



School of IT
Technical Report



The University of Sydney

**k-STARs: SEQUENCES OF SPATIO-TEMPORAL
ASSOCIATION RULES
TECHNICAL REPORT 589**

FLORIAN VERHEIN

JULY, 2006

k-STARs: Sequences of Spatio-Temporal Association Rules

Florian Verhein

School of Information Technologies, the University of Sydney, Australia
fverhein@it.usyd.edu.au

Abstract

A Spatio-Temporal Association Rule (STAR) describes how objects move between regions over time. Since they describe only a single movement between two regions, it is very difficult to see larger patterns in the dataset by considering only the set of STARs. It is especially difficult on complex datasets where the underlying patterns overlap. At best we will miss important patterns - being unable to “see the forest for the trees”, and at worst this can lead to false interpretations. We introduce the k -STAR pattern which describes the sequences of STARs that objects obey. Since a k -STAR captures sequences of object movements it solves these problems. We also allow space and time gaps between successive STARs, as well as supporting ‘replenishable’ k -STARs so we are able to capture the rich set of patterns that exist in real world data. We define a lattice on the k -STARs that allows the user to drill down and drill up in order to explore the patterns in detail, or view them at a higher level. We introduce two important measures; min-1-support and min-1-confidence that allow us to achieve the above. This paper gives a rigorous theoretical treatment of k -STARs, proving various anti-monotonic and weakly anti-monotonic properties that can be exploited to mine k -STARs efficiently. We describe an algorithm, k -STARMiner, that uses these results to mine the lattice of k -STARs¹.

1 Introduction

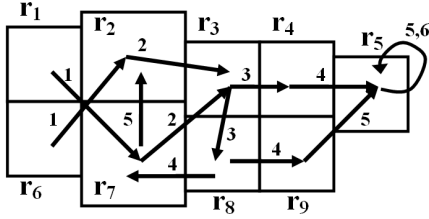
A Spatio-Temporal Association Rule (STAR) tells us how objects move through space over time. In previous work [13] we defined STARs as describing how objects move between regions over time:

¹This research was partially funded by the Australian Research Council (ARC) Discovery Grant, Project ID: DP055900

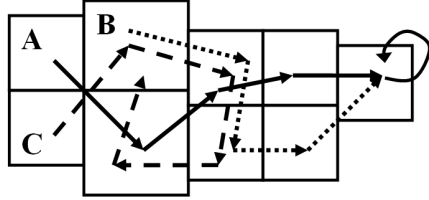
Definition 1 (STAR). $\zeta = (r_a, TI_a) \Rightarrow (r_c, TI_c)$
Objects appearing in region r_a during time interval TI_a will appear in region r_c during time interval TI_c , where $TI_a < TI_c$.

We were interested in those rules with high support and confidence - that is, where enough objects satisfied the rule (*support*), and where the probability that the rule holds was high enough (*confidence*). We considered datasets where many uniquely identifiable objects move throughout different sized regions, such a mobile phone users through the cells of a mobile phone network. The dataset was composed of a sequence of snapshots of the objects’ locations at successive timestamps. The time intervals TI were sets of these timestamps for flexibility. Our algorithm, which we call *STARMiner*, scanned the dataset once and efficiently mined all STARs with sufficient support and confidence.

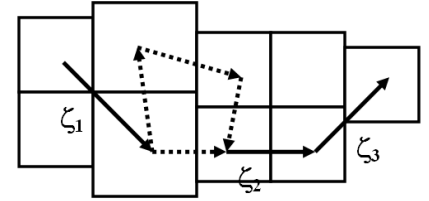
STARs tell us the individual movements that objects make. But this does not tell us anything about their movements beyond the time interval pair $TI' = [TI_a, TI_c]$. Consider Figure 1(a) which shows 9 STARs and an indication of the time intervals for which they apply. Each STAR tells us that the movement it describes is interesting, but apart from the time intervals we have no clue where objects go next. We can have many STARs entering and leaving the same region at roughly the same time, so we cannot tell which path to follow - such as in r_4 . This is especially confusing when many paths intersect. Furthermore, if there is a significant time delay between when a particular group of objects enter and leave the region, we may in fact be misled by the time intervals. For example, there is no reason why objects cannot move from r_1 to r_7 , spread out into other regions (‘go their separate ways’), then converge back into r_7 before moving to r_2 together, rather than immediately moving to r_3 which would be a first assumption. Note that if they merely stayed in r_7 we would have rules telling us this ($r_7 \rightarrow r_7$).



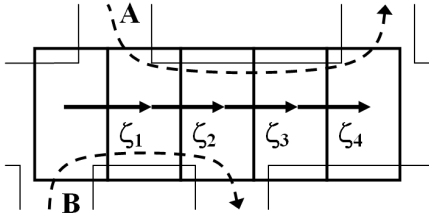
(a) 9 STARs, labeled with a number indicating whether it applies before, after, or at the same time as other STARs.



(b) One possible underlying sequence of movements in (a). k -STARs allow us to mine these sequences, removing the ambiguity. The self-loop tells us that objects remain stationary.



(c) Space and time gaps are allowed, so $\langle \zeta_1, \zeta_2, \zeta_3 \rangle$ is mined.



(d) k -STAR ‘replenishment’ allows us to create $\langle \zeta_1, \zeta_2, \zeta_3, \zeta_4 \rangle$ for mining patterns such as roads.

Figure 1. Motivating examples.

So it should be clear that in a situation such as depicted in Figure 1(a), it is very difficult to draw any conclusions about the overall object trajectories and paths - that is, the *sequence* of movements they make - because we cannot *drill up*. Figure 1(b) shows one possible sequence that could have produced Figure 1(a). It shows three groups of objects $\{A, B, C\}$ and the paths they take. Unlike the Figure 1(a) there is no ambiguity. It also tells us much more useful information that can be used to predict object movements much more accurately. Finally, we can always *drill down* to subsequences for closer examination.

In this paper we present the theory of k -STARs - sequences of STARs that describe the *sequences* of

movements that objects make over time. But in addition to the simple sequences considered above, we also allow quite general patterns that we think are important for studying object mobility datasets:

First, we allow space and time gaps between individual STARs. Figure 1(c) shows three STARs that are considered interesting (ζ_1 , ζ_2 and ζ_3) and some paths that objects follow, but with insufficient support and confidence to be of interest. Specifically, objects move along ζ_1 , then some follow the top path while others follow the bottom path, before they merge again to follow ζ_2 and ζ_3 . The sequence $\langle \zeta_1, \zeta_2, \zeta_3 \rangle$ accurately describes this pattern, but if we did not allow space or time gaps we would miss it. The sequence $\langle \zeta_1, \zeta_2, \zeta_3 \rangle$ is also different to mining $\langle \zeta_1, \zeta_2 \rangle$ and ζ_3 separately, which would tell us that not enough (or indeed any) objects travel the entire sequence.

Secondly, we allow limited ‘replenishment’ of patterns. This is useful because it allows us to mine patterns that are supported by many objects, but they might not all travel down the entire length of the sequence. Consider Figure 1(d) which shows two groups of objects (A and B). Enough objects travel the paths to ensure that each of ζ_1, \dots, ζ_4 are interesting, but none travel the entire sequence. The sequences $\langle \zeta_1, \zeta_2 \rangle$ and $\langle \zeta_2, \zeta_3, \zeta_4 \rangle$ are therefore interesting. But what about $\Upsilon = \langle \zeta_1, \zeta_2, \zeta_3, \zeta_4 \rangle$? since all *consecutive and overlapping* subsequences of Υ of length 2 are interesting ($\langle \zeta_1, \zeta_2 \rangle$, $\langle \zeta_2, \zeta_3 \rangle$, $\langle \zeta_3, \zeta_4 \rangle$) we say (for now²) that Υ is “interesting at level 2”. We do this because there are many situations where such a pattern is useful, such as roads and highways. Not all objects travel their entire length, so we would not get a k -STAR that represents them if we do not allow this ‘replenishment’. Instead, we would simply get many small sequences. This means we would never know that there was a highway. Secondly, the pattern gives a high level view of the objects motion, and the levels (there may be more than one) at which the sequence is interesting tells us a lot about the behaviour of the objects. If we want more detail we can always *drill down* the lattice defined by these k -STARs to explore them in more detail. For example, a section of highway may be particularly interesting. The overlap of consecutive sequences ensures we find only ‘real’ sequences, not just an arbitrary concatenation of sequences. Finally, note that STARs may have $r_a = r_c$ which allows us to express sequences that include the scenario when the objects

²We rigorously define our interestingness measures in Section 3.

remain still. There is an example of this in Figure 1(a) and 1(b). Note that this is different to the time gap scenario, which tells us that objects did *not* remain stationary, but did not generate any interesting movements either.

The above leads to two measures of ‘interestingness’ that apply to k-STARs: *min-l-support* and *min-l-confidence*.

Our **contributions** are as follows:

1. We define k-STARs - sequences of Spatio-Temporal Association Rules - a new type of pattern for mining object mobility databases. These are flexible enough to mine a wide variety of sequences that are interesting in real world applications. They allow us to capture stationary patterns, space and time gaps and replenishable sequences that enable us to mine high level patterns that would otherwise be hidden. We describe a *lattice* on all k-STARs that allows us to *drill up* and *drill down* for exploratory data mining. This greatly improves the ability to interpret the many patterns that are found.
2. We introduce two novel measures for evaluating the interestingness of k-STARs: *min-l-support* and *min-l-confidence*. We prove various anti-monotonic and weakly anti-monotonic properties of *min-l-support* and *min-l-confidence*. We rigorously develop the theory necessary to mine k-STARs from STARs by exploiting these properties. We describe an algorithm that mines all k-STARs using these results.

