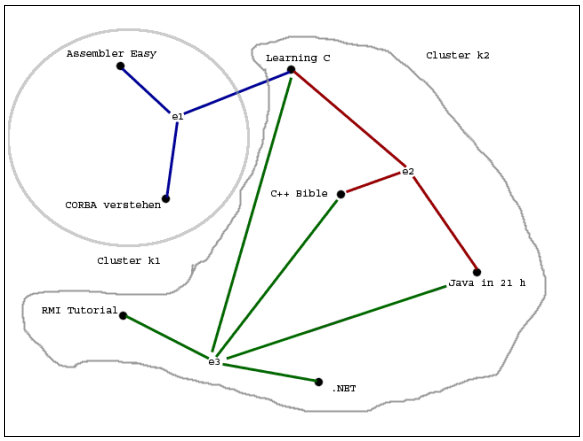
Eberhard Karls
Universität
Tübingen

Wilhelm–Schickard–Institut für Informatik
Paralleles Rechnen

Seminarausarbeitung

Vorstellung der Arbeit

„Recommendation systems: a probabilistic analysis"



Ausarbeitung von
Robert Sösemann, Dezember 2002

Betreuer: Prof. Michael Kaufmann

Inhaltsverzeichnis

I Abstract

Die vorliegende Seminararbeit versteht sich als zusammenfassender Überblick eines Artikels des IBM Almaden Research Centers.
Unter dem Titel „Recommendation Systems: a probabilistic analysis"- zu finden auf [www.almaden.ibm.com/cs/k53/algopapers/focs98rec.ps] - beschreiben dort die Autoren R. Kumar et al. ein einfaches und wahrscheinlichkeitstheoretisch fundiertes Modell für passives Collaboratives Filtering.

Statt einer einfachen Wiederholung der Ergebnisse, liegen die Schwerpunkte dieser Ausarbeitung in der Einleitung und Motivation des Themenkomplexes und einer Vorstellung der beschriebenen Algorithmen.

II Einleitung und Motivation

1 Herausforderungen im elektronischen Handel

Die Vorteile die das Internet - vor allem der elektronische Handel – mit sich bringen sind weitestgehend bekannt. Sowohl für Verkäufer wie auch Käufer heißt elektronischer – also automatisierter Geschäftsverkehr vor allem eines - Zeit- und Kosteneinsparung.

Wo früher ausschließlich teures und auch - fehlbares - menschliches Personal Transaktion abgewickelt hat, Kunden betreut und beraten hat, finden sich heute immer häufiger Softwarekomponente, die in Form von elektronischen Webshops den vollautomatischen Geschäftsverkehr mit klangvollen Namen wie Customer-Relationship-Mangement (=Tools zur Verwaltung riesiger Mengen von Kundendaten) oder Supply-Chain-Management (= automatisierte Zulieferungs-/Lagerverwaltung) bewältigen.

Auch für die Käufer heißt E-Commerce zumeist Gutes. Nämlich größere Preis- und Angebotstransparenz, größere Auswahl und schnellere Kaufabwicklung. Da bringt deutlich bessere Preise, kürzere Wege und geringeren Zeitaufwand mit sich.

Mit der Selbstverständlichkeit einer neuen Technik wächst auch das Bewusstsein für seine Nachteile. Dazu zählt vor allem einer der großen Vorteile dieser Technik. Die Anonymität und grosse Menge potentieller Geschäftspartner – seien es nun Anbieter oder Nachfrager. Aus betriebswirtschaftlicher Sicht ist es problematisch, wenig über die Kunden zu wissen. Die Unkenntnis, was wer wann normalerweise kauft, bedeutet „elektronische Schaufenster" falsch zu gestalten, Werbung an falscher Stelle zu platzieren und Lager mit unnötigen Mengen der falschen Produkte zu füllen. Dies ist teuer, ineffizient und stellt für alle Beteiligten den Nutzen des elektronischen Handels in Frage.

Die vorschnelle Lösung des Anonymitätsproblems, nämlich die Haltung und Analyse detaillierter Kundendaten - stillte den Wissensdurst der Verkäufer. Nun kannten die Firmen fast wie früher im „Tante-Emma"-Laden das Kaufverhalten ihrer Kunden, konnten besser beraten, bewerben und planen.

