα διαστήματα που ανιχνεύονται από τον sPIEC$^{\text{b}}$ με $t = 0.9$ περιέχουν τα λιγότερα χρονικά σημεία, καθιστώντας τον αλγόριθμο περισσότερο επιλεκτικό. Κατά συνέπεια, αυτή η εκτέλεση έχει τα περισσότερα ψευδή αρνητικά και μια μικρότερη μετρική ανάκλησης. Αυτό το γεγονός εξηγεί τη διαφορά f1-score μεταξύ των διαγραμμάτων (6.8γ′) και (6.8β′). Έτσι, γι′ αυτό το συγκεκριμένο

σύνολο δεδομένων και σύστημα εισόδου, η τιμή κατωφλίου 0.7 για τον sPIEC^{b} παράγει τα πιο ακριβή αποτελέσματα.

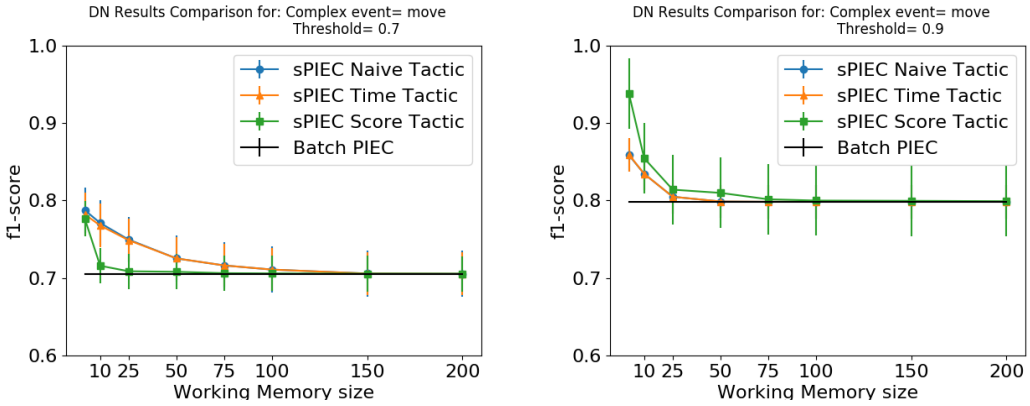
6.2.2.3 Πειράματα με είσοδο τα αποτελέσματα της πιθανοτικής μεθόδου DN

Σχήμα 6.9: Συγκρίσεις με είσοδο τα δεδομένα της μεθόδου DN με το ground truth για την LTA: συμβάδιση



(α′) Πιθανοτικό Κατώφλι: 0.7 (β′) Πιθανοτικό Κατώφλι: 0.9

Σχήμα 6.10: Συγκρίσεις με είσοδο τα δεδομένα της μεθόδου DN με τις στιγμιαίες αναγνωρίσεις της ίδια μεθόδου για την LTA: συμβάδιση



(α′) Πιθανοτικό Κατώφλι: 0.7 (β′) Πιθανοτικό Κατώφλι: 0.9

Σε σύγκριση με το ground truth του dataset (Σχήμα 6.9), το sPIEC^{b} επιτυγχάνει μία τιμή f1-score μεγαλύτερη από 75% για τις εν λόγω παραμέτρους. Σύμφωνα με το σχήμα 6.9α′, η Τακτική Σκορ

συγκλίνει στα αποτελέσματα του PIEC σημαντικά γρηγορότερα από τις υπόλοιπες.
Σε σύγκριση με τις στιγμιαίες πιθανότητες που προκύπτουν από τη μέθοδο Diagonal Newton (Σχήμα 6.10), ο αλγόριθμος sPIEC$^{\text{b}}$ αποδίδει καλύτερα για υψηλές τιμές πιθανοτικού κατωφλιού ($t = 0.9$).

6.2.2.4 Γενικά Συμπεράσματα

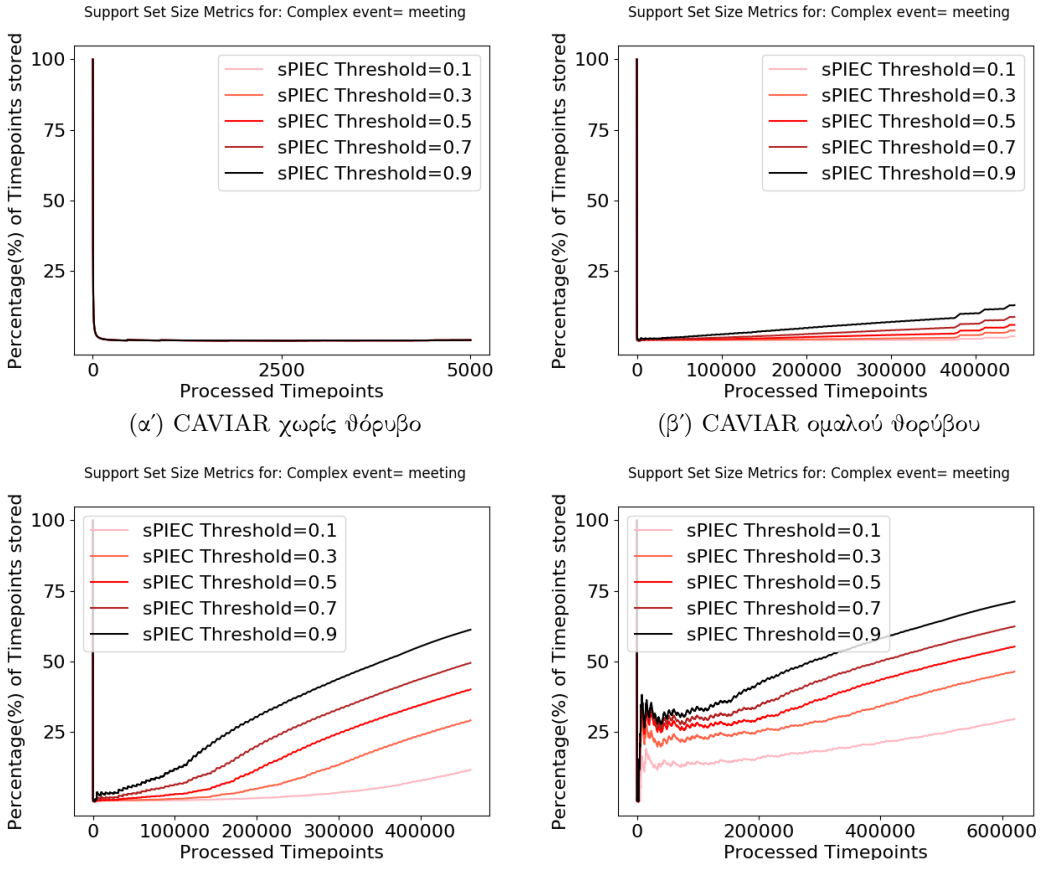
Λόγω της σύντομης διάρκειας των σύνθετων γεγονότων σε αυτό το dataset, παρατηρούμε μόνο μικρές μετατοπίσεις στο f1-score καθώς το μέγεθος της μνήμης εργασίας του sPIEC$^{\text{b}}$ αυξάνεται. Ειδικότερα, για ένα μέγεθος μνήμης εργασίας μεγαλύτερο από 100 στοιχεία, ο αλγόριθμος sPIEC$^{\text{b}}$ παράγει τα ίδια αποτελέσματα με το PIEC. Επιπλέον, υπάρχει πολύ μικρή απόκλιση από το PIEC ακόμα και για εκτελέσεις του sPIEC$^{\text{b}}$ με μνήμη εργασίας που χωράει δύο στοιχεία, ενθαρρύνοντας την χρήση του sPIEC$^{\text{b}}$ σε ένα περιβάλλον ροής δεδομένων.
Παρατηρούμε ότι σε ορισμένες περιπτώσεις (Σχήμα 6.8α′ και 6.8β′) ο sPIEC$^{\text{b}}$ αποδίδει καλύτερα από τον PIEC. Αυτή η παρατήρηση μπορεί να δικαιολογηθεί από το γεγονός ότι η υπερχείλιση μνήμης στον sPIEC$^{\text{b}}$ διαγράφει χρονικά σημεία που θα αποτελούσαν ψευδή θετικά μέγιστων πιθανοτικών διαστημάτων, αυξάνοντας την ακρίβεια της εκτέλεσης. Ωστόσο, υπάρχουν περιπτώσεις στις οποίες η υπερχείλιση μνήμης διαγράφει αληθή θετικά, μειώνοντας τη μέτρική ανάκλησης της εκτέλεσης (Σχήμα 6.8γ′).
Βλέπουμε, λοιπόν, πως κάποιες φορές η διαγραφή χρονικών σημείων από μέγιστα πιθανοτικά διαστήματα οδηγεί σε πιο ακριβής αναγνώριση. Η παραπάνω παρατήρηση μας ωθεί στην αναθεώρηση του ορισμού του μέγιστου πιθανοτικού διαστήματος (Ορισμός 2) ώστε αυτά να μην διατηρούν πλεονάζοντα χρονικά σημεία.

6.2.3 Μετρικές σχετικά με το μέγεθος του support set

Σε αυτή την ενότητα θα παρουσιάσουμε ορισμένα αποτελέσματα της εκτέλεσης του sPIEC με απεριόριστη μνήμη στα δεδομένα του CAVIAR. Πιο συγκεκριμένα, επισυνάψαμε όλες τις αναγνωρίσεις σύνθετων γεγονότων του εκάστατε συστήματος στιγμιαίας αναγνώρισης σε μία ροή δεδομένων και μελετήσαμε τη χρησιμοποιήση του *support set* από τον αλγόριθμο μας κατά την εξέλιξη της ροής αυτής. Στα παρακάτω διαγράμματα φαίνεται το ποσοστιαίο μέγεθος του *support set* σε σύγκριση με το πλήθος επεξεργασμένων χρονικών σημείων της ροής κάθε στιγμή. Οι εκτελέσεις γίνονται για διάφορες τιμές του πιθανοτικού κατωφλιού ώστε να εξεταστεί η εξάρτηση του μεγέθους του *support set* από αυτό.