Section 2 reviews related literature. Section 3 defines k-STARs, their measures and properties of the measures. In Section 4 we provide and prove the lemmas required for efficient mining of k-STARs and describe our algorithm. We also show the data structures we use, and describe the k-STAR lattice. Throughout, we give many examples to help the reader understand the theory. We conclude in Section 5.

2 Related Work

There has been work on spatial association rules (examples include [10, 5]) and temporal association rules (examples include [3, 7]) but little work has addressed both spatial and temporal dimensions. Much of the work that does can be categorised as traditional association rule mining [1] or sequential association rule mining [2] *applied* to a spatio-temporal problem, such as in [9].

Prior to [13], the work by Tao et al. [11] was the only research found that addressed the problem of STARs (albeit briefly) in the Spatial-Temporal domain. As an application of their work they show a brute force algorithm for mining specific spatio-temporal association rules. They considered association rules of the form $(r_i, \tau, p) \Rightarrow r_j$, with the interpretation “if an object is in region r_i at some time t , then with probability p it will appear in region r_j by time $t + \tau$ ”.

Other interesting work that deals with spatio-temporal patterns in the spatio-temporal domain (but are not association type patterns) include [12, 14, 6, 8, 4]. Mamoulis et al. [8] mine *periodic* patterns in objects moving between regions. Wang et al. [14] introduce what they call *flow patterns* that describe the changes of events over space and time. They consider events occurring in regions, and how these events are connected with changes in neighbouring regions as time progresses. Ishikawa et al. [6] describe a technique for mining object mobility patterns in the form of Markov transition probabilities from an indexed spatio-temporal dataset of moving points. In this case, the transition probability p_{ij} of an (order 1) Markov chain is $P(r_j|r_i)$ where r_i and r_j are regions, which is the confidence of a spatio-temporal association rule, although this is not mentioned by the authors. Tsoukatos et al. [12] mine frequent sequences of *non* spatio-temporal values for regions.

The work we have listed above is quite different from our previous work on STARs [13]. Furthermore, except for [4], none of these consider sequences, which is the core of this work. Temporal sequence mining has received significant interest since [2] and many algorithms have been proposed. But this type of transaction database sequence mining is really quite different to the sorts of sequences we consider here. We cannot map k-STARs into these sequence mining algorithms as objects traveling through regions does not translate to transactions and items. Cao et al. [4] make a similar observation for their work. Our work treats the problem of mining sequences of object movements in spatio-temporal data and we deal with many aspects that are specific and unique to such data, such as the ‘replenishment’ concept and space and time gaps.

Cao et al. [4] is the most relevant work. They consider the mining of frequent spatio-temporal sequential patterns, where the patterns are sequences of line segments that approximate an object’s movements over time. Since they consider stings of (x, y, t) co-

ordinates, they cannot mine patterns where there is a space or time gap simply because their pattern cannot express this type of behaviour - it assumes the line segments don't have a break. Since we mine sequences of *STARs*, which apply to two regions and two time intervals, the elements of our sequence are able to express more complicated patterns. Their patterns are also fundamentally different to ours. We consider a set of region while [4] use the object coordinates. We mine patterns supported by many objects, while [4] mine recurring patterns of the *same* object. Hence the research problems addressed are quite different. The ability to mine space and time gaps, as well as replenishable patterns distinguishes our work. Cao et al. also do not consider the confidence of the sequences. Confidence is the conditional probability that an object will satisfy the rule, given that it is in a location where the rule applies. It is therefore very important in using the rules to predict what objects will do. The challenge however is that *confidence* is not anti-monotonic or monotonic, so searches for highly confident rules cannot prune the search space. We show that our *min-l-confidence* measure is weakly anti-monotonic, so we can overcome this problem. We note that instead of *STARs*, we could use the line segments produced by [4]. In this case we could mine data without using regions and be able to produce different rules than in [4]. To the best of our knowledge, the *min-l-confidence* and *min-l-support* measures that form the core of this work are also unique. These measures are tailored to mining the types of spatio-temporal patterns we consider and their properties allow us to build a lattice that can be used to view the *k-STARs* at various levels of abstraction - drilling up or down to explore the *k-STARs*.

3 k-STARs

While *STARs* capture how objects move between regions during each TI' , *k-STARs* capture the *sequence* of k moves that objects make. Hence *STARs* are a special case of *k-STARs* with $k = 1$.

Before developing the theory, we need some **notation and definitions**. Let $TI_A(\zeta)$ be a function returning the time interval in the antecedent of the *STAR* ζ . Similarly, let $TI_C(\zeta)$ return the time interval in the consequent of the rule and let $TI(\zeta)$ be the time interval pair $TI' = [TI_A(\zeta), TI_C(\zeta)]$. Let $R_A(\zeta)$ ($R_C(\zeta)$) return the region in the antecedent (consequent). For example, in the rule $\zeta = (r_1, TI_i) \Rightarrow (r_2, TI_{i+1})$ we have $TI_A(\zeta) = TI_i$,

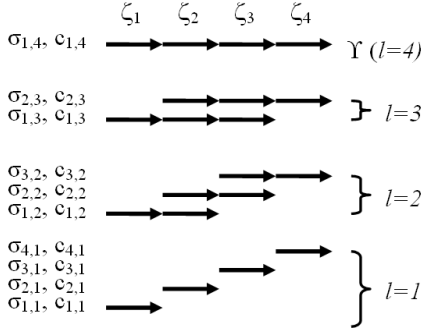
$TI_C(\zeta) = TI_{i+1}$, $R_A(\zeta) = r_1$ and $R_C(\zeta) = r_2$. For convenience define $TI_i < TI_j$ if $i < j$. That is, TI_i occurs before TI_j and they do not overlap. If $TI = [t, t]$, that is, it consists of a single timestamp, we abbreviate it as $TI = t$. Define $O(r, TI)$ as set of objects making an appearance in region r during TI . To keep things compact we also define $O_A(\zeta) = O(R_A(\zeta), TI_A(\zeta))$ and $O_C(\zeta) = O(R_C(\zeta), TI_C(\zeta))$. Let $O(\zeta)$ be the set of objects that follow the rule ζ - that is, the objects that were in $R_A(\zeta)$ during $TI_A(\zeta)$ and then in $R_C(\zeta)$ during $TI_C(\zeta)$. This means that $O(\zeta) = O_A(\zeta) \cap O_C(\zeta)$. Define the *support* of a *STAR* ζ as $\sigma(\zeta) = |O(\zeta)|$ and the support of a region r during time interval TI as $\sigma(r, TI) = |O(r, TI)|$. Define the *confidence* of ζ as $c(\zeta) = \sigma(\zeta) / \sigma(R_A(\zeta), TI_A(\zeta)) = \sigma(\zeta) / |O_A(\zeta)|$ - that is, the fraction of objects that followed the rule, given that they were in the antecedent of the rule. This can be interpreted as the conditional probability that the rule holds: $P(o \in O_C(\zeta) | o \in O_A(\zeta))$. The beginning of Example 3 relates to *STARs*. [13] contains a detailed example of *STARs*.

Definition 2. A *k-STAR* is a sequence of *STARs* $\Upsilon = \langle \zeta_1, \zeta_2, \dots, \zeta_k \rangle : k = 1, 2, \dots$ such that $TI_C(\zeta_i) \leq TI_A(\zeta_{i+1})$ with equality only allowed when $R_C(\zeta_i) = R_A(\zeta_{i+1})$. We say Υ' is a sub-*k-STAR* of Υ , written $\Upsilon' \sqsubset \Upsilon$ if $\Upsilon' = \langle \zeta_i, \zeta_{i+1}, \dots, \zeta_j \rangle : 1 \leq i \leq j \leq k$. That is, a sub-sequence with no gaps.

Example 1. The sequence $\langle (r_1, TI_1) \Rightarrow (r_2, TI_2), (r_2, TI_2) \Rightarrow (r_4, TI_3), (r_5, TI_5) \Rightarrow (r_6, TI_6) \rangle$ is valid. Note that there is a time (and space) gap between the occurrence of the second and the third *STAR* in the sequence and the first two rules share a time interval. The sequence $\langle (r_1, TI_1) \Rightarrow (r_2, TI_2), (r_2, TI_1) \Rightarrow (r_4, TI_2) \rangle$ is not a valid sequence because both rules apply to the same pair of time intervals. $\langle (r_1, TI_1) \Rightarrow (r_2, TI_2), (r_3, TI_2) \Rightarrow (r_4, TI_2) \rangle$ is not valid because $TI_C(\zeta_1) = TI_A(\zeta_2)$ but $R_C(\zeta_1) \neq R_A(\zeta_2)$. $\langle (r_1, TI_1) \Rightarrow (r_1, TI_2), (r_2, TI_3) \Rightarrow (r_4, TI_4) \rangle$ is valid. The first rule tells us that objects remain stationary.

Figure 2(a) gives an example of *sub-k-STARs*.

We allow gaps in time and space between successive rules so that we don't miss interesting rules if some objects stray from the path briefly. Note that objects moving slowly pose no problems, we would just get sequences containing *STARs* with the same region in both the antecedent and consequent.



(a) All 10 *sub-k-STARs* of $\Upsilon = \langle \zeta_1, \zeta_2, \zeta_3, \zeta_4 \rangle$. $\sigma_{i,l}$ and $c_{i,l}$ are defined in Sections 3.1 and 3.2.

$O(r_i, t_j)$	r_1	r_2	r_3
t	$\{a, b, c, d, e, f\}$	\emptyset	$\{h\}$
$t+1$	\emptyset	$\{a, b, c, d\}$	$\{e, f, g, h\}$
$t+2$	$\{c, d\}$	$\{a, b\}$	$\{e, f, g, h\}$
$t+3$	\emptyset	$\{a, b, c, d, e, f, g, h\}$	\emptyset
$t+4$	$\{a\}$	$\{e, f, g\}$	$\{b, c, d, e, h\}$
$t+5$	\emptyset	$\{e, f, h\}$	$\{c, d, e\}$

(b) Example database for objects $\{a, b, c, d, e, f, g, h\}$.

