Data Mining 2: Finden häufiger Muster Sommer 2018
Christian Borgelt ab 24.05.2018

4. Übungsblatt

Aufgabe 14 Eclat-Algorithmus

a) Führen Sie den Eclat-Algorithmus (auf der Grundlage des Präfixbaum-Schemas) für die rechts gezeigte Transaktionsdatenbank und einen minimalen Support von $s_{\min} = 3$ aus!

$a\ d\ f$	$a\ b\ d$
$a\ c\ d\ e$	$b\ d\ e$
$b\ d$	$b\ c\ e\ g$
$b\ c\ d$	$c\ d\ f$
$b\ c$	$a\ b\ d$

b) Wie wird im Eclat-Algorithm der Support bestimmt?
Wie unterscheidet sich dieses Schema von dem, das vom Apriori-Algorithmus benutzt wird? Was sind die Vor- und Nachteile dieses Zählverfahrens?

c) Warum nutzt Eclat (normalerweise) die Apriori-Eigenschaft nicht vollständig aus? Wie könnte man die Apriori-Eigenschaft vollständig ausnutzen?

d) Warum ist es im Eclat-Algorithmus vorteilhaft, die weniger häufigen Items zuerst zu verarbeiten? (Hinweis: Betrachten Sie die durchschnittliche Länge der Listen von Transaktionskennungen.)

Aufgabe 15 Eclat-Algorithmus

a) Warum ist eine Bitvektor-Darstellung der Transaktionslisten nicht unbedingt besser als eine explizite Liste der Transaktionskennungen? (Hinweis: Was passiert mit den Listen der Transaktionskennungen in der rekursiven Verarbeitung?)

b) In den bedingten Datenbanken, die in der Suche nach häufigen Itemmengen auftreten, kann man oft Transaktionen zusammenfassen, die durch Eliminieren von Items gleich geworden sind. Wie kann man dies im Eclat-Algorithmus ausnutzen?

c) Zeigen Sie, daß eine Differenzmenge (*diffset*) eines Items (bzgl. einer bedingten Datenbank), d.h., $D(a \mid I) = K(I) - K(I \cup \{a\})$, sowohl größer als auch kleiner sein kann als ihre Überdeckung (*cover*) (abhängig von der Datenbank).

Aufgabe 16 Item-Reihenfolge

a) Wie kann man im Apriori-Algorithmus eine lokale Item-Reihenfolge ausnutzen?

b) Wie kann man im Eclat-Algorithmus eine lokale Item-Reihenfolge ausnutzen? Warum ist dies in diesem Algorithmus besonders einfach?

c) Wie kann man die Ausgabe der gefundenen häufigen Itemmengen durch die Reihenfolge, in der die Itemmengen in einem Teile-und-Herrsche-Verfahren besucht werden, optimieren? Geht dies auch für eine lokale Item-Reihenfolge? Warum?

Aufgabe 17 LCM-Algorithmus

a) Wie funktioniert das Occurrence-Deliver-Schema des LCM-Algorithmus? Wie verhält es sich zum Eclat-Algorithmus? Wie unterscheidet es sich? Welche zusätzlichen Informationen/Daten werden für den LCM-Algorithmus benötigt (verglichen mit Eclat)?

b) Warum erlaubt der LCM-Algorithmus, die Suche nach häufigen Itemmengen mit einem Speicherbedarf proportional zur Größe der Transaktionsdatenbank durchzuführen, während der Eclat-Algorithmus dies nicht erlaubt, obwohl die Verarbeitungsschemata ähnlich sind?
(Hinweis: Betrachten Sie, in welcher Reihenfolge die Teilprobleme gelöst werden.)

c) Führen Sie den LCM-Algorithmus für die rechts gezeigte Transaktionsdatenbank und einen minimalen Support von $s_{\min} = 3$ aus!

$a\ d\ f$	$a\ b\ d$
$a\ c\ d\ e$	$b\ d\ e$
$b\ d$	$b\ c\ e\ g$
$b\ c\ d$	$c\ d\ f$
$b\ c$	$a\ b\ d$