Allerdings widersprach dies dem Wunsch der Onlinekunden nach Anonymität, Unbeeinflussbarkeit und Datenschutz.
Mit der Sensibilisierung für Datenschutz und Vertrauenswürdigkeit von Online-Geschäftspartnern mussten neue Verfahren gefunden werden, um zum Nutzen beider Seiten aus anonymisierten Daten wertvolles Wissen zu ziehen.

Neben der Schwierigkeit, nur wenige persönliche Informationen bei der Beratung zu haben, kommt ein weiteres Problem. Im Gegensatz zu kleinen lokalen Geschäften, die sowohl in der Zahl der möglichen Kunden, wie auch ihrer Produktvielfalt begrenzt sind, bieten typische Vertreter des E-Commerce, wie AMAZON und EBAY eine riesige Anzahl von Artikel einer praktisch unbegrenzten Käuferschaft an.

Während früher ein kleiner Buchhändler noch relativ sicher sagen konnte, dass Käufer des Buches „Robinson Crusoe" auch an anderen Abenteuerromanen, wie „Gullivers Reisen" interessiert ist, sind heute sowohl die Interessen, die kulturellen Unterschiede und nicht zuletzt die schiere Menge der angebotenen Waren unüberschaubar geworden.
Wie sicher lässt sich unter diesen Bedingungen noch vorhersagen, dass ein pakistanischer Programmierer neben einem Java-Kompendium noch den Roman „Per Anhalter durch die Galaxis" und die DVD „Herr der Ringe" kauft nur weil in Europa und USA, Millionen von Studenten dieses Käuferverhalten zeigen.

Noch schwieriger wird die Frage der Korelation bei unterschiedlichen Produktarten, gibt es doch z.B. bei Kleidungsstücken, Urlaubsreisen oder Möbelstücken wesentlich weniger klare Kategorisierungen wie bei Literatur, Musik oder Filmen.

2 Collaboratives Filtern

Gerade in solchen Fällen, wenn vorgegebene Produktkategorien und Nutzervorlieben kaum ein Zusammenhang haben oder darüber noch keine Erfahrungswerte vorliegen, müssen Verfahren angewendet werden, die auf Basis bestehender Kundendaten solche Zusammenhänge dynamisch finden.

Eine solches Verfahren stellt das Collaborative Filtern, dar. Ähnlich anderen Verfahren des „Data Mining", werden algorithmisch aus großen, scheinbar unausagekräftigen Datenbeständen implizit enthaltene Rückschlüsse und Regeln extrahiert.
Rückschlüsse z.B., dass überdurchschnittlich viele Kunden die Java-Bücher kaufen, auch den Roman „Per Anhalter durch die Galaxis" kaufen. Und das, obwohl beide Bücher weder von der Art noch vom Thema große Gemeinsamkeiten haben.

Der Name setzt sich aus dem Begriff „Kollaborativ" und „Filtern" zusammen, was gut die Wesensmerkmale des Verfahrens beschreibt. Um möglichst wahre – also wahrscheinliche und allgemeingültige Aussagen zu treffen, werden die Daten aller Benutzer miteinbezogen - eben kollaborativ. Diese „Zusammenarbeit" noch unausagekräftiger Einzelaussagen bei der Erkennung bestimmter Muster funktioniert wie eine Art Filter, der aus der Datenmasse nur solche Aussagen extrahiert die wahrscheinlichkeitstheoretisch tatsächlich überdurchschnittlich oft zutreffen.

In der Praxis, findet Collaboratives Filtern als Produktempfehlungssystem bei Onlineshops Anwendung. Die Basis sind fast immer anonymisierter Kundendaten über bereits abgeschlossene Käufe.
Anonymisiert heißt in diesem Zusammenhang, dass die verwendeten Algorithmen weder sensible persönliche Daten wie Geschlecht, Alter, Wohnort verwenden, noch das aus den Ergebnissen Informationen über einzelne Kunden zurückgerechnet werden können.

Unterscheiden werden in der Arbeit zwei Arten von Collaborativen Filtern – aktives und passives – wobei die vorliegende Arbeit sich mit letzterem beschäftigt.
Beide Verfahren suchen Produkte die Kunden interessieren könnten. Während das passive Verfahren nur die bisherigen getätigten Käufe als Beleg für Interesse verwendet, bezieht die andere Variante die Kunden aktiv mit ein, und lässt sie Produkte bewerten.