Figure 2. Examples

Example 2. To avoid complicating this example, assume for the moment that the support of a k -STAR is the number of objects that support all the ζ_i and we are looking for k -STARs with support above 3. Consider the data in Figure 2(b). The following sequence is frequent: $\langle (r_1, t) \Rightarrow (r_2, t+1), (r_2, t+3) \Rightarrow (r_3, t+4) \rangle$. This rule says that objects move from region r_1 to r_2 during $[t, t+1]$, then there are no significant movements until they all move from r_2 to r_3 . This is true: during $[t+1, t+3]$ we have two objects that remain in r_2 , and another two make a brief excursion to r_1 . If we did not allow for the case $TI_C(\zeta_i) < TI_A(\zeta_{i+1})$ we would miss the overall movement $r_1 \rightarrow r_2 \rightarrow r_3$. Note that if one of $\{c, d\}$ did not make the excursion to r_1 , the following rule would be frequent: $\langle (r_1, t) \Rightarrow (r_2, t+1), (r_2, t+1) \Rightarrow (r_2, t+2), (r_2, t+2) \Rightarrow (r_2, t+3), (r_2, t+3) \Rightarrow (r_3, t+4) \rangle$. Note the difference between this scenario and the one described above, and the correspondingly different sequences they generate.

An example of objects moving slowly is $\{e, f, g\}$, which support the 3-STAR $\langle (r_3, t+1) \Rightarrow (r_3, t+2), (r_3, t+2) \Rightarrow (r_2, t+3), (r_2, t+3) \Rightarrow r_2, t+4 \rangle$. The reader may notice that g is not always in the dataset. Appearing or disappearing objects pose no problem.

We will however impose a user defined upper limit on how long the time gap between successive

STARs can be. We call this threshold *maxDelay*. We will also make use of a user defined *Neighbourhood function* $N(r, t)$ which returns a set of regions that the user considers ‘neighbours’ t units of time in the future, excluding r . We then restrict the k -STARs by imposing the additional condition $R_A(\zeta_{i+1}) \in N(R_C(\zeta_i), dist(TI_C(\zeta_i), TI_A(\zeta_{i+1})))$, where *dist* is an appropriate measure of the time between two *TIs*. The neighbourhood function and *maxDelay* parameters allow the user to express what k -STARs they consider interesting. These can of course be altered dynamically.

Note that we *cannot* just consider the k -STAR as a sequence of regions and time pairs. This is because the latter cannot express the space or time gaps. The k -STAR expresses more information than just where and at what time objects appear - it also tells us about how the objects moved between the regions. That is, whether a movement is significant or not. If simple region sequences are desired, they are easy to extract from k -STARs using a simple (surjective) function.

We are interested in k -STARs that have support and confidence above a threshold. If the support is above a threshold we shall sometimes call the STAR or k -STAR *frequent*. We also say that an object ‘follows’ or supports a rule if and only if it contributes to the support count of a rule. We will often use the notation $\Upsilon_k = \langle \zeta_1, \zeta_2, \dots, \zeta_k \rangle$, but the subscript of Υ does not always have to be the length of the k -STAR. The length of Υ is $|\Upsilon|$.

3.1 Support of k -STARs

We say an object l -supports a k -STAR $\Upsilon = \langle \zeta_1, \zeta_2, \dots, \zeta_k \rangle$ if it follows a sub- k -STAR of length l . The set of objects that do this is $O(\Upsilon)_l = \bigcup_{i=1}^{k-l+1} (\bigcap_{j=i}^{i+l-1} O(\zeta_j))$. Hence we define the l -support of Υ as $\sigma(\Upsilon)_l = |O(\Upsilon)_l|$, $1 \leq l \leq k$. This means we count objects that ‘entered’ the sequence along the way as long as they support l rules in a row, and have not been counted before. However, we are only interested in those k -STARs for which *each* sub- k -STAR of length l is frequent. If we did not require this (that is, if we use *l-support* as our measure), then there exist cases where a rule has high support but a sub rule of length l that has zero support. This is clearly undesirable. We are therefore interested only in rules where *each* sub- k -STAR of length l is frequent. It doesn’t matter if sequences longer than l are not frequent. Accordingly, we will find it convenient to define $\sigma_{i,l} = |\bigcap_{j=i}^{i+l-1} O(\zeta_j)|$, the support of the sub- k -STAR $\langle \zeta_i, \dots, \zeta_{i+l-1} \rangle$ of length l .

Definition 3. The min- l -support of $\Upsilon = \langle \zeta_1, \zeta_2, \dots, \zeta_k \rangle$ is $\sigma(\Upsilon)_l^{min} = \min_{i \in \{1, \dots, k-l+1\}} \sigma_{i,l}$, $1 \leq l \leq k$.

If we let $l = k$, then $\sigma(\Upsilon_k)_k = \sigma(\Upsilon_k)_k^{min} = \sigma_{1,k} = |\cap_{i=1}^k O(\zeta_i)|$, which is the number of objects that support *all* of the ζ_i . In this case we can drop the *min* superscript. For $k = 1$ these definitions reduce to $\sigma(\zeta_1)$.

Example 3. Consider again Figure 2(b). Let $\zeta_1 = (r_1, t) \Rightarrow (r_2, t+1)$, $\zeta_2 = (r_2, t+3) \Rightarrow (r_3, t+4)$ and $\zeta_3 = (r_3, t+4) \Rightarrow (r_3, t+5)$. Clearly, $\sigma(\zeta_1) = |\{a, b, c, d\}| = 4$, $\sigma(\zeta_2) = |\{b, c, d, e, h\}| = 5$ and $\sigma(\zeta_3) = |\{c, d, e\}| = 3$. Consider the k -STAR $\Upsilon_3 = \langle \zeta_1, \zeta_2, \zeta_3 \rangle$. We have $\sigma_{1,2} = |\{b, c, d\}| = 3$, $\sigma_{2,2} = |\{c, d, e\}| = 3$ and $\sigma_{1,3} = |\{c, d\}| = 2$. Hence $\sigma(\Upsilon_3)_1^{min} = 3$, $\sigma(\Upsilon_3)_2^{min} = 3$ and $\sigma(\Upsilon_3)_3^{min} = 2$.

Varying l gives us great flexibility. With $l < k$ we can find ‘roads’ or similar patterns where objects travel along the road for a while, then turn off. This means we don’t limit our attention to only those objects that support the first rule. In a sense the rule can be replenished by more objects. l determines how long the objects must travel along the ‘road’ before they contribute to the k -STAR. If $l = 1$ we effectively don’t care which objects support the rule, as long as some do. On the other hand, if $l = k$, then we are only interested in rules where enough objects ‘travel the entire length of the rule’. In this case, if extra objects join the ‘road’ in regions other than $O_A(\zeta_1)$, they will not count toward support of the sequence.

We have a number of consequences. Let $\Upsilon_k = \langle \zeta_1, \zeta_2, \dots, \zeta_k \rangle$ and $\Upsilon_{k+1} = \langle \zeta_1, \zeta_2, \dots, \zeta_k, \zeta_{k+1} \rangle$ or $\Upsilon_{k+1} = \langle \zeta_{k+1}, \zeta_1, \zeta_2, \dots, \zeta_k \rangle$. That is, we add the extra ζ to either the front or the back of Υ_k .

Fact 1. $\sigma(\Upsilon_k)_l^{min} \geq \sigma(\Upsilon_{k+1})_l^{min}$. That is, min- l -support is anti-monotonic in k .

Proof. It’s a consequence of the min function. \square

Lemma 1. $\sigma(\Upsilon_k)_l^{min} \geq \sigma(\Upsilon_k)_{l+1}^{min}$. That is, min- l -support is anti-monotonic in l .

Proof. $\sigma(\Upsilon_k)_l = \min_{i \in \{1, \dots, k-l+1\}} |\cap_{j=i}^{i+l-1} O(\zeta_j)| \geq \min \{ \min \{ |\cap_{j=i}^{i+l} O(\zeta_j)| : i \in \{1, \dots, k-l\} \}, \{ |\cap_{j=i}^{i+l-1} O(\zeta_j)| \} \} \geq \min_{i \in \{1, \dots, k-l\}} |\cap_{j=i}^{i+l} O(\zeta_j)| = \sigma(\Upsilon_k)_{l+1}$ by the anti-monotonic property of set intersection. \square

In summary we have $\sigma(\Upsilon_k)_l^{min} \geq \sigma(\Upsilon_k)_{l+1}^{min} \geq \sigma(\Upsilon_{k+1})_{l+1}^{min}$.

This means that if $\Upsilon_k = \langle \zeta_1, \dots, \zeta_k \rangle$ is l_1 -frequent, then it is l -frequent for all $l \leq l_1$. Furthermore, any sub- k -STAR $\Upsilon' = \langle \zeta_i, \dots, \zeta_j \rangle$: $1 \leq i \leq j \leq k$ is also at least min- l_1 -frequent.

3.2 Confidence of k -STARs

The confidence of association rules can be interpreted as the probability that objects support the rule given that they have the *opportunity* to support the rule. Indeed, this applies for traditional association rules and for STARs. For the traditional rule $A \rightarrow B$ [1] the confidence is defined as $P(B|A)$ and only transactions supporting itemset A can support the rule. For STARs, only objects appearing in $R_A(\zeta)$ during $TI_A(\zeta)$ can support ζ . Using this observation, the *confidence* of the k -STAR $\Upsilon_k = \langle \zeta_1, \zeta_2, \dots, \zeta_k \rangle$ is the probability that an object o follows all k consecutive STARs given that it is in $O_A(\zeta_1)$. That is: $P(o \in \cap_{j=1}^k O(\zeta_j) | o \in O_A(\zeta_1))$. But similarly to support, we are also interested in those k -STARs for which *each sub- k -STAR of length l* is confident when $l < k$.

