The background features a complex, abstract design. It includes a network of thin, light-colored lines forming a web-like structure. Overlaid on this are various data visualizations: a grid of small, light-colored squares on the left, a series of small, light-colored circles in the center, and a series of small, light-colored triangles on the right. The overall color palette is muted, with shades of brown, beige, and light blue.

Sequential Pattern and Sequential Pattern Mining

Sequence Databases & Sequential Patterns

- ❑ Sequential pattern mining has broad applications
 - ❑ Customer shopping sequences
 - ❑ Purchase a laptop first, then a digital camera, and then a smartphone, within 6 months
 - ❑ Medical treatments, natural disasters (e.g., earthquakes), science & engineering processes, stocks and markets, ...
 - ❑ Weblog click streams, calling patterns, ...
 - ❑ Software engineering: Program execution sequences, ...
 - ❑ Biological sequences: DNA, protein, ...
- ❑ Transaction DB, sequence DB vs. time-series DB
- ❑ Gapped vs. non-gapped sequential patterns
 - ❑ Shopping sequences, clicking streams vs. biological sequences


Sequential Pattern and Sequential Pattern Mining

- Sequential pattern mining: Given a set of sequences, find the **complete set of frequent subsequences** (i.e., satisfying the min_sup threshold)

A sequence database

SID	Sequence
10	<a(<u>ab</u> c)(a <u>c</u>)d(cf)>
20	<(ad)c(bc)(ae)>
30	<(ef)(<u>ab</u>)(df) <u>c</u> b>
40	<eg(af)cbc>

A sequence: < (ef) (ab) (df) c b >



- An element may contain a set of **items** (also called **events**)
- Items within an element are unordered and we list them alphabetically

<a(bc)dc> is a subsequence of <a(abc)(ac)d(cf)>

- Given support threshold min_sup = 2, <(ab)c> is a sequential pattern

Sequential Pattern Mining Algorithms

- ❑ Algorithm requirement: Efficient, scalable, finding complete set, incorporating various kinds of user-specific constraints
- ❑ The Apriori property still holds: If a subsequence s_1 is infrequent, none of s_1 's super-sequences can be frequent
- ❑ Representative algorithms
 - ❑ **GSP** (Generalized Sequential Patterns): Srikant & Agrawal @ EDBT'96)
 - ❑ Vertical format-based mining: **SPADE** (Zaki@Machine Learning'00)
 - ❑ Pattern-growth methods: **PrefixSpan** (Pei, et al. @TKDE'04)
- ❑ Mining closed sequential patterns: **CloSpan** (Yan, et al. @SDM'03)
- ❑ Constraint-based sequential pattern mining

The background features a complex network of thin, light-colored lines forming a web-like structure. Scattered throughout are numerous small, colored dots in shades of green, blue, and orange. A prominent, darker, reddish-brown geometric shape, resembling a stylized 'X' or a complex polygon, is centered in the upper half. The overall aesthetic is technical and data-driven.

GSP: Apriori-Based Sequential Pattern Mining



GSP: Apriori-Based Sequential Pattern Mining

- Initial candidates: All singleton sequences
 - <a>, , <c>, <d>, <e>, <f>, <g>, <h>
- Scan DB once, count support for each candidate
- Generate length-2 candidate sequences

$min_sup = 2$

Cand.	sup
<a>	3
	5
<c>	4
<d>	3
<e>	3
<f>	2
<g>	1
<h>	1

	<a>		<c>	<d>	<e>	<f>
<a>	<aa>	<ab>	<ac>	<ad>	<ae>	<af>
	<ba>	<bb>	<bc>	<bd>	<be>	<bf>
<c>	<ca>	<cb>	<cc>	<cd>	<ce>	<cf>
<d>	<da>	<db>	<dc>	<dd>	<de>	<df>
<e>	<ea>	<eb>	<ec>	<ed>	<ee>	<ef>
<f>	<fa>	<fb>	<fc>	<fd>	<fe>	<ff>