3 Unterschiede zu bisherigen Forschungsergebnissen

Unterschiedlichste Forschungs- und Wissenschaftszweige befassen sich mit der Vorhersage von Kundenverhalten.

Psychologen, Verhaltensforscher und Marketingspezialisten versuchen anhand empirischer Studien typische Verhaltensmuster aufzudecken. Meistens mangelt es den Ergebnissen solcher Untersuchen an mathematischer Grundlage um sie in konkrete Softwarealgorithmen umzusetzen.
Weder können solche Erfahrungswerte als allgemeingültig und damit übertragbar gelten, noch scheinen sie als einfaches Modell ausgereift und widerspruchsfrei genug, um wahrscheinlichkeitstheoretischen überprüft und bewiesen werden zu können.

Auch in anderen Gebieten der Informatik befasst man sich mit ähnlichen Fragestellungen. Die Autoren benennen drei Gebiete die sich ebenfalls mit dem Erkennen impliziter Muster beschäftigen, deren Ergebnisse aber für Recommendation Systeme nicht anwendbar sind.
Erwähnt werden *Clustering*, ein Verfahren bei dem nach bestimmten vorgegebenen Metriken Ähnlichkeiten in Datensätzen erkannt werden sollen, *Semantic Indexing*, ein Verfahren des Information Retrieval um semantisch ähnliche Dokumente zu finden, sowie lernende Systeme im weitesten Sinn.

Als einer der Hauptunterschiede zum Collaborativen Filtern, wird die Existenz eines festen Ähnlichkeitsmaßes genannt. Im Gegensatz zu den „tf/idf" beim Information Retrieval und den festen Optimierungskriterien bei lernenden Expertensystemen, existiert dies beim Collaborativen Filtern nicht unbedingt.

Die vorgestellte Arbeit des IBM Almaden Research Centers will diese Lücke mit einem einfachen Modell schließen. Das Fehlen einer festen Metrik, macht es um so nötiger ein eigenes Qualitätsmaß für Produktempfehlungen einzuführen. Das von den Autoren vorgestellte Modell enthält eine solches quantitatives Qualitätsmaß – die *Utility Funktion*.