Definition 4. The min- l -confidence of $\Upsilon_k = \langle \zeta_1, \zeta_2, \dots, \zeta_k \rangle$ is $c(\Upsilon_k)_l^{min} = \min_{i \in \{1, \dots, k-l+1\}} P(\cap_{j=i}^{i+l-1} O(\zeta_j) | O_A(\zeta_i))$.

That is, it is the minimum confidence of all the $k-l+1$ sub- k -STARs of length l . This means that if we are given a k -STAR with $c_{min}(\Upsilon_k)_l = \alpha$, we can say that any object appearing in any of the regions at times specified by the antecedents of the rules will follow the rule for at least l steps with probability at least α . This is very useful (and indeed required) for making use of such rules. The special case $l = k$ reduces to the probability that the objects traverse the k -STAR given that they appeared in the first region and time interval referenced in the rule.

All the probabilities we mention regarding objects are measured relative to the total number of objects, N . For example, $P(O_A(\zeta_i)) = |O_A(\zeta_i)|/N$. We will find it useful to let $c_{i,l} = \frac{\sigma_{i,l}}{|O_A(\zeta_i)|}$, the confidence the sub- k -STAR $\langle \zeta_i, \dots, \zeta_{i+l-1} \rangle$ of length l . We therefore have the following result, which follows directly from the definitions:

Fact 2. $c(\Upsilon_k)_l^{min} = \min_{i \in \{1, \dots, k-l+1\}} c_{i,l}$.

We have a number of consequences. Let $\Upsilon_k = \langle \zeta_1, \zeta_2, \dots, \zeta_k \rangle$ and $\Upsilon_{k+1} = \langle \zeta_1, \zeta_2, \dots, \zeta_k, \zeta_{k+1} \rangle$ or $\Upsilon_{k+1} = \langle \zeta_{k+1}, \zeta_1, \zeta_2, \dots, \zeta_k \rangle$ as before.

Fact 3. $c(\Upsilon_k)_l^{min} \geq c(\Upsilon_{k+1})_l^{min}$. That is, min- l -confidence is anti-monotonic in k .

Proof. This is a trivial consequence of the anti-monotonic properties of the min function. \square

Fact 4. $c_{i,l} \geq c_{i,l+1}$

Proof. $c_{i,l} = \frac{\sigma_{i,l}}{\sigma(O_A(\zeta_i))} \geq \frac{\sigma_{i,l+1}}{\sigma(O_A(\zeta_i))} = c_{i,l+1}$ by the anti-monotonicity of support. \square

Lemma 2. *If $p \in \{1, \dots, k-l+1\}$ then $c(\Upsilon_k)_{l+j}^{min} \leq c_{p,l}, \forall j \in \{0, \dots, k-l+1-p\}$.*

Proof. This is true when $j = 0$ by definition of $c(\Upsilon_k)_l^{min}$. Consider $j > 0$. $c(\Upsilon_k)_{l+j}^{min} = \min_{i \in \{1, \dots, k-(l+j)+1\}} c_{i,l+j} \leq \min_{i \in \{1, \dots, k-(l+j)+1\}} c_{i,l+j-1} \dots \leq \min_{i \in \{1, \dots, k-(l+j)+1\}} c_{i,l} \leq c_{p,l}$ since $c_{i,l+j} \leq c_{i,l+j-1} \dots \leq c_{i,l}$ (Fact 4). \square

Note that this works because the denominator of each $c_{i,l}$ is the same as that of $c_{i,l+1}$ for all $i \in \{1, \dots, k-l\}$. Note also that if $p = k-l+1$ then Lemma 2 says only $c(\Upsilon_k)_l^{min} \leq c_{k-l+1,l}$, which is obvious. This is because if we increase l by one in the $c_{i,l}$, the denominator of $c_{k-l+1,l}$ is lost in the sense that it does not appear in any of the resulting $c_{i,l+1}$. That is, by incrementing l we will have one less term in $c(\Upsilon_k)_l^{min}$ as there is one less *sub-k-STAR* of length $l+1$ than there is of length l (The reader may like to refer to Figure 2(a) to see this). However, the lost term (effectively $c_{k-l+1,l+1}$, which we ‘loose’ because it doesn’t exist) might be the smallest, and it might be so only because its denominator is large. In this case it is possible that $c(\Upsilon_k)_l^{min} < c(\Upsilon_k)_{l+1}^{min}$.

Example 4. *Let $\Upsilon = \langle \zeta_1, \dots, \zeta_4 \rangle$ as in Figure 2(a) and suppose $\sigma(\zeta_i) = \sigma_{i,1} = 1 \forall i \in \{1, 2, 3, 4\}$, $|O_A(\zeta_1)| = 1$, $|O_A(\zeta_2)| = 2$, $|O_A(\zeta_3)| = 3$ and $|O_A(\zeta_4)| = 4$. Then $c_{1,1} = 1$, $c_{2,1} = 1/2$, $c_{3,1} = 1/3$ and $c_{4,1} = 1/4$ so $c(\Upsilon)_1^{min} = 1/4$. If we increase l to 2, we might have $\sigma_{i,2} = 1 \forall i \in \{1, 2, 3\}$ (since support is anti-monotonic this is the maximum we may have). Then $c_{1,2} = 1$, $c_{2,2} = 1/2$, $c_{3,2} = 1/3$ and $c(\Upsilon)_2^{min} = 1/3$ which is greater than $c(\Upsilon)_1^{min}$.*

Note that Lemma 2 does not imply that $c(\Upsilon_k)_l^{min} \geq c(\Upsilon_k)_{l+j}^{min}$ for any $j > 0$. That is, the case $c_{p,l} \geq c(\Upsilon_k)_{l+j}^{min} \geq c(\Upsilon_k)_l^{min}$, $j \in \{0, \dots, k-l+1-p\}$ is possible.

Lemma 3. *$c(\Upsilon_k)_l^{min} \geq c(\Upsilon_k)_{l+1}^{min}$, $l < k$ if and only if $\exists i \in \{1, \dots, k-l\} : c_{i,l+1} \leq c_{k-l+1,l}$. That is, $c(\Upsilon_k)_l^{min} < c(\Upsilon_k)_{l+1}^{min}$ if and only if $c_{i,l+1} > c_{k-l+1,l} \forall i \in \{1, \dots, k-l\}$. We say that *min-l-confidence* is weakly anti-monotonic in l .*

Proof. We have $c(\Upsilon_k)_l^{min} = \min_{i \in \{1, \dots, k-l+1\}} c_{i,l} = \min \{ \min_{i \in \{1, \dots, k-l\}} c_{i,l}, c_{k-l+1,l} \} \geq \min \{ \min_{i \in \{1, \dots, k-l\}} c_{i,l+1}, c_{k-l+1,l} \}$ using Fact 4. Now, if $\exists i \in \{1, \dots, k-l\} : c_{i,l+1} \leq c_{k-l+1,l}$ we clearly have $\min \{ \min_{i \in \{1, \dots, k-l\}} c_{i,l+1}, c_{k-l+1,l} \} = \min_{i \in \{1, \dots, k-l\}} c_{i,l+1} = c(\Upsilon_k)_{l+1}^{min}$. On the other hand, if $c_{i,l+1} > c_{k-l+1,l} \forall i \in \{1, \dots, k-l\}$ then $c(\Upsilon_k)_{l+1}^{min} = \min_{i \in \{1, \dots, k-l\}} c_{i,l+1} > c_{k-l+1,l} \geq \min_{i \in \{1, \dots, k-l+1\}} c_{i,l} = c(\Upsilon_k)_l^{min}$. Since we have shown $\alpha \Rightarrow \beta$ and $\bar{\alpha} \Rightarrow \bar{\beta}$ we have $\alpha \iff \beta$, where $\alpha = \exists i \in \{1, \dots, k-l\} : c_{i,l+1} \leq c_{k-l+1,l}$ and $\beta = c(\Upsilon_k)_l^{min} \geq c(\Upsilon_k)_{l+1}^{min}$, $l < k$. \square

The term *weakly anti-monotonic* means it is anti-monotonic in all except perhaps one sub-k-STAR.

Example 5. *Consider $\Upsilon = \langle \zeta_1, \dots, \zeta_4 \rangle$ of Figure 2(a) again. The following are possible: $c_{1,1} = 0.3$, $c_{2,1} = 0.2$, $c_{3,1} = 0.4$, $c_{4,1} = 0.1$, $c_{1,2} = 0.3$, $c_{2,2} = 0.2$, $c_{3,2} = 0.4$, $c_{1,3} = 0.3$, $c_{2,3} = 0.05$, $c_{1,4} = 0.3$. We then have $c(\Upsilon)_1^{min} = 0.1$, $c(\Upsilon)_2^{min} = 0.2$, $c(\Upsilon)_3^{min} = 0.05$, $c(\Upsilon)_4^{min} = 0.3$. Note how this is weakly anti-monotonic. As another example, suppose $c_{1,l} = 0.1$, $l \in \{1, 2, 3, 4\}$, $c_{2,l} = 0.2l \in \{1, 2, 3\}$, $c_{3,l} = 0.3l \in \{1, 2\}$, $c_{4,1} = 0.4$ which is completely anti-monotonic. The case $c_{1,l} = 0.4$, $l \in \{1, 2, 3, 4\}$, $c_{2,l} = 0.3l \in \{1, 2, 3\}$, $c_{3,l} = 0.2l \in \{1, 2\}$, $c_{4,1} = 0.1$ is completely monotonic.*

The following lemma is a sufficient (but not necessary) condition for which we can apply Lemma 3. It is useful because it means we do not need to consider the $c_{i,l+1}$. Instead we use the $c_{i,l}$ which we will already know³ if we mine the rules by joining smaller ones together as outlined in Section 4. This saves considerable calculations. This lemma can be skipped on a first reading.