	<a>		<c>	<d>	<e>	<f>
<a>		<(ab)>	<(ac)>	<(ad)>	<(ae)>	<(af)>
			<(bc)>	<(bd)>	<(be)>	<(bf)>
<c>				<(cd)>	<(ce)>	<(cf)>
<d>					<(de)>	<(df)>
<e>						<(ef)>
<f>						

SID	Sequence
10	<(bd)cb(ac)>
20	<(bf)(ce)b(fg)>
30	<(ah)(bf)abf>
40	<(be)(ce)d>
50	<a(bd)bcb(ade)>

- Length-2 candidates:
 $36 + 15 = 51$
- Without Apriori pruning:
 $8 * 8 + 8 * 7 / 2 = 92$ candidates

GSP (Generalized Sequential Patterns): Srikant & Agrawal @ EDBT'96)

GSP Mining and Pruning

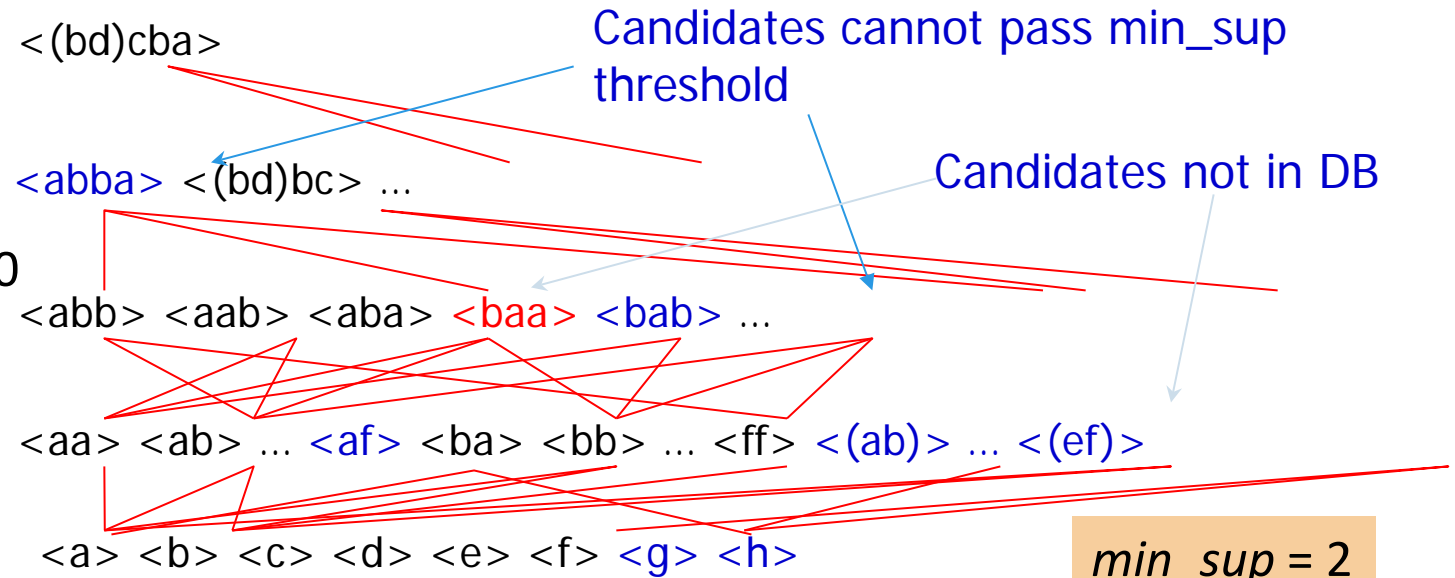
5th scan: 1 cand. 1 length-5 seq. pat.

4th scan: 8 cand. 7 length-4 seq. pat.

3rd scan: 46 cand. 20 length-3 seq. pat. 20 cand. not in DB at all

2nd scan: 51 cand. 19 length-2 seq. pat. 10 cand. not in DB at all

1st scan: 8 cand. 6 length-1 seq. pat.