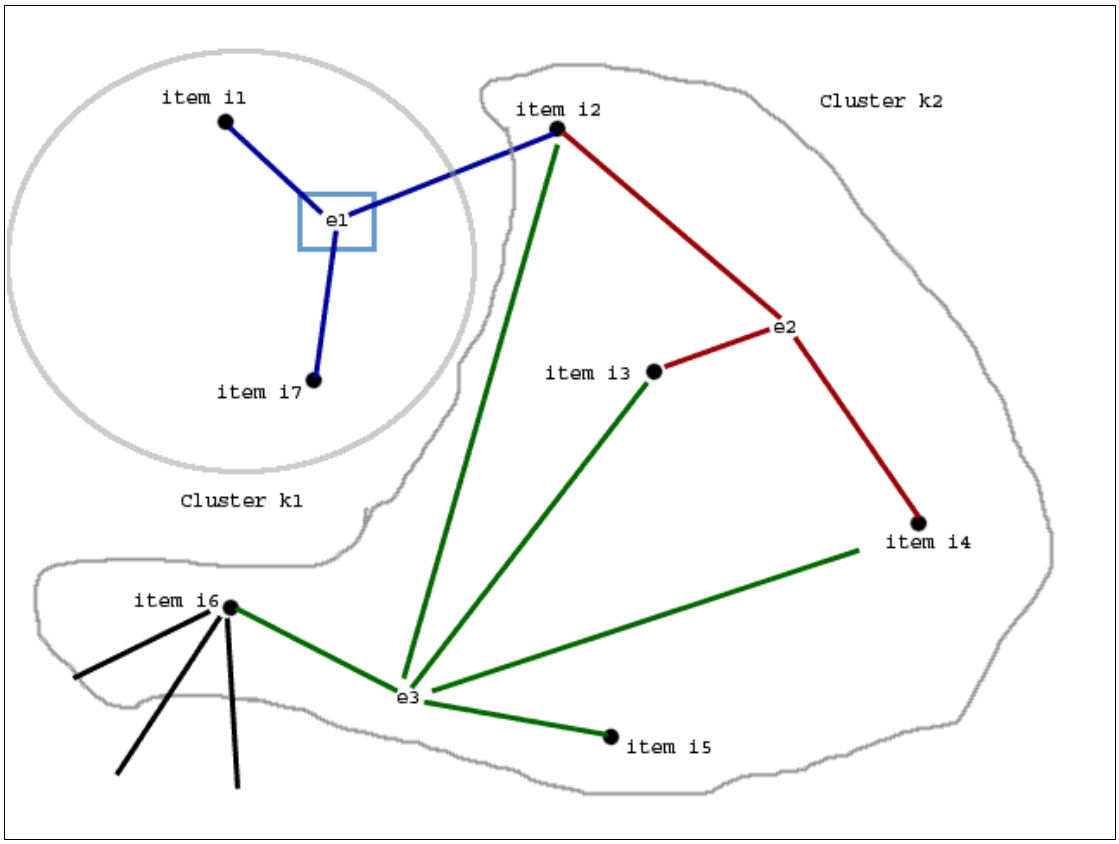
III Modell und Notation

Um die Utility Funktion und die darauf basierenden Algorithmen besser zu verstehen stellt der folgende Abschnitt das von den Autoren aufgestellte Modell und die darin verwendete Notation vor.

1 Darstellung als Graph

Alle vorgestellten Algorithmen arbeiten auf einem Graphen. Dieser repräsentiert das jeweilige Zusammenspiel von Nutzer und ihren Kaufpräferenzen.

Anhand der folgenden Grafik sollen Eigenschaften und die einzelnen Elemente des Graphen erläutert werden.



Angenommen ein Online-Buchladen hat eine Anzahl n Produkten oder allgemeiner **Items $I = \{i_1,\ldots,i_n\}$**. Im Graph erscheinen sie als Knotenpunkte mit den Beschriftungen „item *“.

Im Modell gibt es außerdem eine Anzahl von m **Kunden $E = \{e_1,\ldots,e_m\}$**.

Eine weitere wichtige Größe im Modell sind die so genannten **Samples s(e)**, die eine einzelne Kaufentscheidung eines bestimmten Kunden e repräsentieren. Im hier vorgestellten Modell wird angenommen, dass von jedem Kunden die selbe Anzahl Samples bekannt ist.

Sind nun von einem Kunden mindestens 2 solcher Samples vorhanden, können die beiden jeweiligen Items mit einer Kundenkante verbunden werden. Eine Kante druckt eine Art implizite Zusammengehörigkeit bzw. Ähnlichkeit zwischen den beiden Produkten aus.
Diese ist umso stärker, je mehr Kanten, zwischen Items eingetragen sind. Mehrfachkanten entstehen entweder wenn ein einzelner Kunden dasselbe Item mehrmals kauft oder mehrere Kunden ebenfalls denselben Geschmack zeigten.

2 Produktcluster

Als Definition für für Interesse geht das Modell von folgender Idee aus.
Zwischen Produkten jeder Art besteht eine Art Ähnlichkeit. Hierbei geht es nicht um ein vorher fest definierte Ähnlichkeitsmaß, wie es z.B. in den bekannten Kategorisierungen „Sachbuch" oder „Belletristik" steckt. Vielmehr soll sich die Ähnlichkeit dynamisch aus den bekannten Kundendaten herauskristallisieren. Je mehr Kunden also sowohl Produkt A als auch Produkt B kaufen, umso mehr nimmt das Modell an, dass zwischen A und B eine überdurchschnittliche Ähnlichkeit besteht.

Aus diesen Ähnlichkeitsinformationen kann man dann Produktempfehlungen ableiten. Ein Kunde der bereits A gekauft hat, wird höchstwahrscheinlich auch an B interessiert sein. Warum das so ist, ist dem Modell nicht bekannt. Es geht von rein statistischen Vorhersagen aus.
In der Sprache der Autoren, spricht man von so genannten *Clustern* die man in der Menge der Items sucht. Eine Cluster ist eine Menge von Items, die überdurchschnittlich mehr durch Kanten wie oben beschrieben verknüpft sind. In der obigen Grafik ist eine Cluster als besonders starke Verflechtung von Items im Hypergraph dargestellt.
Diese sind, um das Modell einfach zu halten, als disjunkt und in ihrer Größe ungefähr gleich groß definiert worden. Mathematisch ist ist ein Cluster nichts weiter als eine

Funktion C : [n] --> [k],

die einem Item einem bestimmten Cluster zuweisen.
Eine weitere Vereinfachung ist, dass nicht einzelne Produkte empfohlen werden, sondern nur ganze Cluster. Somit empfiehlt das Modell Kunden die für Cluster 1 eine besondere Präferenz zeigen, alle Produkte aus dem Cluster mit gleicher Wahrscheinlichkeit.