Lemma 4. *Let $i_{p,l}^{min} = \{i : c_{i,l} \leq c_{j,l} \forall j \in \{p, \dots, k-l+1\}\}$ where $p \in \{1, \dots, k-l\}$ (clearly $i_{p,l}^{min} \subset \{p, \dots, k-l+1\}$). If $\{k-l+1\} \neq i_{p,l}^{min}$ then $c(\Upsilon_k)_l^{min} \geq c(\Upsilon_k)_{l+1}^{min}$. More generally, $c(\Upsilon_k)_l^{min} \geq c(\Upsilon_k)_{k-i+1}^{min} \forall i \in i_{p,l}^{min}$. This gives the condition under which *min-l-confidence* is completely anti-monotonic in l .*

Proof. It suffices to show that $\{k-l+1\} \neq i_{p,l}^{min}$ implies $\exists i \in \{1, \dots, k-l\} : c_{i,l+1} \leq c_{k-l+1,l}$ and use Lemma 3. If $\{k-l+1\} \neq i_{p,l}^{min}$ then we have either $k-l+1 \in i_{p,l}^{min} \wedge |i_{p,l}^{min}| > 1$ or $k-l+1 \notin i_{p,l}^{min} \wedge |i_{p,l}^{min}| \geq 1$ since $i_{p,l}^{min} \neq \emptyset$.

³They are produced in the calculation of $c(\Upsilon_k)_l^{min}$.

In the first case, $c_{i,l} = c_{k-l+1,l} \forall i \in i_{p,l}^{min}$ and in the second case $c_{i,l} < c_{k-l+1,l} \forall i \in i_{p,l}^{min}$. Hence $c_{i,l} \leq c_{k-l+1,l} \forall i \in i_{p,l}^{min}$. Since in both cases $i_{p,l}^{min}$ contains values other than $k-l+1$, it contains at least one value in $\{1, \dots, k-l\}$. That is, $\exists i \in \{1, \dots, k-l\} : c_{i,l} < c_{k-l+1,l}$. Using Fact 4 this implies $\exists i \in \{1, \dots, k-l\} : c_{i,l+1} < c_{k-l+1,l}$ as required.

The second statement trivially holds when $\{k-i+1\} = i_{p,i}^{min}$. Consider any $i' \in i_{p,l}^{min}$, $i' \neq k-i+1$. We consider two cases, $c(\Upsilon_k)_l^{min} \geq c_{i',l+1}$ and $c(\Upsilon_k)_l^{min} < c_{i',l+1}$. In the first case the result follows immediately from Lemma 2. In the second case there must exist an i'' such that $i'' < i'$ for which $c(\Upsilon_k)_l^{min} \geq c_{i'',l+1}$, and again the result follows from Lemma 2. If the required i'' did not exist, then $c(\Upsilon_k)_l^{min} < c_{i,l+1} < c_{i',l} \forall i \in \{1, \dots, i'\}$. But this means there is a $c_{j,l} < c_{i',l}$, $i \in \{i'+1, \dots, k\}$ such that $c(\Upsilon_k)_l^{min} = c_{j,l}$, so i' can't be in $i_{p,l}^{min}$ by the definition of $i_{p,l}^{min}$ and we have a contradiction. \square

Specifically, if $index_{cmin} = \min\{i_{min}\}$ then $c(\Upsilon_k)_l^{min} \geq c(\Upsilon_k)_{k-i+1}^{min} \forall i = \{index_{cmin}, \dots, k-l\}$. The above lemma is very useful whenever $\{k-l+1\} \neq i_{min}$. However, if $\{k-l+1\} = i_{min}$ then we have no option other than to evaluate the $c_{i,l+1}$ (and use Lemma 3). The reader may like to apply the above Lemma to Example 5.

4 Mining k-STARs

In the simplest form, we wish to find all ‘interesting’ k-STARs. We allow the user to specify what they mean by ‘interesting’ using a number of parameters. The user can select the minimum k desired ($minK$) and the minimum value of l ($minL$) for which all k-STARs must be min-l-confident and min-l-frequent. The $minL$ threshold will only be used once $k > minL$. The user can also specify the maximum difference between k and l that they desire ($maxD$). The user must also specify $minSup$, $minConf$ and $maxDelay$ and a *neighbourhood function* $N(r, t)$ as defined in Section 3. Of course the user is free to ignore many of these parameters by setting $maxD = \infty$, $minK = minL = 1$, $maxDelay = \infty$ and $N(\cdot) =$ the set of all regions. However, we suggest $minL > 1$ is required for interesting rules - otherwise objects don't need to travel along the sequence at all.

Our goal therefore is to find all k-STARs with *min-l-support* above $minSup$ and *min-l-confidence*

above $minConf$ for any $l \geq minL$ with $k \geq minK$, $k - l_{max} \leq maxD$ and satisfying $maxDelay$ and $N(\cdot)$. These k-STARs will be output, together with their support and confidence values for the relevant ls (that is, elements of $l_\sigma(\Upsilon)$ and $l_c(\Upsilon)$ as defined below). Note that we do *not* require that a k-STAR must be both *min-l-confident* and *min-l-frequent* for *all* the l we output. A k-STAR might be confident for a higher l than it is frequent for, or vice versa. We don't want to exclude these patterns. For example, suppose Υ is frequent for all $l = 1, \dots, k$ but only confident for $l = minL$. We still want to report to the user that it is frequent for the $k > minL$ because it is interesting. On the other hand if it was not confident for any $l \geq minL$ then Υ would not output it as it does not satisfy $minL$ for any l . Note that $maxD = 0$ is useful for only getting rules that are *maximally* frequent and confident (are min-l-frequent and min-l-confident with $l = k$).

We leverage the anti-monotonic and weak-anti-monotonic properties outlined earlier. We grow the k-STARs from shorter k-STARs by joining them together, exploiting the lemmas in this section.

Let $l_\sigma(\Upsilon)$ be the set of l for which Υ is frequent. That is, $l_\sigma(\Upsilon) = \{l : \sigma(\Upsilon)_l^{min} \geq minSup\}$. Similarly, let $l_c(\Upsilon) = \{l : c(\Upsilon)_l^{min} \geq minConf\}$. Finally, let $l_{\sigma max}(\Upsilon) = \max(l_\sigma(\Upsilon))$ and $l_{c max}(\Upsilon) = \max(l_c(\Upsilon))$ be the maximum values of these sets. Since *min-l-support* is anti-monotonic in l (Lemma 1), $l_\sigma(\Upsilon)$ will always have the form $\{1, 2, 3, \dots, l_{\sigma max}(\Upsilon)\}$. $l_c(\Upsilon)$ on the other hand may have gaps as it is only weakly anti-monotonic (Lemma 3). In the first example of Example 5 for instance, if $minConf = 0.2$ then $l_c(\Upsilon) = \{2, 4\}$. That is, 1 and 3 are not present so the rule is not *min-l-confident* for $l \in \{1, 3\}$ but it is for $l \in \{2, 4\}$.

In the following lemmas, Let $\Upsilon_\alpha = \langle \zeta_1, \dots, \zeta_m \rangle$ and $\Upsilon_\beta = \langle \zeta_{m+1}, \dots, \zeta_{m+n} \rangle$ be non-overlapping and $TI_C(\zeta_{\alpha_m}) \leq TI_A(\zeta_{\beta_{m+1}})$ so that $\Upsilon = \Upsilon_\alpha \cup \Upsilon_\beta = \langle \zeta_1, \dots, \zeta_m, \zeta_{m+1}, \dots, \zeta_{m+n} \rangle$ is a valid k-STAR.

Lemma 5. Joining k-STARs for min-l-support: $l_{\sigma max}(\Upsilon) \leq l_{\sigma max}^u(\Upsilon_\alpha, \Upsilon_\beta)$ (hence $l_\sigma(\Upsilon) \subset \{1, 2, \dots, l_{\sigma max}^u\}$) where $l_{\sigma max}^u(\Upsilon_\alpha, \Upsilon_\beta)$ is given in Figure 3(e).

Proof. This follows from Fact 1, Lemma 1 and the fact that the $l_{\sigma max}(\cdot)$ is the maximum element of $l_\sigma(\cdot)$. That is, any subsequence Υ'_l of length l of Υ_α or Υ_β is also a subsequence of Υ , so if $\sigma(\Upsilon'_l)^{min} < minSup$ then $\sigma(\Upsilon)_j^{min} < minSup$ for $j = l$ by Fact 1 and for all $j > l$ by Lemma 1. On the other

hand if $l_{\sigma_{max}}(\Upsilon_\alpha) = m$ then no subsequence with $\sigma(\Upsilon'_l)^{min} < minSup$ exists and so it is possible that $l_{\sigma_{max}}(\Upsilon) > l_{\sigma_{max}}(\Upsilon_\alpha)$. The same goes for Υ_β . Putting these together in the four combinations gives us the result. \square

Since *min-l-confidence* is only weakly anti-monotonic in l , it is more complicated.

Lemma 6. Joining k-STARs for min-l-confidence: $l_c(\Upsilon) \subset l_c^u(\Upsilon_\alpha, \Upsilon_\beta)$ where $l_c^u(\Upsilon_\alpha, \Upsilon_\beta)$ is given in Figure 3(f).