❑ Repeat (for each level (i.e., length-k))

❑ Scan DB to find length-k frequent sequences

❑ Generate length-(k+1) candidate sequences from length-k frequent sequences using Apriori

❑ set $k = k+1$

❑ Until no frequent sequence or no candidate can be found

min_sup = 2	
SID	Sequence
10	<(bd)cb(ac)>
20	<(bf)(ce)b(fg)>
30	<(ah)(bf)abf>
40	<(be)(ce)d>
50	<a(bd)bcb(ade)>

The background features a complex, abstract design. It includes a network of red lines connecting green dots, resembling a graph or data structure. There are also faint, overlapping geometric shapes and patterns in shades of purple, blue, and orange. A white banner with a subtle geometric pattern runs across the middle, containing the title text.

SPADE: Sequential Pattern Mining in Vertical Data Format

Sequential Pattern Mining in Vertical Data Format: The SPADE Algorithm

- ❑ A sequence database is mapped to: <SID, EID>
- ❑ Grow the subsequences (patterns) one item at a time by Apriori candidate generation

SID	Sequence
1	<a(<u>abc</u>)(a <u>c</u>)d(cf)>
2	<(ad)c(bc)(ae)>
3	<(ef)(<u>ab</u>)(df) <u>cb</u> >
4	<eg(af)cbc>

$min_sup = 2$


Ref: SPADE (Sequential
Pattern Discovery
using Equivalent Class)
[M. Zaki 2001]

SID	EID	Items
1	1	a
1	2	abc
1	3	ac
1	4	d
1	5	cf
2	1	ad
2	2	c
2	3	bc
2	4	ae
3	1	ef
3	2	ab
3	3	df
3	4	c
3	5	b
4	1	e
4	2	g
4	3	af
4	4	c
4	5	b
4	6	c

a		b		...
SID	EID	SID	EID	...
1	1	1	2	
1	2	2	3	
1	3	3	2	
2	1	3	5	
2	4	4	5	
3	2			
4	3			

ab			ba			...
SID	EID (a)	EID(b)	SID	EID (b)	EID(a)	...
1	1	2	1	2	3	
2	1	3	2	3	4	
3	2	5				
4	3	5				

aba				...
SID	EID (a)	EID(b)	EID(a)	...
1	1	2	3	
2	1	3	4	

The background features a complex, abstract design. It includes a network of red lines connecting green dots, resembling a graph or a spatial data visualization. There are also faint, overlapping geometric shapes and patterns in shades of purple, blue, and orange. A white banner with a subtle geometric pattern runs horizontally across the middle of the image, serving as a backdrop for the title text.

PrefixSpan: Sequential Pattern Mining by Pattern-Growth

PrefixSpan: A Pattern-Growth Approach

SID	Sequence
10	<a(<u>a</u> bc)(a <u>c</u>)d(cf)>
20	<(a <u>d</u>)c(bc)(a <u>e</u>)>
30	<(e <u>f</u>)(<u>a</u> b)(d <u>f</u>) <u>c</u> b>
40	<eg(a <u>f</u>)c <u>b</u> c>

Prefix	Suffix (Projection)
<a>	<(abc)(ac)d(cf)>
<aa>	<(_bc)(ac)d(cf)>
<ab>	<(_c)(ac)d(cf)>

□ Prefix and suffix

□ Given <a(abc)(ac)d(cf)>

□ Prefixes: <a>, <aa>, <a(ab)>, <a(abc)>, ...

□ Suffix: Prefixes-based projection

□ PrefixSpan Mining: Prefix Projections

□ Step 1: Find length-1 sequential patterns

□ <a>, , <c>, <d>, <e>, <f>

□ Step 2: Divide search space and mine each projected DB

□ <a>-projected DB,

□ -projected DB,

□ ...

□ <f>-projected DB, ...

PrefixSpan (Prefix-projected
Sequential pattern mining)
Pei, et al. @TKDE'04

PrefixSpan: Mining Prefix-Projected DBs

SID	Sequence
10	<a(<u>a</u> bc)(a <u>c</u>)d(cf)>
20	<(ad)c(bc)(ae)>
30	<(ef)(<u>a</u> b)(df) <u>c</u> b>
40	<eg(af)cbc>

Length-1 sequential patterns
<a>, , <c>, <d>, <e>, <f>

prefix <a>

<a>-projected DB

<(abc)(ac)d(cf)>

<(_d)c(bc)(ae)>

<(_b)(df)cb>

<(_f)cbc>

Length-2 sequential patterns
<aa>, <ab>, <(ab)>,
<ac>, <ad>, <af>

prefix

-projected DB

prefix <c>, ..., <f>

...