3 Qualitätsfunktion

Um die von einem solchen Recommendation System getroffene Empfehlungen zu bewerten, also eine Aussage zu treffen, wie nah Ihre Empfehlungen an den wirklichen Kundenvorlieben sind, beschreiten die Autoren einen anderen Weg als die bisherigen Arbeiten zum Thema. Während dort die fertigen Algorithmen im nach hinein anhand bekannter Kundenaussagen bewertet wurden, wird hier gleich im Modell der Begriff der Utility, also Nützlichkeit einer Empfehlung aufgenommen.
Die Autoren definieren dies als

Funktion U : [m] x [n] --> [0, 1],

die der Empfehlung eines Item für einen bestimmten Kunden einen Wert zwischen 0 und 1 zuweist. Der Wert ist umso näher an 1, desto mehr er die aus p bekannte Präferenz für dieses Produkt annähert.

Doch wie entscheidet die Funktion, was für den Kunden nützlich ist oder nicht?
Die Antwort ist einfach – anhand seiner bisherigen Einkäufe in Form von Samples in unserem Graph.
Es wird also versucht die wahren Präferenzen des Kunden zu treffen, indem man versucht, die bekannten Präferenzen zu treffen.
Im Modell ist ein Kunde charakterisiert, durch den **Vektor $p(e) = \{p_1,...,p_k\}$**.
Dieser gibt für jeden Cluster die Wahrscheinlichkeit an, mit der der Kunde ein Produkt daraus auswählt.

Am Beispiel unserer Grafik ergeben sich folgende Verteilungen (normiert auf 1):

Kunde e_1: $p(e_1) = \{0.5, 0.5\}$
Kunde e_2: $p(e_2) = \{0.33, 0,66\}$
Kunde e_3: $p(e_3) = \{0.33, 0,66\}$

Demnach wären die Empfehlungen eines Algorithmus dann nützlich, wenn er Kunden e1 mit gleicher Wahrscheinlichkeit Produkte aus Cluster k1 und k2 empfiehlt, und den Kunden e2 und e3 häufiger Produkte aus Cluster k2.
Als allgemeines Ziel des Algorithmus kann dann die Maximierung der Summe $\Sigma_i p_i(e)$ gelten.
Dies wird im folgenden mit

Π(Name des Algorithmus, p)

bezeichnet.

Für die unten aufgeführten Algorithmen benötigen wir die folgenden Daten:

```
//Zuordnung Kunde-Samples, generiert und erweitert bei Kauf
samplesArray[kunde][sample];

//Zuordnung Kunde-Cluster normiert auf SUM_k preference[e][k]=1
preferences[kunde][cluster];

//welcher Cluster ist der beliebteste?
totalClusterPref[cluster];

forall cluster
  forall user
  {
       totalClusterPref[cluster] += preference[user][cluster];
  }

totalClusterPref.sort();
preferedCluster = totalClusterPref[first];
```

4 Zwei Benchmarks – Richtschnur für die Algorithmen

Obwohl die Autoren im Modell von der Existenz von Clustern ausgehen, sind diese später dem Algorithmus nicht bekannt. Er soll sie, ebenso wie die Wahrscheinlichkeitsverteilung für die Kunden, nur annähern können, d.h. sofern sie existieren aus den Daten extrahieren .
Um später beurteilen zu können, wie gut ein solcher approximativer Algorithmus ist, sind Richtwerte nötig die als Gütemaßstab gelten sollen. Die Autoren führen dazu zwei Benchmarks, den starken OPT und den schwachen OPT_w ein. OPT kennt sowohl die Verteilungen p(e) als auch die einzelnen Cluster in Form der Funktion C. Der schwächere OPT_w kennt dagegen nur die Cluster.