Proof. It follows from Facts 3 and 4 and the fact that $l_c(\cdot)$ contains the maximum l for which a k-STAR is min-l-confident. That is, any subsequence Υ'_l of length l of Υ_α or Υ_β is also a subsequence of Υ , so if $c(\Upsilon'_l)^{min} < minConf$ then $c(\Upsilon)_j^{min} < minConf$ for $j = l$ by Fact 3 but it might not be true for $j > l$ since *min-l-confidence* is weakly anti-monotonic in l . However, if any subsequence $\Upsilon_{i,l}$ of Υ_α has $c(\Upsilon_{i,l})^{min} < minConf$ then $c(\Upsilon_{i,j})^{min} < minConf$ for all $j \geq l$ by Fact 4 and hence $c(\Upsilon)_j^{min} < minConf$ for all $j \geq l$. On the other hand, if $l_{c_{max}}(\Upsilon_\alpha) = m$ then it is possible that $l_{c_{max}}(\Upsilon) > l_{c_{max}}(\Upsilon_\alpha)$. Nothing like this holds for Υ_β . That is, unlike Lemma 5, we don't have symmetry in Υ_α and Υ_β . Putting these together in the five combinations gives us the result. \square

Figures 3(a) and 3(b) illustrate each of the cases in Lemmas 5 and 6 respectively.

The above lemmas reduce the search spaces for $l_{\sigma_{max}}(\Upsilon)$ and $l_c(\Upsilon)$. In the case of Lemma 5, $l_{\sigma_{max}}^u(\Upsilon_\alpha, \Upsilon_\beta)$ is an upper-bound on $l_{\sigma_{max}}(\Upsilon)$ while in Lemma 6, $l_c^u(\Upsilon_\alpha, \Upsilon_\beta)$ is a superset of $l_c(\Upsilon)$. The following lemmas give us a quick way to perform the search in the remaining spaces $l_{\sigma_{max}}(\Upsilon) \leq l_{\sigma_{max}}^u(\Upsilon_\alpha, \Upsilon_\beta)$ and $l_c(\Upsilon) \subset l_c^u(\Upsilon_\alpha, \Upsilon_\beta)$ to find the desired $l_{\sigma_{max}}(\Upsilon)$ and $l_c(\Upsilon)$. Clearly, if $l_\sigma^u(\Upsilon_\alpha, \Upsilon_\beta) = 1$ then $l_{\sigma_{max}}(\Upsilon) = 1$ and if $l_c^u(\Upsilon_\alpha, \Upsilon_\beta) = \{1\}$ then $l_c(\Upsilon) = \{1\}$. To test whether $\Upsilon = \Upsilon_\alpha \cup \Upsilon_\beta$ is min-l-frequent (min-l-confident) when $l > 1$ and for all $l \leq l_{\sigma_{max}}^u(\Upsilon_\alpha, \Upsilon_\beta)$ ($l \in l_c^u(\Upsilon_\alpha, \Upsilon_\beta)$) we need to evaluate $\sigma(\delta_{l,j})^{min}$ ($c(\delta_{l,j})^{min}$) for all j and l where $\delta_{l,j}$ is the j th k-STAR of length l that overlaps both Υ_α and Υ_β . For example, ex1 in Figure 3(c) lists the maximum number of possible $\delta_{l,j}$ when joining $\Upsilon_\alpha = \langle \zeta_1, \zeta_2, \zeta_3 \rangle$ and $\Upsilon_\beta = \langle \zeta_4, \zeta_5, \zeta_6 \rangle$.

Lemma 7. Υ is min-l-frequent ($l > 1$) if $\min_j (\sigma(\Upsilon_\alpha)_l^{min}, \sigma(\delta_{l,j})^{min}, \sigma(\Upsilon_\beta)_l^{min}) \geq minSup$.

Proof. This follows from the fact that $\sigma(\Upsilon)_l^{min} = \min_j (\sigma(\Upsilon_\alpha)_l^{min}, \sigma(\delta_{l,j})^{min}, \sigma(\Upsilon_\beta)_l^{min})$. \square

Lemma 8. Υ is min-l-confident ($l > 1$) if $\min_j (c(\Upsilon_\alpha)_l^{min}, c(\delta_{l,j})^{min}, c(\Upsilon_\beta)_l^{min}) \geq minConf$ and $l \in l_c^u(\Upsilon_\alpha, \Upsilon_\beta)$.

Proof. This follows from the fact that $c(\Upsilon)_l^{min} = \min_j (c(\Upsilon_\alpha)_l^{min}, c(\delta_{l,j})^{min}, c(\Upsilon_\beta)_l^{min})$. \square

In both Lemmas, if any $x(\Upsilon_Y)_l^{min}$ don't exist (this happens when $l > |\Upsilon_Y|$), they are removed from the calculation. Note $l \leq |\Upsilon_\alpha| + |\Upsilon_\beta|$ so this will happen.

Lemma 7 (Lemma 8) says that to determine the $l_\sigma(\Upsilon)$ ($l_c(\Upsilon)$) and their corresponding support (confidence) values, we need to evaluate all the $\delta_{l,j}$ s corresponding to $l \in \{1, 2, \dots, l_{\sigma_{max}}^u(\Upsilon_\alpha, \Upsilon_\beta)\}$ ($l \in l_c^u(\Upsilon_\alpha, \Upsilon_\beta)$) for support (confidence). It should be clear that Lemma 5 (Lemma 6) has already reduced this search space. Note that when checking the δ s for $l \in \{1, 2, \dots, l_{\sigma_{max}}^u(\Upsilon_\alpha, \Upsilon_\beta)\}$ ($l \in l_c^u(\Upsilon_\alpha, \Upsilon_\beta)$) we can use the anti-monotonic property of min-l-support (min-l-confidence) in k . This is because $l = |\delta_{l,j}|$ in these computations and $\delta_{l,i} \sqsubset \delta_{l',j} \forall i, j, l, l' : 1 \leq l \leq l' \wedge 1 \leq i \leq l' - l + j$. That is, when $\sigma(\delta_{l,i})^{min} < minSup$ ($c(\delta_{l,i})^{min} < minConf$) we stop searching $\{1, 2, \dots, l_{\sigma_{max}}^u(\Upsilon_\alpha, \Upsilon_\beta)\}$ ($l_c^u(\Upsilon_\alpha, \Upsilon_\beta)$) for any l' where $\exists j : \delta_{l,i} \sqsubset \delta_{l',j}$.

Figures 3(c) and 3(d) demonstrate these Lemmas by showing the δ s we need to check. It should be clear that $\delta_{1,j}$ makes no sense as it cannot overlap both Υ_α and Υ_β .

Note that the δ are all that we need to calculate, as we know the $\sigma(\Upsilon_\alpha)_l^{min}$, $\sigma(\Upsilon_\beta)_l^{min}$, $c(\Upsilon_\alpha)_l^{min}$ and $c(\Upsilon_\beta)_l^{min}$ we need for Lemmas 7 and 8 already. This is because we know the $l_\sigma(\Upsilon_\alpha)$, $l_\sigma(\Upsilon_\beta)$, $l_c(\Upsilon_\alpha)$, $l_c(\Upsilon_\beta)$ and their respective *min-l-confidences* and *min-l-supports* since we have already mined Υ_α and Υ_β , by our problem definition. Similarly, we have also generated *all* the new sub-k-STARs of Υ that are now possible. Specifically, all the δ s that are confident or frequent are valid k-STARs. So it should be clear that this procedure not only creates $\Upsilon = \Upsilon_\alpha \cup \Upsilon_\beta$, but also all new sub-k-STARs of Υ . Specifically, we join all *suffixes* of Υ_α to all *prefixes* of Υ_β . Other sub-k-STARs of Υ_α and Υ_β already exist as Υ_α and Υ_β exist.

4.1 Algorithm: k-STARMiner

We saw at the end of the previous section how to join two *arbitrary* k-STARs. When mining *all* rules

ex.	$l_{\sigma_{max}}(\Upsilon_\alpha)$	$l_{\sigma_{max}}(\Upsilon_\beta)$	$l_{\sigma_{max}}^u(\Upsilon_\alpha, \Upsilon_\beta)$
1	3	3	6
2	2	3	2
3	3	2	2
4	2	1	1

(a) Joining $\Upsilon_\alpha = \langle \zeta_1, \zeta_2, \zeta_3 \rangle$ and $\Upsilon_\beta = \langle \zeta_4, \zeta_5, \zeta_6 \rangle$ for min-l-support. Four examples illustrate the four cases in Lemma 5.

ex.	$\delta_{l,j}$
1	$\delta_{2,1} = \langle \zeta_3, \zeta_4 \rangle, \delta_{3,1} = \langle \zeta_2, \zeta_3, \zeta_4 \rangle, \delta_{3,2} = \langle \zeta_3, \zeta_4, \zeta_5 \rangle, \delta_{4,1} = \langle \zeta_1, \zeta_2, \zeta_3, \zeta_4 \rangle, \delta_{4,2} = \langle \zeta_2, \zeta_3, \zeta_4, \zeta_5 \rangle, \delta_{4,3} = \langle \zeta_3, \zeta_4, \zeta_5, \zeta_6 \rangle, \delta_{5,1} = \langle \zeta_1, \zeta_2, \zeta_3, \zeta_4, \zeta_5 \rangle, \delta_{5,2} = \langle \zeta_2, \zeta_3, \zeta_4, \zeta_5, \zeta_6 \rangle, \delta_{6,1} = \langle \zeta_1, \zeta_2, \zeta_3, \zeta_4, \zeta_5, \zeta_6 \rangle$
2	$\delta_{2,1} = \langle \zeta_3, \zeta_4 \rangle$
3	$\delta_{2,1} = \langle \zeta_3, \zeta_4 \rangle$
4	

