

## PrefixSpan: A Pattern-Growth Approach

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

Prefix	Suffix (Projection)	
<a></a>	<(abc)(ac)d(cf)>	
<aa></aa>	<(_bc)(ac)d(cf)>	
<ab></ab>	<(_c)(ac)d(cf)>	
4		

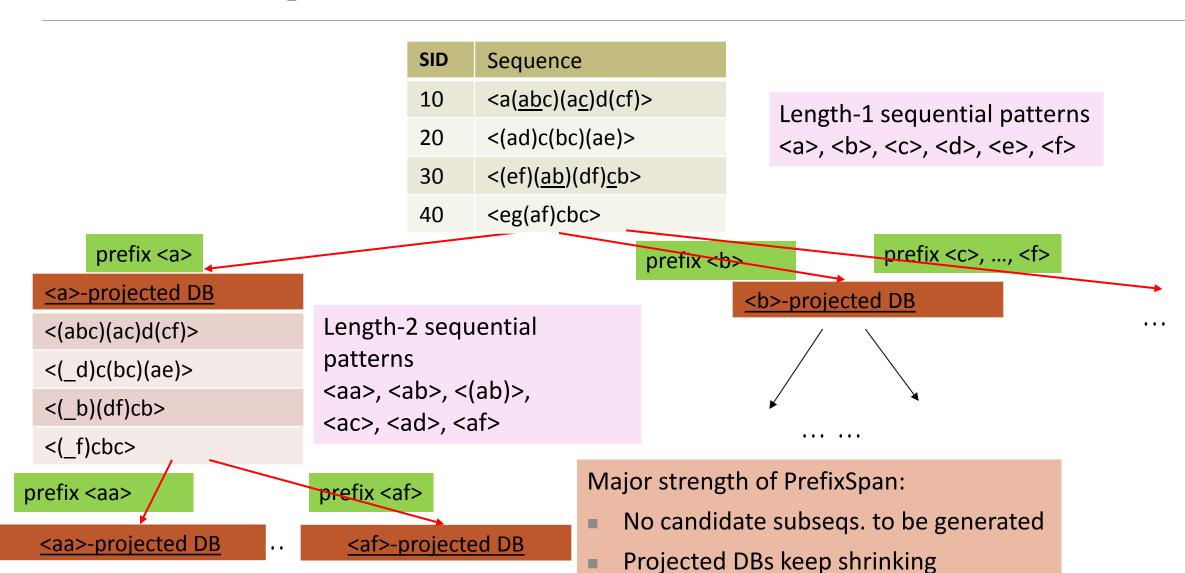
- Prefix and suffix
- Given <a(abc)(ac)d(cf)>
- Prefixes: <a>, <aa>,
  <a(ab)>, <a(abc)>, ...
  - Suffix: Prefixes-based projection

- PrefixSpan Mining: Prefix Projections
  - Step 1: Find length-1 sequential patterns
    - □ <a>, <b>, <c>, <d>, <e>, <f>
  - Step 2: Divide search space and mine each projected DB
    - <a>-projected DB,
    - <b>-projected DB,

    - <f>-projected DB, ...

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

## PrefixSpan: Mining Prefix-Projected DBs



## 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