6.2.3.1 Πειράματα με χρήση του πιθανοτικού συστήματος Prob-EC

Σχήμα 6.11: Μέγεθος *support set* κατά την αναγνώριση των δραστηριοτήτων συνάντησης



(α′) CAVIAR χωρίς θόρυβο

(β′) CAVIAR ομαλού θορύβου

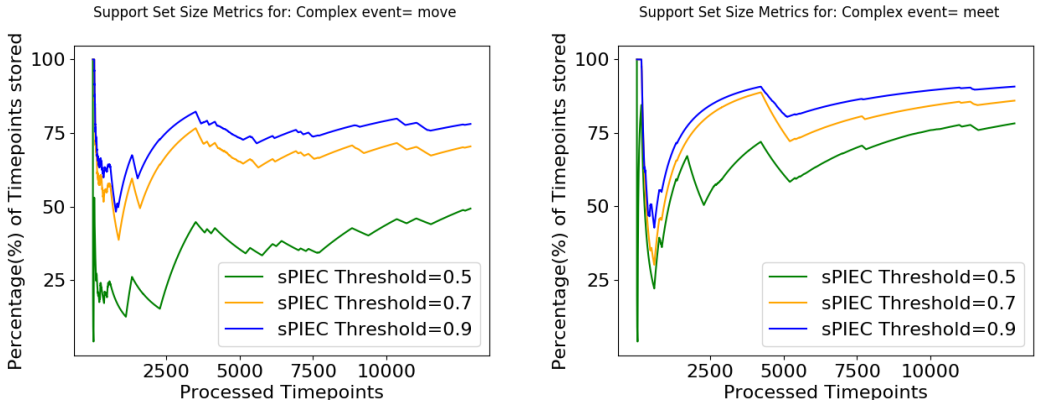
(γ′) CAVIAR ενδιάμεσου θορύβου

(δ′) CAVIAR ισχυρού θορύβου

Το Σχήμα 6.11 παρουσιάζει το πλήθος των χρονικών σημείων που αποθηκεύονται στο *support set* κατά την επεξεργασία των αναγνωρίσεων της δραστηριότητας συνάντησης που συνήγαγε το Prob-EC σε περιβάλλοντα διαφόρων επιπέδων θορύβου. Σε αυτό φαίνεται πως υπάρχει μία σαφής συσχέτιση μεταξύ του επιπέδου θορύβου και του μεγέθους του προκύπτοντος *support set*. Παρατηρούμε ότι σε σκηνικά με ασήμαντο θόρυβο, (Σχήματα 6.11α′ και 6.11β′), ο sPIEC^{b} απαιτεί μια μικρή ποσότητα μνήμης εργασίας για να συμπεράνει τα σωστά μέγιστα πιθανοτικά διαστήματα της ροής εισόδου. Ωστόσο, τα πιο θορυβώδη περιβάλλοντα (Σχήματα 6.11γ′ και 6.11δ′) απαιτούν μια μνήμη εργασίας συγκρίσιμη με το μέγεθος της ροής. Επομένως, το μέγεθος της μνήμης εργασίας θα πρέπει να περιορίζεται σε τέτοιες περιπτώσεις, όπως περιγράφηκε στο Κεφάλαιο 5.

6.2.3.2 Πειράματα με είσοδο τα αποτελέσματα της πιθανοτικής μεθόδου OSLα

Σχήμα 6.12: Μέγεθος *support set* κατά την αναγνώριση των δραστηριοτήτων συμβαδίσης και συνάντησης

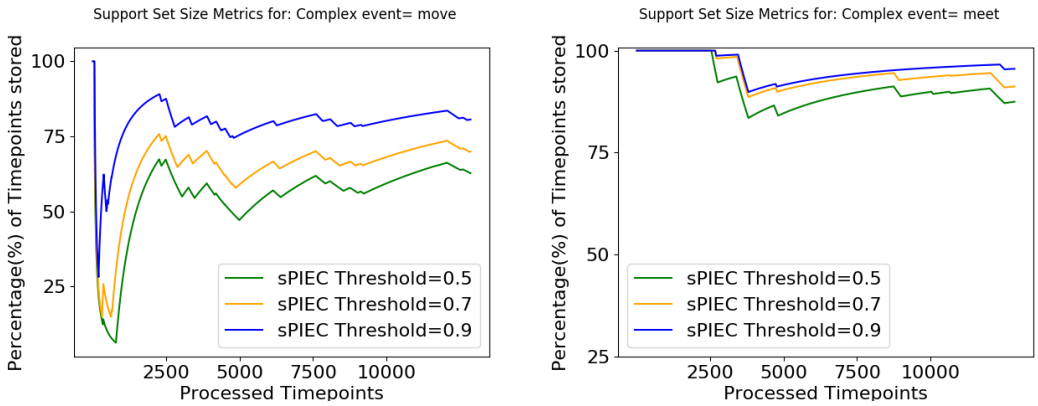


(α΄) Συμβάδιση (β΄) Συνάντηση

Στο Σχήμα 6.12 φαίνεται η πορεία του μεγέθους του *support set* καθώς ο αλγόριθμος επεξεργάζεται μία ροή από στιγμιαίες πιθανότητες γεγονότων που έχουν παραχθεί με τη μέθοδο $OSL\alpha$. Σε αυτά τα διαγράμματα είναι αρκετά αισθητή η επίδραση της τιμή πιθανοτικού κατωφλιού στο μέγεθος του παραγόμενου *support set*. Πιο συγκεκριμένα, στο διάγραμμα 6.12α΄ παρατηρούμε την μεγάλη απόκλιση στο μέγεθος της δομής μεταξύ των τιμών κατωφλιού 0.5 και 0.7. Επιπλέον, παρατηρούμε ότι η αναγνώριση του LTA συνάντησης είναι λίγο πιο απαιτητική σε μνήμη και ότι σε κάποιες περιπτώσεις το μέγεθος του *support set* καταλήγει στο 85% των συνολικών δεδομένων.

6.2.3.3 Πειράματα με είσοδο τα αποτελέσματα της πιθανοτικής μεθόδου DN

Σχήμα 6.13: Μέγεθος *support set* κατά την αναγνώριση των δραστηριοτήτων συμβάδισης και συνάντησης



(α′) Συμβάδιση (β′) Συνάντηση

Παραπάνω (Σχήμα 6.13) βλέπουμε ανάλογα διαγράμματα για την επίδειξη του χρησιμοποιούμενου μεγέθους *support set* κατά την εκτέλεση του sPIEC^{b} με στιγμιαίες πιθανότητες που προκύπτουν με την μέθοδο DN. Ιδιαίτερο ενδιαφέρον παρουσιάζει το διάγραμμα 6.13β′ όπου το *support set* συντηρεί πρακτικά όλο το dataset. Αυτή η περίπτωση αναδεικνύει την ανάγκη χρησιμοποίησης του αλγορίθμου sPIEC^{b} για να τεθεί ρητά ένα όριο μνήμης.

6.2.3.4 Γενικά Συμπεράσματα

Είναι σαφές από όλα τα παραπάνω διαγράμματα ότι η εκτέλεση του αλγορίθμου sPIEC με απεριόριστη μνήμη σε ένα περιβάλλον ροής δεδομένων δεν είναι ρεαλιστική επιλογή για το συγκεκριμένο dataset. Ο αλγόριθμος χρησιμοποιεί ένα *support set* με μέγεθος ίσο με περίπου το 75% του αριθμού των επεξεργασμένων χρονικών σημείων για το μεγαλύτερο μέρος του χρόνου εκτέλεσης. Πιο συγκεκριμένα, καταλήγουμε στα εξής:

- Το *support set* αυξάνεται κάθε φορά που υπάρχει μια νέα τρέχουσα ελάχιστη τιμή στη λίστα prefix. Επομένως, μια είσοδος με πολλές χαμηλές στιγμιαίες πιθανότητες (κοντά στο 0) απαιτεί ένα μεγάλο *support set*, καθώς οι τιμές στη λίστα prefix θα είναι, κατά κύριο λόγο, φθίνουσες. Επομένως, υποψιάζομαστε ότι παρατηρούμε πτώση του μεγέθους του *support set* στα διαγράμματα όταν υπάρχουν συνεχόμενες υψηλές στιγμιαίες πιθανότητες (πραγματοποίηση γεγονότος) και αύξηση όταν ο sPIEC λαμβάνει συνεχόμενες χαμηλές στιγμιαίες πιθανότητες (κενά ή άσχετα καρέ βίντεο). Δεδομένου ότι στα περισσότερα βίντεο το πλήθος των καρέ όπου δεν πραγματοποιείται κάποιο συμβάν υπερτερεί του πλήθους των εμφανίσεών του, αυτό το αποτέλεσμα δεν πρέπει να μας εκπλήσσει.
- Ο sPIEC απαιτεί λιγότερη ενεργή μνήμη για την αναγνώριση των LTA συμβάδισης σε σχέση με την αναγνώριση των LTA συνάντησης (Σχήματα 6.12 και 6.13).