(c) The $\delta_{l,j}$ of Lemma 7 for Figure 3(a).

$$l_{\sigma_{max}}^u(\Upsilon_\alpha, \Upsilon_\beta) \doteq \begin{cases} l_{\sigma_{max}}(\Upsilon_\alpha) + l_{\sigma_{max}}(\Upsilon_\beta) & \text{if } l_{\sigma_{max}}(\Upsilon_\alpha) = m \text{ and } l_{\sigma_{max}}(\Upsilon_\beta) = n \quad (1) \\ l_{\sigma_{max}}(\Upsilon_\alpha) & \text{if } l_{\sigma_{max}}(\Upsilon_\alpha) < m \text{ and } l_{\sigma_{max}}(\Upsilon_\beta) = n \quad (2) \\ l_{\sigma_{max}}(\Upsilon_\beta) & \text{if } l_{\sigma_{max}}(\Upsilon_\alpha) = m \text{ and } l_{\sigma_{max}}(\Upsilon_\beta) < n \quad (3) \\ \min(l_{\sigma_{max}}(\Upsilon_\alpha), l_{\sigma_{max}}(\Upsilon_\beta)) & \text{if } l_{\sigma_{max}}(\Upsilon_\alpha) < m \text{ and } l_{\sigma_{max}}(\Upsilon_\beta) < n \quad (4) \end{cases}$$

(e) Equation for Lemma 5

$$l_c(\Upsilon_\alpha, \Upsilon_\beta) \doteq \begin{cases} l_c(\Upsilon_\alpha) \cap (l_c(\Upsilon_\beta) \cup \{n+1, \dots, m\}) \cup \{m+1, \dots, m+n\} & \text{if } l_{c_{max}}(\Upsilon_\alpha) = m > n \quad (1) \\ l_c(\Upsilon_\alpha) \cap l_c(\Upsilon_\beta) \cup \{m+1, \dots, m+n\} & \text{if } l_{c_{max}}(\Upsilon_\alpha) = m = n \quad (2) \\ l_c(\Upsilon_\beta) \cap (l_c(\Upsilon_\alpha) \cup \{m+1, \dots, n\}) \cup \{n+1, \dots, n+m\} & \text{if } l_{c_{max}}(\Upsilon_\alpha) = m < n \quad (3) \\ l_c(\Upsilon_\alpha) \cap (l_c(\Upsilon_\beta) \cup \{n+1, \dots, m\}) & \text{if } l_{c_{max}}(\Upsilon_\alpha) < m > n \quad (4) \\ l_c(\Upsilon_\alpha) \cap l_c(\Upsilon_\beta) & \text{if } l_{c_{max}}(\Upsilon_\alpha) < m \leq n \quad (5) \end{cases}$$

(f) Equation for Lemma 6

ex.	a	b	$l_c(a)$	$l_c(b)$	$l_c^u(a, b)$
1	Υ_α	Υ_β	$\{1, 2, 3\}$	$\{1, 3\}$	$\{1, 3, 4, 5, 6\}$
2	Υ'_α	Υ'_β	$\{1, 2, 3, 4\}$	$\{2\}$	$\{2, 3, 4, 5, 6\}$
3	Υ''_α	Υ''_β	$\{1, 2\}$	$\{1, 2\}$	$\{1, 2, 5, 6\}$
4	Υ'_α	Υ'_β	$\{1, 2, 3\}$	$\{1\}$	$\{1, 3\}$
5	Υ_α	Υ_β	$\{1, 3\}$	$\{3\}$	$\{3\}$

(b) Joining $\Upsilon_\alpha = \langle \zeta_1, \zeta_2, \zeta_3 \rangle, \Upsilon_\beta = \langle \zeta_4, \zeta_5, \zeta_6 \rangle, \Upsilon'_\alpha = \langle \zeta_1, \zeta_2, \zeta_3, \zeta_4 \rangle, \Upsilon'_\beta = \langle \zeta_5, \zeta_6 \rangle, \Upsilon''_\alpha = \langle \zeta_1, \zeta_2 \rangle$ and $\Upsilon''_\beta = \langle \zeta_3, \zeta_4, \zeta_5, \zeta_6 \rangle$ for min-l-confidence. Five examples illustrate the five cases in Lemma 6.

ex.	$\delta_{l,j}$
1	$\delta_{2,1}, \delta_{3,1}, \delta_{3,2}, \delta_{4,1}, \delta_{4,2}, \delta_{4,3}, \delta_{5,1}, \delta_{5,2}, \delta_{6,1}$ as in (c).
2	$\delta_{2,1} = \langle \zeta_4, \zeta_5 \rangle, \delta_{3,1} = \langle \zeta_3, \zeta_4, \zeta_5 \rangle, \delta_{3,2} = \langle \zeta_4, \zeta_5, \zeta_6 \rangle, \delta_{4,1} = \langle \zeta_2, \zeta_3, \zeta_4, \zeta_5 \rangle, \delta_{4,2} = \langle \zeta_3, \zeta_4, \zeta_5, \zeta_6 \rangle$, and $\delta_{5,1}, \delta_{5,2}$ and $\delta_{6,1}$ as in (c).
3	$\delta_{2,1} = \langle \zeta_2, \zeta_3 \rangle$ and $\delta_{5,1}, \delta_{5,2}$ and $\delta_{6,1}$ as in (c).
4	$\delta_{3,1}, \delta_{3,2}$ as in ex2 above.
5	$\delta_{3,1}, \delta_{3,2}$ as in (c).

(d) The $\delta_{l,j}$ of Lemma 8 for Figure 3(b).

Figure 3. Examples of Lemmas 5, 6, 7 and 8. Equations for Lemmas 5 and 6.

we join all sub-k-STARs before joining the k-STARs together by traversing the *inverse tree of suffixes of k-STARs* upward. We only join *singe* ζ at a time (that is, $|\Upsilon_\beta| = 1$). The min-l-support (min-l-confidence) tree, which is a subset of the lattice we define later, is defined as follows: let $\Upsilon = \langle \zeta_1, \dots, \zeta_n \rangle$. Then Υ' has a link to Υ in the inverse suffix tree if and only if (1) $\Upsilon' = \langle \zeta_j, \dots, \zeta_n \rangle : 1 < j \leq n$ and (2) $n - j + 1 \in l_\sigma(\Upsilon)$ ($n - j + 1 \in l_c(\Upsilon)$). The first condition says that Υ' is a suffix of Υ and the second says that Υ is min-l-frequent (min-l-confident) when l is the length of Υ' . Figures 4(b) and 4(c) show examples of these trees along *one* cross-section.

The simplest way of growing the k-STARs is to add one STAR to the end at a time. That is, $n = 1$ in Lemmas 5 and 6 and we will use only cases $\{(1), (2)\}$ and $\{(1), (2), (4)\}$ respectively. Also, there will only be one $\delta_{l,j}$ so we drop the j , and the δ_l will correspond precisely to the suffixes in the tree. Figure 4(a) shows the flow of STARs over time in what we call a FlowGraph. It is a circular array of width $w + 1$ where w is set so that it corresponds to

maxDelay. Each cell r, j references all the suffix trees where $r = R_C(\zeta_n)$ and $TI_j = TI_C(\zeta_n)$. This can be seen in Figures 4(b) and 4(c) also.

The **algorithm** is relatively simple. We call STARMiner [13] which efficiently produces all *frequent* STARs, $S_{TI'}$, for the current time interval pair $TI' = [TI_{curr-1}, TI_{curr}]$ as well as $O(\zeta) : \zeta \in S_{TI'}$. That is, all ζ with $\sigma(\zeta) > minSup$. They may or may not be confident because min-l-support is anti-monotonic, while min-l-confidence is only weakly anti-monotonic. For each $\zeta \in S_{TI'}$ we check $R_A(\zeta)$ and its neighbours both at $TI_A(\zeta)$ and back in time through the FlowGraph window. That is, we check each cell $FlowGraph[r][j]$ where $r \in N(R_A(\zeta), j), j \in \{1, 2, \dots, w\}$. For each tree we find, we traverse up the tree and join ζ to the k-STARs using the results from Section 4. We traverse both the confidence and support trees at the same time and keep only those k-STARs that satisfy both the min-l-support and min-l-confidence criteria as defined in Section 4. This creates new inverse suffix trees, which we add to the FlowGraph. We then

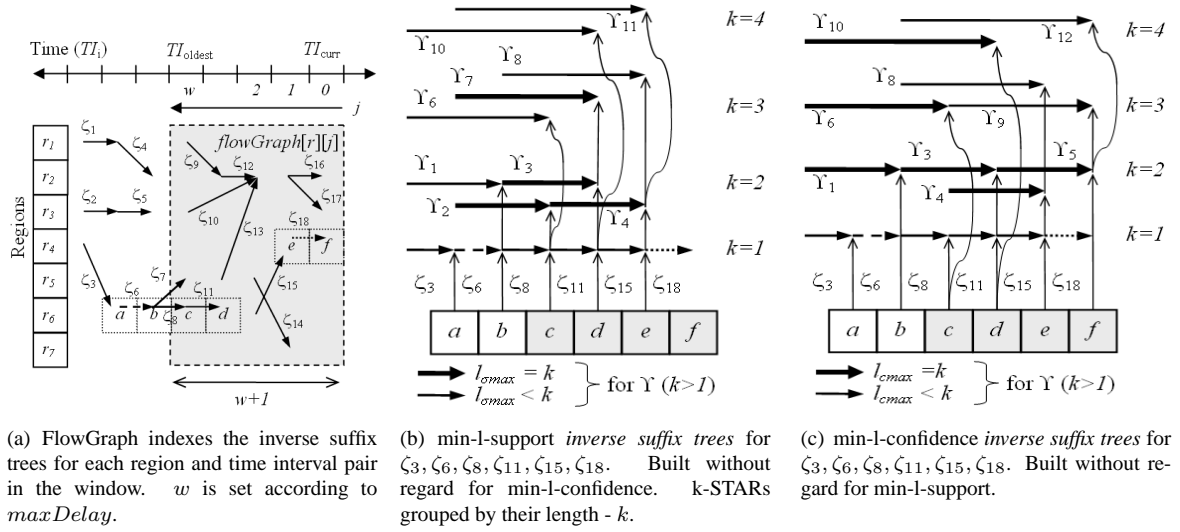


Figure 4. Examples involving $\{\zeta_3, \zeta_6, \zeta_8, \zeta_{11}, \zeta_{15}, \zeta_{18}\}$. $\{\zeta_3, \zeta_8, \zeta_{11}, \zeta_{15}\}$ are confident and frequent. ζ_6 is frequent only. ζ_{18} is confident only.

progress to the next time interval pair, which means the window (width $w + 1$) moves to the right. While doing this, we can easily build a complete lattice of the results as defined in Section 4.2, or simply output the k -STARS for later analysis.

