

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	<e <u>g</u> (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