... ..

prefix <aa>

<aa>-projected DB

..

prefix <af>

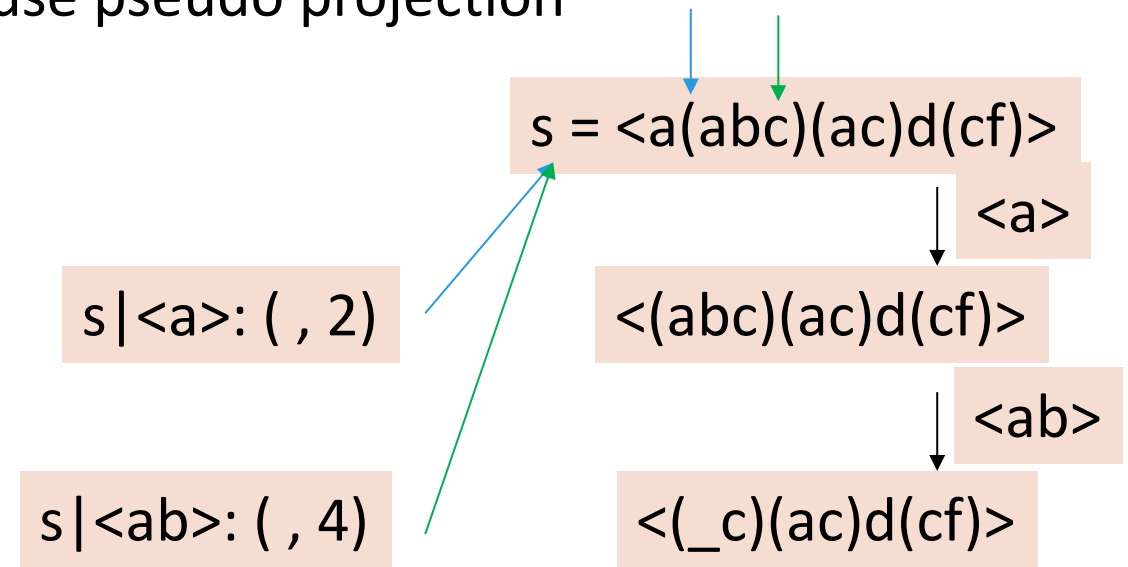
<af>-projected DB

Major strength of PrefixSpan:

- No candidate subseqs. to be generated
- Projected DBs keep shrinking

Implementation Consideration: Pseudo-Projection vs. Physical Projection

- ❑ Major cost of PrefixSpan: Constructing projected DBs
 - ❑ Suffixes largely repeating in recursive projected DBs
- ❑ When DB can be held in main memory, use pseudo projection
 - ❑ No physically copying suffixes
 - ❑ **Pointer to the sequence**
 - ❑ **Offset of the suffix**
- ❑ But if it does not fit in memory
 - ❑ Physical projection
- ❑ Suggested approach:
 - ❑ Integration of physical and pseudo-projection
 - ❑ Swapping to pseudo-projection when the data fits in memory