- Το *support set* είναι ανάλογο της τιμής του πιθανοτικού κατωφλιού. Αυτό το πειραματικό συμπέρασμα οφείλεται στον τρόπο κατασκευής της λίστας prefix. Όσο πιο μεγάλο είναι το κατώφλι, τόσο πιο πολλές μειώσεις θα υφίστανται οι τιμες της λίστας prefix. Έτσι, τα τρέχοντα ελάχιστα της λίστας θα είναι περισσότερα και, κατά συνέπεια, το *support set* θα είναι μεγαλύτερο.

Κεφάλαιο 7

Σύνοψη και Μελλοντικές Επεκτάσεις

7.1 Σύνοψη

Ο στόχος αυτής της εργασίας ήταν η επέκταση του ήδη αναπτυγμένου αλγόριθμου PIEC για να λειτουργεί σε περιβάλλον ροής δεδομένων. Για το σκοπό αυτό, παρουσιάσαμε τον αλγόριθμο sPIEC που κληρονομεί τον κεντρικό μηχανισμό του PIEC για την πραγματοποίηση αναγνώρισης σε χρονικά διαστήματα με βάση τον ορισμό των μέγιστων πιθανοτικών διαστημάτων. Όμως, ο sPIEC χρησιμοποιεί χρονικά παράθυρα για την επεξεργασία μιας ροής στιγμιαίων πιθανοτικών αναγνωρίσεων LTA και μία ενεργή μνήμη για την αποθήκευση των πιθανών σημείων εκκίνησης των μέγιστων πιθανοτικών διαστημάτων. Αναλύσαμε τις λειτουργίες και τις δομές δεδομένων του sPIEC παρέχοντας ταυτόχρονα αποδείξεις για την πολυπλοκότητα και την ορθότητα τους (Κεφάλαιο 4). Στη συνέχεια, σχολιάσαμε την απόδοση του sPIEC σε ένα περιβάλλον περιορισμένης ενεργής μνήμης (sPIEC$^{\mathsf{b}}$) και προτείναμε τέσσερις στρατηγικές για την συντήρηση μνήμης με τέτοιο τρόπο, ώστε να μεγιστοποιήσουμε την ακρίβεια και την ανάκληση του αλγορίθμου (Κεφάλαιο 5). Τέλος, εκτελέσαμε τον αλγόριθμό μας σε ένα dataset αναγνώρισης ανθρώπινης δραστηριότητας και συγκρίναμε τα αποτελέσματά του τόσο με αυτά του αρχικού αλγόριθμου PIEC όσο και με το ground truth για κάθε γεγονός στο dataset. Παράλληλα, προσπαθήσαμε να εντοπίσουμε ένα επαρκές μέγεθος ενεργής μνήμης για τον sPIEC$^{\mathsf{b}}$ (Κεφάλαιο 6).

Η θεωρητική ανάλυση του αλγορίθμου αποδεικνύει τη δυνατότητα επέκτασης της λογικής του PIEC σε ένα περιβάλλον ροής δεδομένων με τη χρήση χρονικών παραθύρων και τη διατήρηση μόνο πληροφοριών που αφορούν έναν περιορισμένο αριθμό επεξεργασμένων χρονικών σημείων. Αυτά τα χρονικά σημεία ικανοποιούν την προϋπόθεση 4.11 και είναι τα μόνα πιθανά σημεία εκκίνησης των μέγιστων πιθανοτικών διαστημάτων που μπορεί να καταλήγουν σε μελλοντικά παράθυρα (Θεώρημα 5). Ενισχύουμε τον PIEC με μια συνάρτηση για την εύρεση μέγιστων πιθανοτικών διαστημάτων με βάση αυτά τα χρονικά σημεία και μια συνάρτηση για την ανίχνευση αυτών των χρονικών σημείων (Αλγόριθμοι 4 και 5).

Η εμπειρική αξιολόγηση του αλγορίθμου παρουσιάζει την απόδοση του sPIEC$^{\mathsf{b}}$ για διάφορα μεγέθη ενεργής μνήμης, στρατηγικές συντήρησης μνήμης, τιμές πιθανοτικού κατωφλιού και συστήματα εισόδου. Εξετάζοντας αυτά τα αποτελέσματα, καταλήγουμε στο συμπέρασμα ότι ο sPIEC$^{\mathsf{b}}$ είναι

επιτυχής στην ανίχνευση μέγιστων πιθανοτικών διαστήματα με μεγάλη ακρίβεια. Επιπλέον, καθορίζουμε τις διαφορές μεταξύ των εκτελέσεων sPIEC^{b} με διάφορες τιμές παραμέτρων και συζητάμε τις τιμές f1-score που προκύπτουν. Για το σκοπό αυτό, παρουσιάσαμε διαγράμματα που συγκρίνουν τα αποτελέσματα του sPIEC^{b} τόσο με τα αποτελέσματα του PIEC όσο και με το ground truth.

7.2 Μελλοντικές Επεκτάσεις

Εξετάζοντας τις ιδιότητες του αλγορίθμου και τις πειραματικές επιδόσεις του, έχουμε σκεφτεί ορισμένες μελλοντικές βελτιώσεις για τον sPIEC και μερικές επιπλέον εφαρμογές του. Πιο συγκεκριμένα:

- Τα συγκριτικά διαγράμματα σχετικά με την απόδοση των sPIEC^{b} και PIEC ως προς το ground truth (Ενότητα 6.2.2) δείχνουν ότι σε ορισμένες περιπτώσεις η αναγνώρισή μας είναι κακή. Αυτό το σφάλμα μπορεί να προέρχεται είτε από τις στιγμιαίες πιθανότητες του συστήματος που χρησιμοποιείται ως είσοδος είτε από τη διαδικασία συμπερασμού διαστημάτων από αυτές τις πιθανότητες. Για να αντιμετωπίσουμε το τελευταίο είδος ανακριβειών, σκοπεύουμε να αναθεωρήσουμε τον ορισμό του μέγιστου πιθανοτικού διαστήματος (Ορισμός 2) και να δοκιμάσουμε εναλλακτικές προσεγγίσεις. Πιο συγκεκριμένα, θα θέλαμε να περιορίσουμε το πλήθος χρονικών σημείων ενός χρονικού διαστήματος που έχουν πολύ χαμηλές πιθανότητες με την περικοπή ενός μέρους του προθέματος ή του επιθέματος του. Μία εναλλακτική στρατηγική θα ήταν η εξαγωγή του ορισμού αυτού χρησιμοποιώντας τεχνικές μηχανικής μάθησης.

- Η διαδικασία διαχωρισμού διαστήματος που εκτελείται μετά την εκτέλεση κάθε παράθυρου (Αλγόριθμος 2, γραμμή 5) είναι αρκετά δαπανηρή για εκτελέσεις που έχουν ως είσοδο ροές δεδομένων με πολλά μέγιστα πιθανοτικά διαστήματα. Θα θέλαμε να χρησιμοποιήσουμε μια συνάρτηση που ανιχνεύει τα προς επέκταση χρονικά διαστήματα χωρίς την εκτέλεση μιας διαδικασίας με τετραγωνική πολυπλοκότητα

- Η πειραματική αξιολόγηση του αλγορίθμου στερείται ποικιλίας όσον αφορά τα χρησιμοποιούμενα σύνολα δεδομένων και τα σενάρια πραγματοποίησης σύνθετων γεγονότων. Το CAVIAR dataset περιέχει λίγα σύνθετα γεγονότα ανά βίντεο μεταξύ ενός μικρού αριθμού δραστών. Θα θέλαμε να εκτελέσουμε παρόμοια πειράματα σε περισσότερα σύνολα δεδομένων που ίσως περιλάμβάνουν σύνθετα γεγονότα τα οποία είναι ταυτόχρονα ή επικαλυπτόμενα, περιλαμβάνουν περισσότερους δράστες ή βασίζονται σε πιο περίπλοκους χρονικούς περιορισμούς.

- Η πολιτική συντήρησης ορισμένων πιθανών σημείων έναρξης διαστημάτων κατά την επεξεργασία μίας ροής μπορεί να εφαρμοστεί και σε άλλους αλγόριθμους αναγνώρισης διαστημάτων. Πιο συγκεκριμένα, σκεφτόμαστε την επέκταση του συστήματος RTEC [11] που συμπεραίνει χρονικά διαστήματα πραγματοποίησης δραστηριοτήτων με βάση σημεία έναρξης και τερματισμού τους. Με τον ορισμό μίας συνθήκης που καθορίζει ποια σημεία έναρξης πρέπει να αποθηκευτούν στη μνήμη εργασίας, μία ανάλογη πολιτική θα μπορούσε να εφαρμοστεί στο RTEC.

Βιβλιογραφία

[1] Giatrakos N., Alevizos E., Artikis A. et al.: Complex event recognition in the Big Data era: a survey. The VLDB Journal (2019).

