Scalable CP approach for Mining Frequent Sequence with gap constraints

http://sites.uclouvain.be/cp4dm/spm/

John O.R. Aoga1, Pierre Schaus2, Tias Guns2
1UCLouvain, ICTEAM, Belgium — 2KU Leuven, DTAI Research group, Belgium
john.aoga@uclouvain.be

Abstract

Sequence mining is an important tool for analyzing large databases of timed events, such as in click stream mining and event log mining. Recently, constraint programming (CP) approaches for pattern mining are gaining interest, due to the modularity of the framework and flexibility to add additional constraints. While CP systems were less scalable than specialized mining systems, we recently showed this can be overcome by hybridizing advanced CP techniques (trailing) with algorithmic improvements. In this work, we study the more involved task of mining under the restriction that the time gap between two matching events must be smaller than a threshold.

We show that this too can benefit greatly from hybridization.

Problem of SPM under gap(M,N)

Find all patterns \( p = (p_0, p_1, \ldots, p_l) \) such that at least \( 0 \) sequences are matched by the pattern satisfying the gap constraints. A sequence \( S \in SDB \) is matched by \( p \) iff \( \exists (i_1, i_2, \ldots, i_j) \) such that:

1. \( i_1 = \epsilon \) or \( i_1 < \epsilon \) and \( i_1 \leq i_j = \epsilon \) or \( i_j > \epsilon \)
2. \( \epsilon \) is the matching position of item \( p_j \) in a sequence \( S \).

Objective

The main objective of this project is to design new constraint in CP to improve the literature.

Sample of SDB

\[
\begin{array}{|c|c|c|}
\hline
\text{sid} & \text{seq} & \text{link}\text{Pos}\text{in}\text{List} & \text{link}\text{Pos}\text{in}\text{Map} \\
\hline
\hline
\text{sid1} & (ABD(C)B) & [B1,B2,(C1,D1),(A1)] & [A1, B1, C1, D1, A1] \\
\hline
\text{sid2} & (B) & [B1,B2,(C1,D1)] & [A1, B1, C1, D1] \\
\hline
\text{sid3} & (ABDBDEBC) & [C1,D1,(B1),(B2),(A1)] & [A1, B1, C1, D1] \\
\hline
\text{sid4} & (ACC) & [B1,C1,(A1)] & [A1, B1, C1] \\
\hline
\end{array}
\]

CP Model

A constraint model consists of variables, domains, and constraints. A CP model over \( P = \{P_0, P_1, \ldots, P_l\} \) is integer variables represents the frequent sequence pattern with threshold \( \theta \), iff the following conditions are satisfied by every valid assignment to \( P \):

1. \( P_k \neq \epsilon \) (\( \epsilon \) represents empty character and the end of pattern)
2. \( N \subset \{i_1, \ldots, L \} \quad \text{iff} \quad P_i = \epsilon 
3. \#(\text{seq}_s) \in SDB \quad \text{if} \quad (P_i \leq s) \quad \text{iff} \quad \theta = \max(1, \{i \mid i \in L \} \{P_i \neq \epsilon\})

Related Work

Specialised Methods:

- PrefixSpan [6]: prefix and prefix-projection profile with DFS;
- eSPADE [8]: vertical database with join rules in DFS/BB;
- LAP-DM-SPAM [7]: idea of least position of items;

CP-based Methods:

- CP-PRISM [4]: one constraint per sequence + reified constraints;
- CP and GapEmbs [3, 2]: global constraint with filtering inspired of PrefixSpan method + maximal gap constraint;
- PPIC [1]: last position of items applied in prefix-projection and Trail-based backing up aware data structure.

PPIC (without gap)

Trail-based backing up aware data structure (PPIC Improvement 1)

PPIC with gap challenge

A \( \text{gap}(M,N) \) constraint changes when a subsequence is included in a sequence, namely it gap between two subsequent symbols \( \geq M \) and \( \leq N \).

Trail-based backing up aware data structure (PPIC Improvement 1)

PPIC-improved backtrack-aware data structures (PPICgap)

Implementation is done in Scala with Oscar Solver [5].

References


BENELEARN 2016
BELGIAN-DUTCH CONFERENCE ON MACHINE LEARNING
https://www.kuleuven-kalck.be/benelearn/