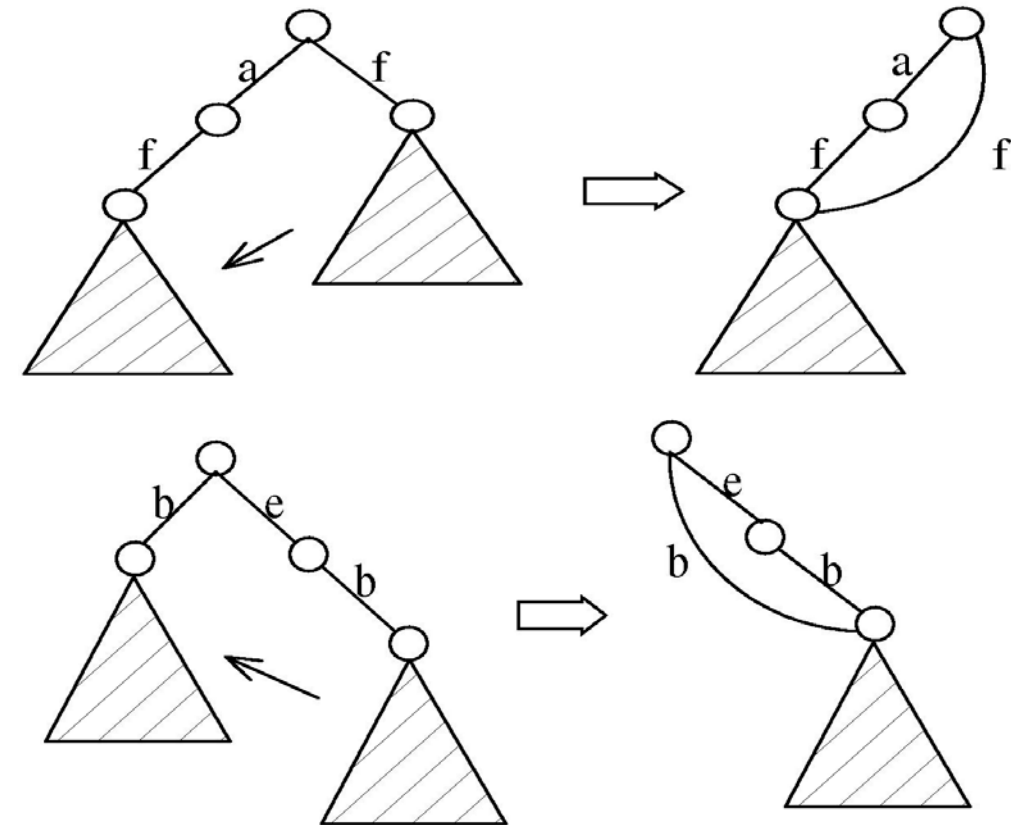


The background features a complex, abstract design. It includes a grid of small plus signs on the left, a network of red lines with green dots in the center, and a white geometric shape on the right. A small inset in the top left shows a heatmap with orange and red areas. The title text is centered within the white shape.

CloSpan: Mining Closed Sequential Patterns

CloSpan: Mining Closed Sequential Patterns

- ❑ A **closed sequential pattern** s : There exists no superpattern s' such that $s' \supset s$, and s' and s have the same support
- ❑ Which ones are closed? $\langle abc \rangle: 20$, $\langle abcd \rangle: 20$, $\langle abcde \rangle: 15$
- ❑ Why directly mine closed sequential patterns?
 - ❑ Reduce # of (redundant) patterns
 - ❑ Attain the same expressive power
- ❑ Property P_1 : If $s \supset s_1$, s is closed iff two project DBs have the same size
- ❑ Explore **Backward Subpattern** and **Backward Superpattern** pruning to prune redundant search space
- ❑ Greatly enhances efficiency (Yan, et al., SDM'03)





Summary

Summary: Sequential Pattern Mining

- ❑ Concepts of Sequential Pattern Mining
- ❑ Sequential Pattern Mining Algorithms
 - ❑ **GSP** (Generalized Sequential Patterns)
 - ❑ Vertical Format-Based Mining: **SPADE**
 - ❑ Pattern-Growth Methods: **PrefixSpan**
- ❑ Mining Closed Sequential Patterns: **CloSpan**

Recommended Readings

- ❑ R. Srikant and R. Agrawal, “Mining sequential patterns: Generalizations and performance improvements”, EDBT’96
- ❑ M. Zaki, “SPADE: An Efficient Algorithm for Mining Frequent Sequences”, Machine Learning, 2001
- ❑ J. Pei, J. Han, B. Mortazavi-Asl, J. Wang, H. Pinto, Q. Chen, U. Dayal, and M.-C. Hsu, "Mining Sequential Patterns by Pattern-Growth: The PrefixSpan Approach", IEEE TKDE, 16(10), 2004
- ❑ X. Yan, J. Han, and R. Afshar, “CloSpan: Mining Closed Sequential Patterns in Large Datasets”, SDM'03