[2] Artikis A., Katzouris N. et al.:A Prototype for Credit Card Fraud Management: Industry Paper. DEBS 2017

[3] Pitsikalis M., Artikis A., Dreo R., Ray C., Comossi E., Jousselme A.:Composite Event Recognition for Maritime Monitoring. n: Proceedings of the 13th ACM International Conference on Distributed andEvent-based Systems, DEBS, pp. 163–174 (2019)

[4] Skarlatidis A., Artikis A., Filipou J., Paliouras G.: A probabilistic logic programming event calculus. TPLP **15**(2), pp. 213–245 (2015)

[5] Alevizos E., Skarlatidis A., Artikis A., Paliouras G.: Probabilistic complex event recognition: A survey. ACM Comput. Surv. **50**(5), 71:1–71:31 (2017)

[6] Raedt L.D., Kimming A., Toivonen H.: ProbLog: A Probabilistic Prolog and its Application in Link Discovery. Machine Learning Lab, Albert-Ludwigs-UniversityFreiburg, Germany

[7] Dries A., Kimmig A., Meert W., Renkens J., den Broeck G.V., Vlasselaer J., Raedt L.D.: Problog2: Probabilistic logic programming. In: Machine Learning and Knowledge Discovery in Databases - European Conference, ECML PKDD 2015, Porto, Portugal, September 7-11, 2015, Proceedings, Part III, pp. 312–315 (2015)

[8] Artikis A., Sergot M., Paliouras G.: A Logic Programming Approach to Activity Recognition. EiMM '10 Proceedings of the 2nd ACM international workshop on Events in multimedia pp. 3-8

[9] Vlassopoulos C., Artikis A.: Towards A Simple Event Calculus for Run-Time Reasoning. Published in COMMONSENSE 2017

[10] Artikis A., Makris E., Paliouras G.: A Probabilistic Interval-based Event Calculus for Activity Recognition. Ann Math Artif Intell (2019).

[11] Artikis A., Sergot M., Paliouras G.: An Event Calculus for Event Recognition. IEEE Transactions on Knowledge and Data Engineering 27:4:895–908 (2015)

[12] Vlassopoulos C.G.: Master Thesis. National and Kapodistrian University of Athens (2019) (In progress) August 2004Multimedia Systems 10(2):164-179

[13] Kowalski, R.A., Sergot, M.J.: A logic-based calculus of events. New Generation Comput. **4**(1), pp. 217–223 (1986)

[14] McCarthy J., Hayes P.J.: Some philosophical problems from the standpoint of artificial intelligence. Stanford University (1968), pp. 18–35

[15] Reiter R.: Knowledge in Action: Logical Foundations for Specifying and Implementing Dynamical Systems. MIT Press (2001), pp. 19–83

[16] Levesque H., Pirri F., Reiter R.: Foundations for the Situation Calculus. Linköping University Electronic Press (1998), pp. 1–8

[17] Kowalski R.:Logic for Problem Solving. Imperial College of Science and Technology, University of London (1979)

[18] Pinto J.A.:Temporal Reasoning in the Situation Calculus. Computer Science University of Toronto (1994)

[19] Kowalski R., Sadri F.:The Situation Calculus and Event Calculus Compared. Department of Computing, Imperial College (1994), pp. 9–10

[20] Kowalski R., Sadri F.:Reconciling the Event Calculus with the Situation Calculus. Department of Computing, Imperial College (1997), pp. 11–13

[21] Van Belleghem K., Denecker M., De Schreye D.:Combining Situation Calculus and Event Calculus. Proceedings of the International Conference on Logic Programming, (1995)

[22] Van Belleghem K., Denecker M., De Schreye D.:On the Relation between Situation Calculus and Event Calculus. The Journal of Logic Programming (1996)

[23] Miller R., Shanahan M.:The Event Calculus in Classical Logic - Alternative Axiomatisations. Linköping University Electronic Press (1999), pp. 8–22

[24] Miller R., Shanahan M.:Some Alternative Formulations of the Event Calculus. University College London, London and Imperial College of Science, Technology and Medicine, London (2002)

[25] Kakas A., Miller R.:A simplereference declarative language for describing narratives with actions. The Journal of Logic Programming (1997)

[26] Kakas A., Miller R.:Reasoning about Actions, Narratives and Ramifications. Linköping University Electronic Press (1998)

[27] Gelfond M., Lifschitz V.:Representing Actions in Extended Logic Programming JICSLP'92, ed. Krzysztof Apt, 560, MIT Press (1992), pp. 866–871

[28] Gelfond M., Lifschitz V.:Representing Action and Change by Logic Programs. The Journal of Logic Programming (1993), 17:301–322

[29] Gelfond M., Lifschitz V.:Classical Negation in Logic Programs and Disjunctive Databases. New Generation Computing (1991), 9:365–385

[30] Allen J.F., Ferguson G.:Actions and Events in Interval Temporal Logic. Journal of Logic and Computation (1994), 4:5:531–579

[31] Doherty P., Gustafsson J., Karlsson L., Kvarnström J.:TAL: Temporal Action Logics Language Specification and Tutorial. Linköping Electronic Articles in Computer and Information Science Vol. 3(1998): nr 015

[32] Kvarnström J.:TALplanner and Other Extensions to Temporal Action Logic. Ph.D. Thesis, Linköping (2005)

[33] Dousson C.:Extending and unifying chronicle representation with event counters. In Proc. of the 15$^{\text{th}}$ ECAI, Lyon, France (2002), pp. 257–261

[34] Dousson C., Le Maigat P.:Chronicle recognition improvement using temporal focusing and hierarchisation. In Proceedings of International Joint Conference on Artificial Intelligence (IJCAI) (2007), pp. 324–329

[35] Shoham Y.:Temporal logic in AI: semantical and ontological considerations. Journal of artificial intelligence (1987), pp. 89–104

[36] Hakeem A., Shah M.:Learning, detection and representation of multi-agent events in videos. In proceedings of Hellenic Conference on Artificial Intelligence, volume 5138. Springer (2008) pp. 586–605

[37] Cervesato I., Montanari A.:A calculus of macro-events. Progress report. In: TIME (2000), pp. 47–58

[38] Chittaro L., Montanari A.:Efficient temporal reasoning in the cached event calculus. Computational Intelligence (1996), 12(3):359–382

[39] Chesani F., Mello P., Montali M., Torroni P.:A Logic-Based, Reactive Calculus of Events. Fundamenta Informaticae, 105(1-2):135–161, (2010)

[40] Busany N., Gal A., Senderovich A., Weidlich M.:Interval-based Queries over Multiple Streams with Missing Timestamps. (2017)

[41] Richardson M., Domingos P.:Markov Logic Networks. Machine Learning 62(1–2), 107–136 (2006)

[42] Skarlatidis A., Paliouras G., Vouros G. A., Artikis A.:Probabilistic Event Calculus based on Markov Logic Networks. In RuleML America, pp. 155–170 (2011)

[43] Artikis A., Weidlich M.:Distribution and Uncertainty in Complex Event Recognition. In RuleML 2015: Rule Technologies: Foundations, Tools and Applications, pp. 70–80

[44] Skarlatidis A., Paliouras G., Artikis A., Vouros G.: Probabilistic event calculus for event recognition. ACM Transactions on Computational Logic16(2), 11:1–11:37(2015)

[45] Biswan R., Thrun S., Fujimura K.:Recognizing activities with multiple cues. In Ahmed M. Elgammal, Bodo Rosenhahn, and Reinhard Klette, editors, Workshop on Human Motion, volume 4814 of Lecture Notes in Computer Science, pp. 255–270. Springer, 2007.

[46] Sanghai S., Domingos P., Weld D.:Learning Models of Relational Stochastic Proccesses. In: Gama, J., Camacho, R., Brazdil, P.B., Jorge, A.M., Torgo, L. (eds.)

[47] Helaoui R., Niepert M., Stuckenschmidt H.:Recognizing interleaved and concurrent activities: A statistical-relational approach. In PerCom, pp. 1–9. IEEE (2011)

[48] Tran S.D., Davis L. S.:Event Modeling and Recognition using Markov Logic Networks. In ECCV, 2008

[49] Morariu V.I., Davis L.S.:Multi-agent event recognition in structured scenarios. In: The 24th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2011, Colorado Springs, CO, USA, 20-25 June 2011, pp. 3289–3296 (2011)

[50] Brendel W., Fern A., Todorovic S.:Probabilistic Event Logic for Interval-Based Event Recognition. The 24th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2011, Colorado Springs, CO, USA, 20-25 June 2011

[51] Wang J., Domingos P.:Hybrid Markov Logic Networks. In Proceedings of the 23rd national conference on Artificial intelligence, volume 2, pp. 1106–1111, 2008.

[52] Sadilek A., Kautz H.:Location-based reasoning about complex multi-agent behavior. Journal of Artificial Intelligence Research, 43:87–133, 2012.

[53] Rabiner L., Juang B.:An introduction to hidden markov models. ASSP Magazine, IEEE, 3(1):4–16, 1986

[54] Rabiner L.:A tutorial on hidden Markov models and selected applications in speech recognition. Proceedings of the IEEE. 77(2):257–286

[55] Sukthankar G., Sycara K.:Robust Recognition of Physical Team Behaviors UsingSpatio-Temporal Models. 5th International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS 2006), Hakodate, Japan, May 8-12, 2006

[56] Singla G., Cook D.J.,, Schmitter-Edgecombe M.:Tracking Activities in Complex Settings Using Smart Environment Technologies. Int J Biosci Psychiatr Technol IJBSPT. 2009 January 1; 1(1): 25–35

[57] Petkovic M., Jonker W., Zivkovic Z.:Recognizing Strokes in Tennis Videos Using Hidden Markov Models. Proceedings of the IASTED International Conference on Visualization, Imaging and Image Processing (VIIP 2001), Marbella, Spain, September 3-5, 2001

[58] Murphy K.P.:Dynamic Bayesian Networks: representation, inference and learning. PhD thesis, University of California, 2002.

[59] Ghahhramani Z.:Learning Dynamic Bayesian Networks. International School on Neural Networks, Initiated by IIASS and EMFCSC NN 1997: Adaptive Processing of Sequences and Data Structures pp 168-197

[60] Park S. Aggarwal J.K.:A hierarchical Bayesian network for event recognition of human actions and interactions. August 2004 Multimedia Systems 10(2):164-179

[61] Lafferty J.D., McCallum A., Pereira F.C.N.:Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In Carla E. Brod- ley and Andrea Pohoreckyj Danyluk, editors, ICML, pages 282–289. Morgan Kaufmann, 2001.