Example 6. Figure 4(a) shows the situation when $TI' = [TI_{curr-1}, TI_{curr}]$, so $S_{TI'} = \{\zeta_{16}, \zeta_{17}, \zeta_{18}\}$. However in this example we will consider the previous $TI'' = [TI_{curr-1}, TI_{curr}]$ where $S_{TI''} = \{\zeta_{14}, \zeta_{15}\}$. Suppose $maxDelay$ and $N(\cdot)$ is set so that d is the only cell found with k -STARS to join to, so we only need to consider the trees starting at ζ_{11} . In Figure 4(b), we first try to join ζ_{11} and ζ_{15} , so we need to check $\delta_2 = \langle \zeta_{11}, \zeta_{15} \rangle$, and we find that $\sigma(\delta_2)_2^{min} \geq minSup$. Hence Υ_4 is a new and interesting k -STAR - so we add it to the newly created tree for ζ_{15} . Since $\sigma(\delta_2)_2^{min} \geq minSup$, we try to join Υ_3 and ζ_{15} . Note that we are traversing up the inverse suffix tree rooted at ζ_{11} . We now need to check $\delta_3 = \langle \zeta_{10}, \zeta_{11}, \zeta_{15} \rangle$ but we find that $\sigma(\delta_3)_3^{min} < minSup$. Hence the maximum l for any other Υ we create by joining ζ_{15} to existing k -STARS will be min- l -frequent with at most $l = 2$. Since $\sigma(\delta_2)_2^{min} \geq minSup$ we know that Υ_8 is frequent and $l_\sigma(\Upsilon_8) = \{1, 2\}$, but it is not maximally frequent. We evaluate $\sigma(\Upsilon_8)_2^{min} = \min\{\sigma(\Upsilon_3)_2^{min}, \sigma(\delta_2)_2^{min}\}$ by Lemma 3(c), using the already known $\sigma(\Upsilon_3)_2^{min}$. Similarly, we know Υ_{11} will be interesting at $l = 2$ since Υ_{10} is, and we evaluate $\sigma(\Upsilon_{11})_2^{min} = \min\{\sigma(\Upsilon_{10})_2^{min}, \sigma(\delta_2)_2^{min}\}$. We have completed the

support suffix tree for (that is, rooted at) ζ_{11} . The procedure for min- l -confidence is similar but in reality, we traverse both trees together and cross prune. For example, Υ_{11} does not satisfy the min- l -confidence criteria (according to Figure 4(c)) so we would not to build Υ_{11} for min- l -support.

We have shown the support and confidence suffix trees independently to avoid complicating matters. Since our goal is to mine k -STARS that are both min- l -confident and min- l -frequent, we only keep those k -STARS for which this holds. In the example, this is the intersection of the k -STARS in the two trees: $\zeta_3, \zeta_6, \zeta_8, \zeta_{11}, \zeta_{15}, \Upsilon_1, \Upsilon_3, \Upsilon_4, \Upsilon_6, \Upsilon_8, \Upsilon_{10}$. That is, we would never have mined the others. As an aside, we implicitly use Lemma 4 by using the suffix tree traversal technique.

We have ignored $maxD$, $minK$ and $minL$ for clarity. They are easy to implement - but note that we can only apply them to prune k -STARS with $l_{cmax} < k$ and $l_{\sigma max} < k$ since short but maximally frequent and confident k -STARS can be grown into longer ones satisfying the constraints. Note also that our algorithm is single pass, and so it suited to stream mining just as STARMiner [13] is.

We do not present experiments for two reasons: (1) lack of real world data. If we had access to such data we could attempt to mine interesting patterns and report such patterns. (2) We could present runtime results, but we feel that the theory is much more important, especially as there is no other algorithm

we can compare k-STARMiner. We could compare it to a brute force technique, but we think this is of little value.

4.2 k-STAR Lattice

The suffix trees we have defined so far are for efficient mining, and are a subset of the k-STAR lattice. We define the *lattice* as linking all k-STARs Υ and Υ' if and only if $\Upsilon' \sqsubset \Upsilon$ and $|\Upsilon'| \in l_\sigma(\Upsilon)$ or $|\Upsilon'| \in l_c(\Upsilon)$. Each node in this lattice is a k-STAR and holds the min-l-confidence and min-l-support value. Using this lattice we can drill down or up through the sequences. Therefore we can explore the resulting k-STARs at high level of abstraction, and *drill down* to relevant sub-k-STARs to find the reasons why a particular k-STAR is interesting. Conversely, we can *drill up* to see how the rules combine together to give the higher level sequences that describe the patterns more coarsely, as well as highlighting long sequences of movements that might be important. Being able to do these things is very useful in making sense of large datasets, and is a key reason behind using k-STARs.

5 Conclusion

We have described useful sequences of Spatio-Temporal Association Rules (k-STARs) that greatly enhance the ability to understand the results of STAR mining. They aggregate individual STARs into sequences that capture interesting patterns such as ‘roads’ and space and time gaps. Furthermore, a lattice defined over the sub-k-STARs enables us to *drill down* or *drill up* to explore the results. The introduction of the *min-l-support* and *min-l-confidence* measures allow us to do these things. We showed that these measures have anti-monotonic and weakly anti-monotonic properties. We rigorously developed the theory required to mine all k-STARs efficiently by exploiting these properties and described an algorithm to do this.

References

- [1] R. Agrawal and R. Srikant. Fast algorithms for mining association rules. In *Proceedings of 20th International Conference on Very Large Data Bases VLDB*, pages 487–499. Morgan Kaufmann, 1994.
- [2] R. Agrawal and R. Srikant. Mining sequential patterns. In *Eleventh International Conference on Data Engineering*, pages 3–14, Taipei, Taiwan, 1995. IEEE Computer Society Press.
- [3] J. M. Ale and G. H. Rossi. An approach to discovering temporal association rules. In *SAC '00: Proceedings of the 2000 ACM symposium on Applied computing*, pages 294–300. ACM Press, 2000.
- [4] H. Cao, N. Mamoulis, and D. Cheung. Mining frequent spatio-temporal sequential patterns. In *Fifth IEEE Conference on Data Mining (ICDM'05)*, 2005.
- [5] Y. Huang, H. Xiong, S. Shekhar, and J. Pei. Mining confident co-location rules without a support threshold. In *Proceedings of the 18th ACM Symposium on Applied Computing ACM SAC*, 2003.
- [6] Y. Ishikawa, Y. Tsukamoto, and H. Kitagawa. Extracting mobility statistics from indexed spatio-temporal datasets. In *STDBM*, pages 9–16, 2004.
- [7] Y. Li, P. Ning, X. S. Wang, and S. Jajodia. Discovering calendar-based temporal association rules. *Data and Knowledge Engineering Special issue: Temporal representation and reasoning*, 44(2):193–218, 2003.
- [8] N. Mamoulis, H. Cao, G. Kollios, M. Hadjieleftheriou, Y. Tao, and D. W. Cheung. Mining, indexing, and querying historical spatiotemporal data. In *KDD '04: Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 236–245. ACM Press, 2004.
- [9] J. Mennis and J. Liu. Mining association rules in spatio-temporal data. In *Proceedings of the 7th International Conference on GeoComputation*, 2003.
- [10] S. Shekhar and Y. Huang. Discovering spatial collocation patterns: a summary of results. In *Proceedings of the 7th International Symposium on Spatial and Temporal Databases SSTD01*, 2001.
- [11] Y. Tao, G. Kollios, J. Considine, F. Li, and D. Papadias. Spatio-temporal aggregation using sketches. In *20th International Conference on Data Engineering*, pages 214–225. IEEE, 2004.
- [12] I. Tsoukatos and D. Gunopulos. Efficient mining of spatiotemporal patterns. In *SSTD '01: Proceedings of the 7th International Symposium on Advances in Spatial and Temporal Databases*, pages 425–442, London, UK, 2001. Springer-Verlag.
- [13] F. Verhein and S. Chawla. Mining spatio-temporal association rules, sources, sinks, stationary regions and thoroughfares in object mobility databases. In *The 11th International Conference on Database Systems for Advanced Applications (DASFAA'06)*, 2006.
- [14] J. Wang, W. Hsu, M.-L. Lee, and J. T.-L. Wang. Flowminer: Finding flow patterns in spatio-temporal databases. In *16th IEEE International Conference on Tools with Artificial Intelligence, 2004 (ICTAI'04)*, pages 14–21, 2004.

School of Information Technologies, J12
The University of Sydney
NSW 2006 AUSTRALIA
T +61 2 9351 4917
F +61 2 9351 3838
www.it.usyd.edu.au

ISBN 1 86487 855 X