[62] Nazerfard E., Das B., Holder L.B., Cook D.J.:Conditional Random Fields for Activity Recognition in Smart Environments. ACM International Health Informatics Symposium, IHI 2010, Arlington, VA, USA, November 11 - 12, 2010, Proceedings

[63] Michelioudakis E., Skarlatidis A., Paliouras G., Artikis A.:OSLα: Online Structure Learning Using Background Knowledge Axiomatization. In: Frasconi P., Landwehr N., Manco G., Vreeken J. (eds) Machine Learning and Knowledge Discovery in Databases. ECML PKDD 2016. Lecture Notes in Computer Science, vol 9851. Springer, Cham

[64] Lowd D., Domingos P.:Efficient Weight Learning for Markov Logic Networks. In: Kok J.N., Koronacki J., Lopez de Mantaras R., Matwin S., Mladenič D., Skowron A. (eds) Knowledge Discovery in Databases: PKDD 2007. PKDD 2007. Lecture Notes in Computer Science, vol 4702. Springer, Berlin, Heidelberg

Παράρτημα Α΄

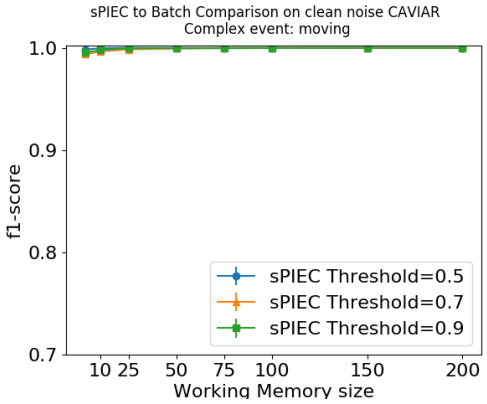
Διαγράμματα Πειραμάτων

Στο παράρτημα αυτό παρουσιάζονται όλα τα διαγράμματα που αφορούν πειραματικά αποτελέσματα με συνδυασμούς παραμέτρων που δεν δείξαμε στο κύριο σώμα της εργασίας.

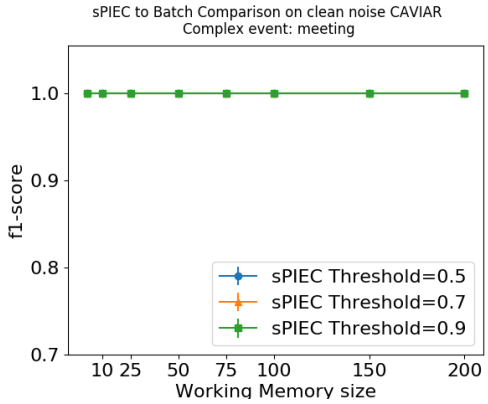
Α′.1 Σύγκρισεις των αποτελεσμάτων του αλγορίθμου ροής δεδομένων με αυτά του αρχικού αλγορίθμου

Α′.1.1 Σύστημα Εισόδου: Prob-EC

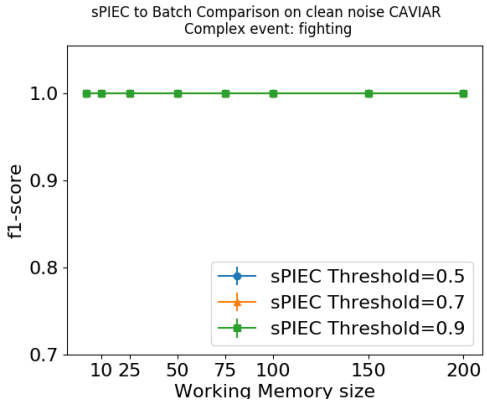
Σχήμα Α′.1: Συγκρίσεις με είσοδο χωρίς θόρυβο



(α′) Σύγκριση για την LTA: συμβάδιση

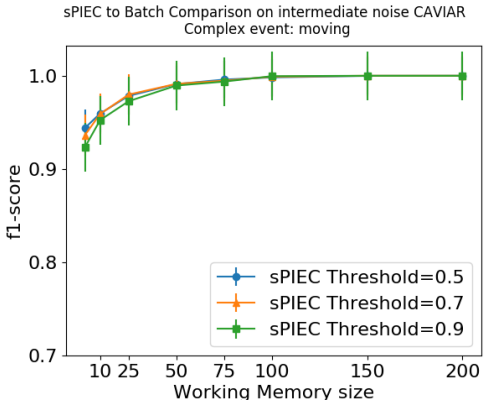


(β′) Σύγκριση για την LTA: συνάντηση

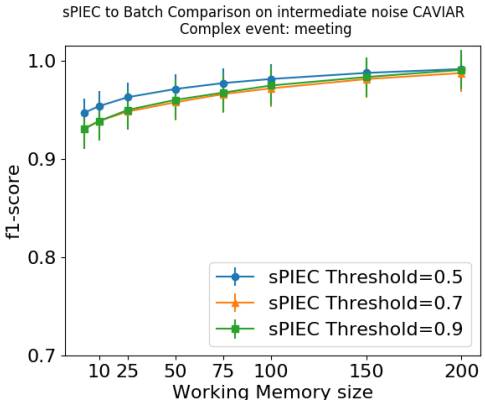


(γ′) Σύγκριση για την LTA: καβγάς

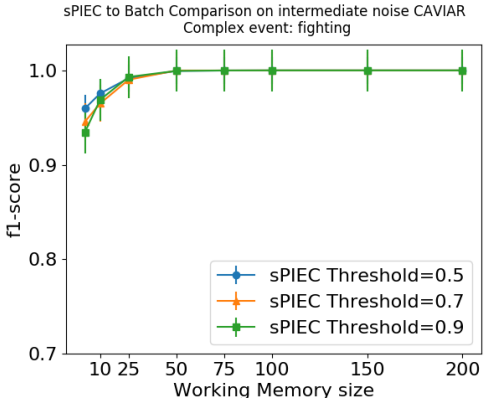
Σχήμα Α΄.2: Συγκρίσεις με είσοδο τα δεδομένα ενδιάμεσου θορύβου



(α΄) Σύγκριση για την LTA: συμβάδιση



(β΄) Σύγκριση για την LTA: συνάντηση

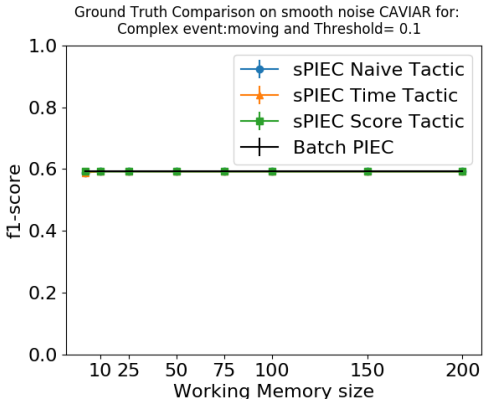


(γ΄) Σύγκριση για την LTA: καβγάς

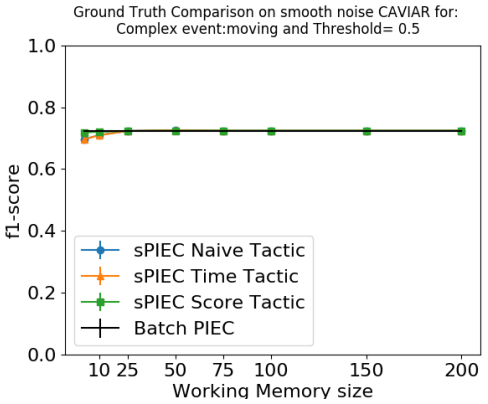
Α′.2 Σύγκριση των αποτελεσμάτων του αλγορίθμου ροής δεδομένων με το ground truth του CAVIAR

Α′.2.1 Σύστημα Εισόδου: Prob-EC

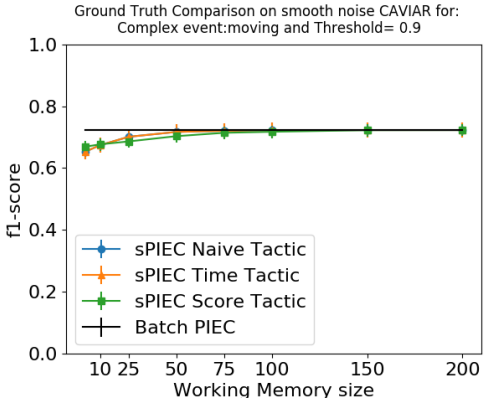
Σχήμα Α′.3: Συγκρίσεις με είσοδο τα δεδομένα ομαλού θορύβου για την LTA: συμβάδιση



(α′) Πιθανοτικό Κατώφλι: 0.1

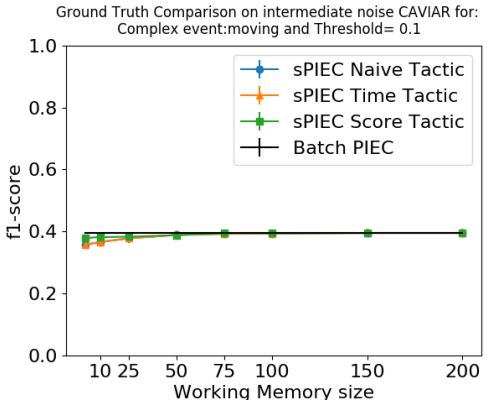


(β′) Πιθανοτικό Κατώφλι: 0.5

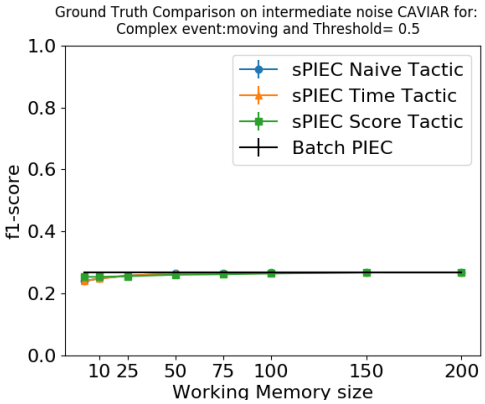


(γ′) Πιθανοτικό Κατώφλι: 0.9

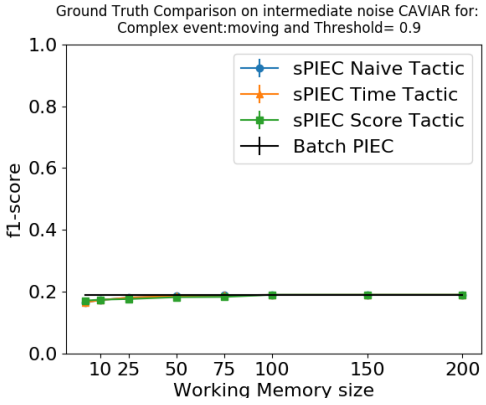
Σχήμα Α΄.4: Συγκρίσεις με είσοδο τα δεδομένα ενδιάμεσου θορύβου για την LTA: συμβάδιση



(α΄) Πιθανοτικό Κατώφλι: 0.1

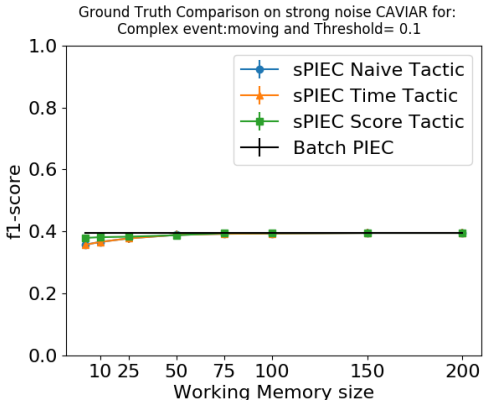


(β΄) Πιθανοτικό Κατώφλι: 0.5

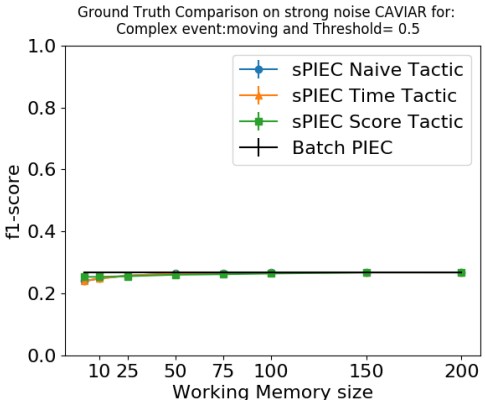


(γ΄) Πιθανοτικό Κατώφλι: 0.9

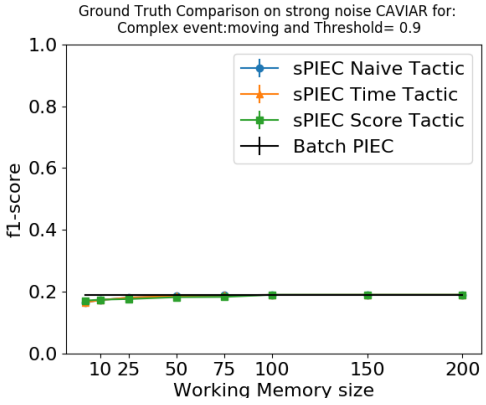
Σχήμα Α΄.5: Συγκρίσεις με είσοδο τα δεδομένα ισχυρού θορύβου για την LTA: συμβάδιση



(α΄) Πιθανοτικό Κατώφλι: 0.1

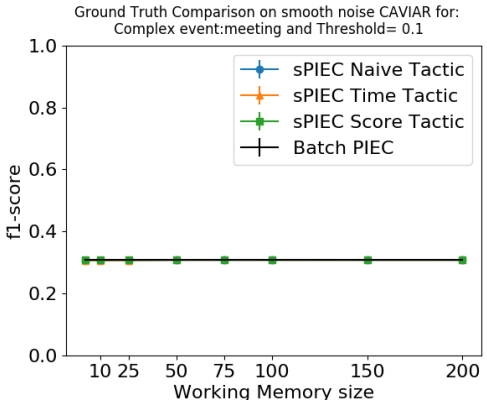


(β΄) Πιθανοτικό Κατώφλι: 0.5

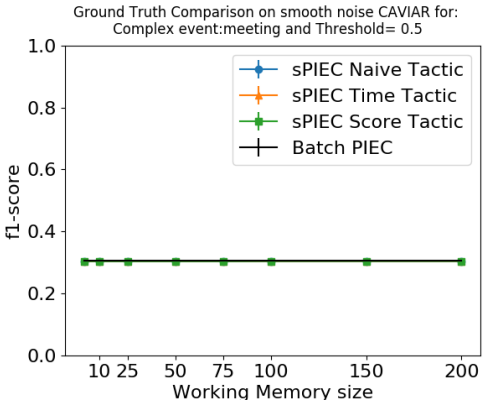


(γ΄) Πιθανοτικό Κατώφλι: 0.9

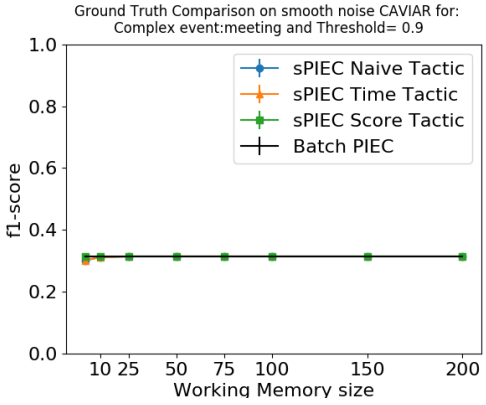
Σχήμα Α΄.6: Συγκρίσεις με είσοδο τα δεδομένα ομαλού θορύβου για την LTA: συνάντηση



(α΄) Πιθανοτικό Κατώφλι: 0.1

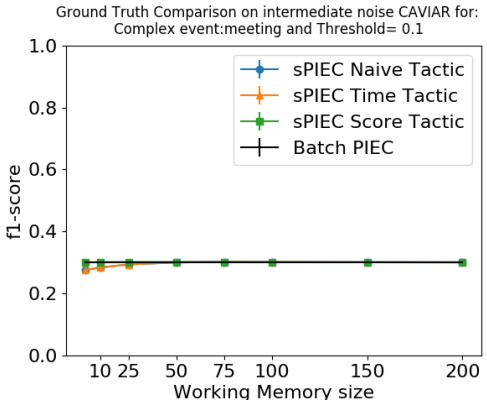


(β΄) Πιθανοτικό Κατώφλι: 0.5

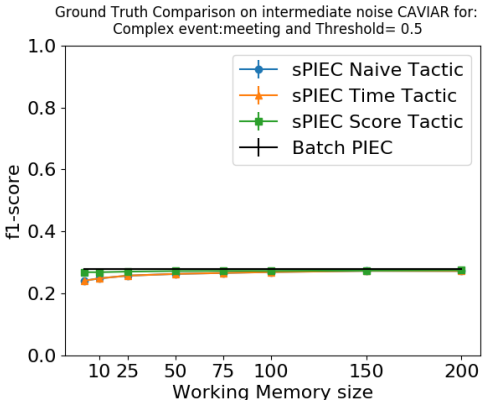


(γ΄) Πιθανοτικό Κατώφλι: 0.9

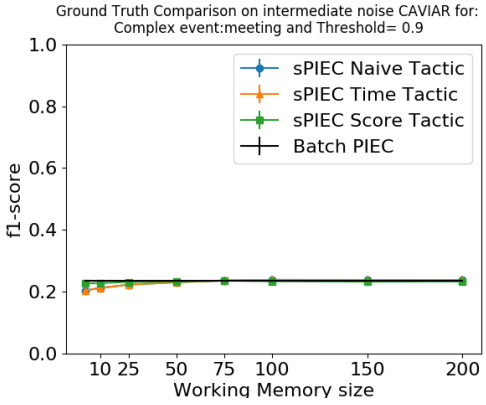
Σχήμα Α΄.7: Συγκρίσεις με είσοδο τα δεδομένα ενδιάμεσου θορύβου για την LTA: συνάντηση



(α΄) Πιθανοτικό Κατώφλι: 0.1

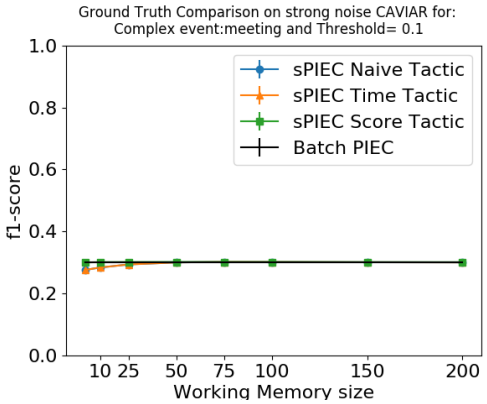


(β΄) Πιθανοτικό Κατώφλι: 0.5

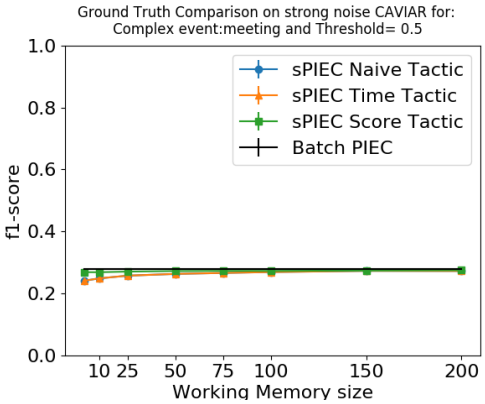


(γ΄) Πιθανοτικό Κατώφλι: 0.9

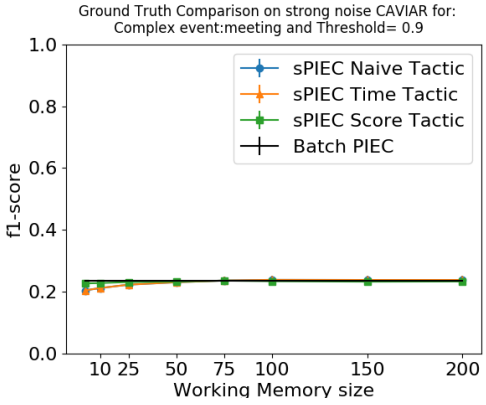
Σχήμα Α΄.8: Συγκρίσεις με είσοδο τα δεδομένα ισχυρού θορύβου για την LTA: συνάντηση



(α΄) Πιθανοτικό Κατώφλι: 0.1



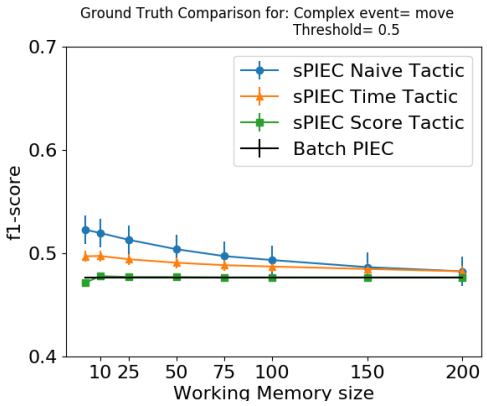
(β΄) Πιθανοτικό Κατώφλι: 0.5



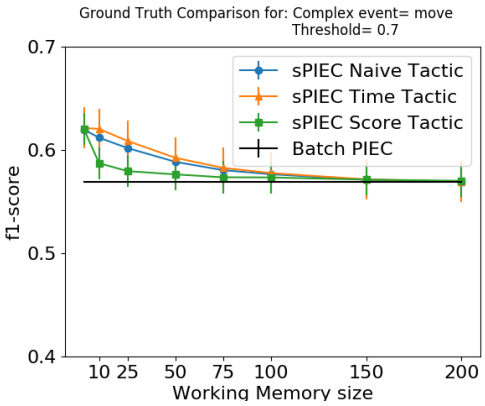
(γ΄) Πιθανοτικό Κατώφλι: 0.9

Α΄.2.2 Σύστημα Εισόδου: OSLα

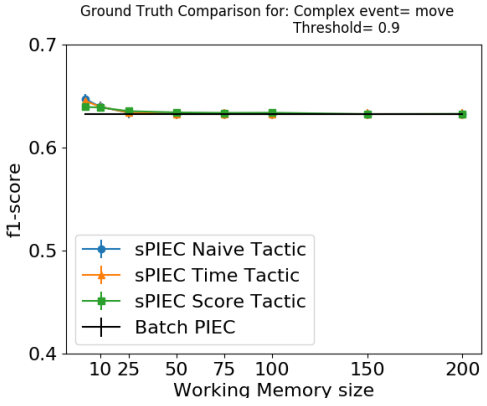
Σχήμα Α΄.9: Συγκρίσεις με είσοδο τα δεδομένα της μεθόδου OSLα με το ground truth για την LTA: συμβάδιση



(α΄) Πιθανοτικό Κατώφλι: 0.5



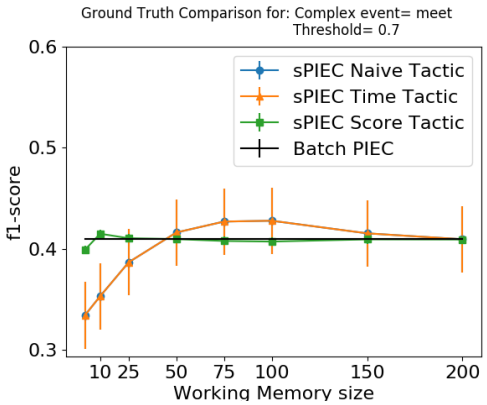
(β΄) Πιθανοτικό Κατώφλι: 0.7



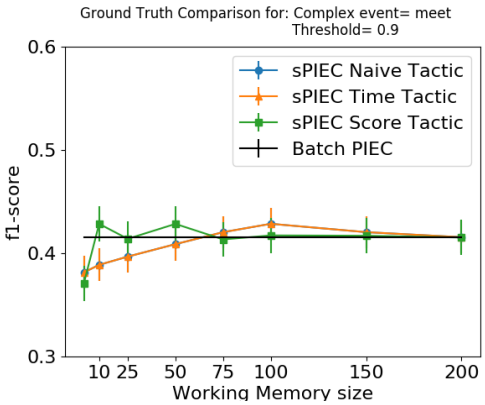
(γ΄) Πιθανοτικό Κατώφλι: 0.9

Α΄.2.3 Σύστημα Εισόδου: DN

Σχήμα Α΄.10: Συγκρίσεις με είσοδο τα δεδομένα της μεθόδου DN με το ground truth για την LTA: συνάντηση



(α΄) Πιθανοτικό Κατώφλι: 0.7

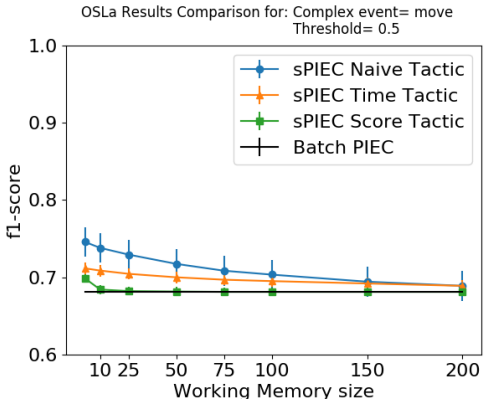


(β΄) Πιθανοτικό Κατώφλι: 0.9

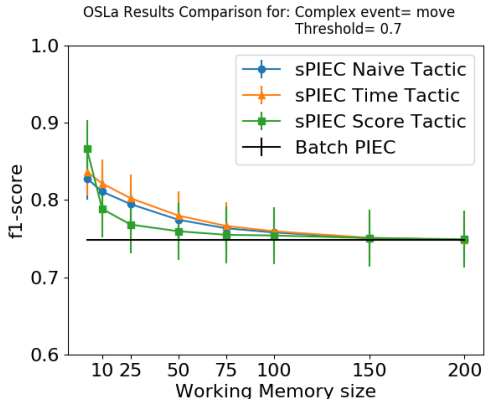
Α΄.3 Σύγκριση των αποτελεσμάτων του αλγορίθμου ροής δεδομένων με την στιγμιαία αναγνώριση του συστήματος εισόδου

Α΄.3.1 Σύστημα Εισόδου: OSLα

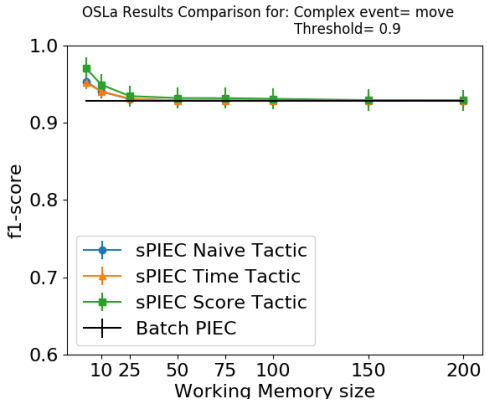
Σχήμα Α΄.11: Συγκρίσεις με είσοδο τα δεδομένα της μεθόδου OSLα με τις στιγμιαίες αναγνωρίσεις της ίδια μεθόδου για την LTA: συμβάδιση



(α΄) Πιθανοτικό Κατώφλι: 0.5

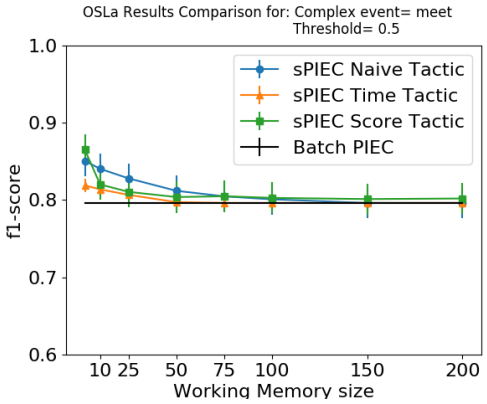


(β΄) Πιθανοτικό Κατώφλι: 0.7

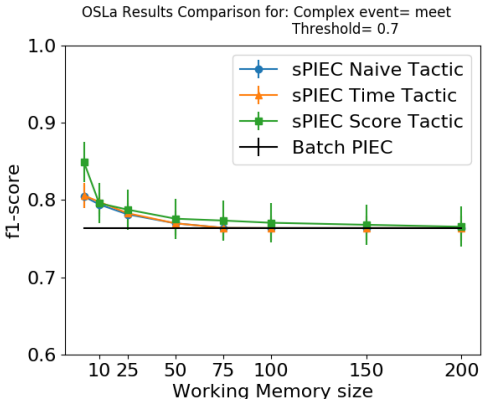


(γ΄) Πιθανοτικό Κατώφλι: 0.9

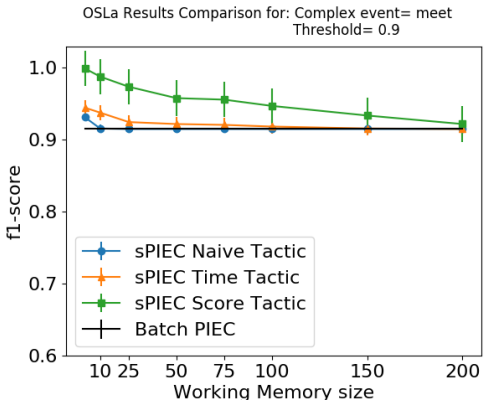
Σχήμα Α′.12: Συγκρίσεις με είσοδο τα δεδομένα της μεθόδου OSLα με τις στιγμιαίες αναγνωρίσεις της ίδια μεθόδου για την LTA: συνάντηση



(α′) Πιθανοτικό Κατώφλι: 0.5



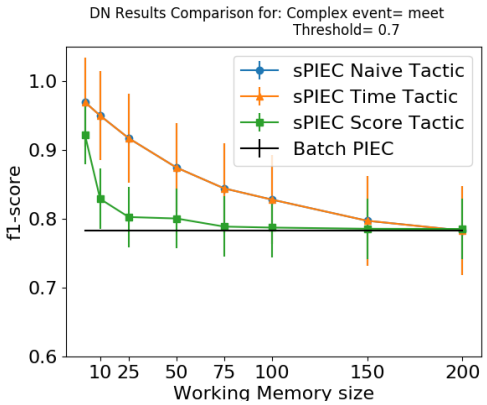
(β′) Πιθανοτικό Κατώφλι: 0.7



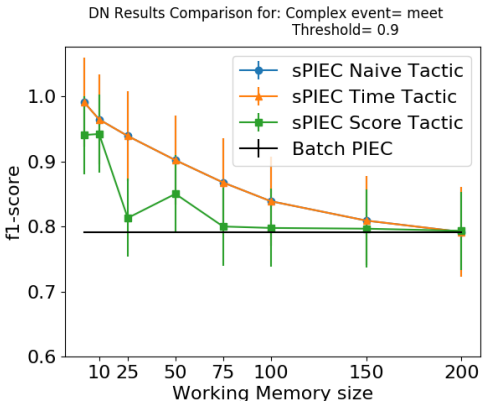
(γ′) Πιθανοτικό Κατώφλι: 0.9

Α′.3.2 Σύστημα Εισόδου: DN

Σχήμα Α′.13: Συγκρίσεις με είσοδο τα δεδομένα της μεθόδου DN με τις στιγμιαίες αναγνωρίσεις της ίδια μεθόδου για την LTA: συνάντηση



(α′) Πιθανοτικό Κατώφλι: 0.7

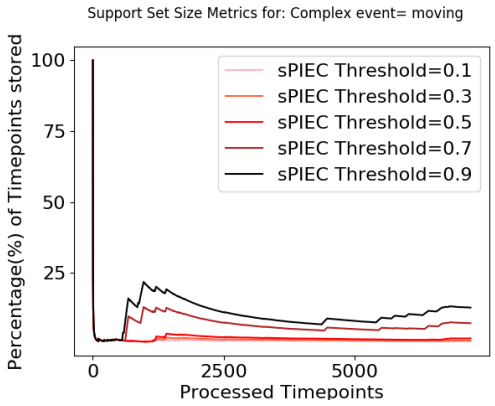


(β′) Πιθανοτικό Κατώφλι: 0.9

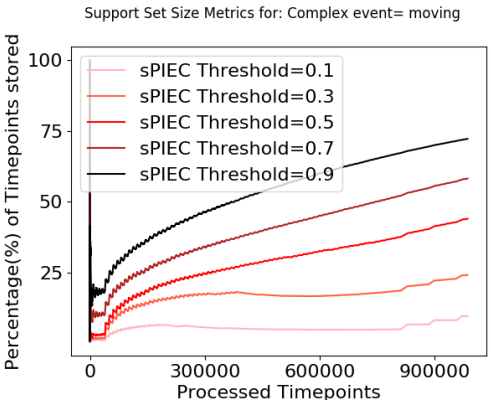
Α′.4 Μετρικές σχετικά με το μέγεθος του support set

Α′.4.1 Σύστημα Εισόδου: Prob-EC

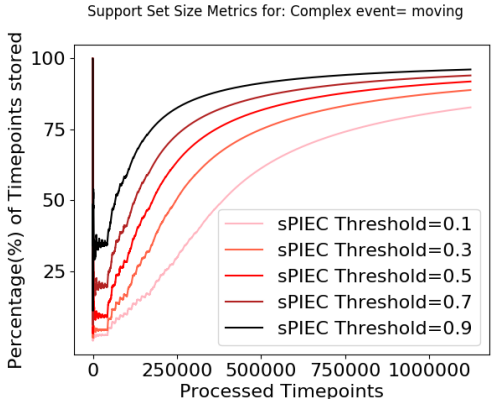
Σχήμα Α′.14: Μέγεθος *support set* κατά την αναγνώριση των δραστηριοτήτων συμβάδισης



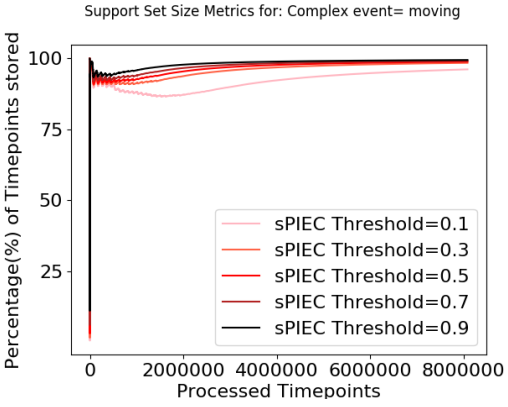
(α′) CAVIAR χωρίς θόρυβο



(β′) CAVIAR ομαλού θορύβου

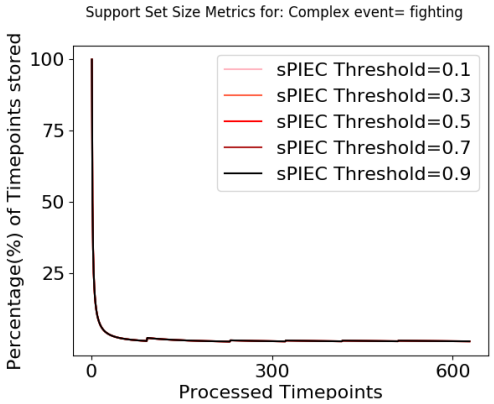


(γ′) CAVIAR ενδιάμεσου θορύβου

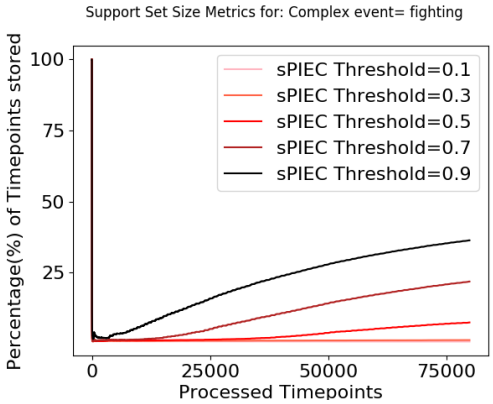


(δ′) CAVIAR ισχυρού θορύβου

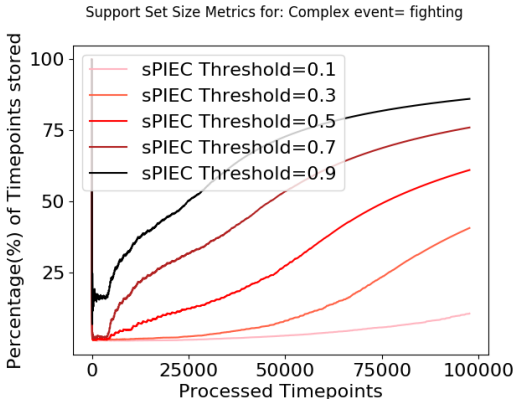
Σχήμα Α΄.15: Μέγεθος *support set* κατά την αναγνώριση των δραστηριοτήτων καβγά



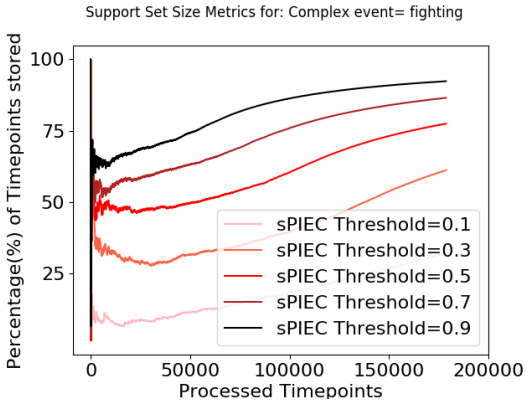
(α΄) CAVIAR χωρίς θόρυβο



(β΄) CAVIAR ομαλού θορύβου



(γ΄) CAVIAR ενδιάμεσου θορύβου



(δ΄) CAVIAR ισχυρού θορύβου