OPT stellt mit diesem Wissen eine obere Schranke dar, welches bei

$$\Sigma_{i=1...k} \max\{p_i(e)\}$$

liegt. Das heißt, dass für jeden Kunden e, nur jene Cluster ausgewählt werden dürfen, für den er den höchsten p_i -Wert hat.

IV Näherungsalgorithmen

Im folgenden werden wichtige Zusammenhänge, die zwischen der Qualität eines bestimmten Algorithmus und wichtigen Kenngrößen wie m, s und k aufgeführt. Anschließend folgen erste Annäherungen an den gewünschten Algorithmus. Diese kennen allerdings im Gegensatz zur Zielvorstellung noch die Clusterstruktur.

1 Bedeutung von Samplezahl s, Nutzerzahl m und Clustermenge k

Es gibt zwei Grenzfälle, in denen der „Wissensvorsprung" von OPT zu OPT_w unbedeutend wird. Dies gilt dann, wenn die Anzahl der Nutzer oder aber die Anzahl der Samples pro Nutzer gegen Unendlich geht.

Dies liegt daran, dass bei großem m auch die Anzahl der Mehrfachkanten zwischen Items sehr groß wird und dadurch überdurchschnittliche Häufungen, also Cluster, auffallen. Ähnlich ist es bei sehr vielen Samples pro Kunden. Je mehr Daten man kennt um so kleiner wird der Unterschied zwischen dem bekannten p(e) und den tatsächlichen Präferenzen.

Ein weiterer intuitiver Zusammenhang wird im Artikel bewiesen. Dieser besagt, dass bei gleich bleibender Anzahl Samples pro Kunde, eine steigende Anzahl Cluster zu schlechteren Ergebnissen.

Die Autoren nähern sich dem gewünschten Algorithmus inkrementell, indem sie zuerst vom einfachsten Fall minimaler Sample- und Clusterzahl ausgehen.

Für den Fall von 2 Clustern gibt es folgende drei Algorithmen um eine Empfehlung zu machen: VIC, VOC und VCR. Im folgenden wird angenommen, es gäbe ein stärkeres Interesse der Kunden an Cluster k1.

2 VIC (= Vote In Cluster)

...empfiehlt Kunden – repräsentiert durch ein Kante – den Cluster in deren beide Items sich befinden. Also eventuell auch den Cluster k2, der nach aktueller Datenlage nicht der mehrheitlichen Präferenz entspricht.
Falls sich jedes Item in einem anderen Cluster befindet – bei einer sog. Crossedge – empfiehlt VIC den „stärkeren" Cluster k1.

Pseudocode:

```
cluster VIC(kunde) {

 sample1 = samplesArray[kunde][1];
 sample2 = samplesArray[kunde][2];

 cluster1 = C(sample1);
 cluster2 = C(sample2);

 //cross-edge
 if(cluster1 != cluster2)
      return preferedCluster ;
 else return cluster1;           //=cluster2
}
```

3 VOC (=Vote Out of Cluster)

...empfiehlt Kunden immer k1 und damit folgt im Gegensatz zu VIC der bekannten Wahrscheinlichkeitsverteilung.

Pseudocode:

```
cluster VOC() {
 return preferedCluster ;
}
```

Obwohl dies relativ restriktiv bezüglich von „Outsiderkunden", ist, nutzt dieser Algorithmus besser den kollaborativen Effekt und erzielt einen besseren Wert bei der Qualitätsfunktion.

Dies soll an folgendem Beispiel gezeigt werden. Nehmen wir folgendes an:

1.000.000.000 Kunden präferieren mit p={0.9, 0.1} Cluster k1 und
1000 Kunden mit p={0,1} Cluster k2.
Wir betrachten nur die Kanten die in Cluster 2 liegen. Für diese liegt die erwartete Gesamtnützlichkeit unter Verwendung von VOC bei:

$$\Pi_{voc} = 0.1^2 * 0.9 * 1.000.000.000 = 9.000.000$$

und damit 9x höher als bei VIC mit:

$$\Pi_{vic} = 1000 + 0.1^2 * 0.1 * 1.000.000.000 = 1.001.000$$

4 VRC (=Vote Randomly on Crossedges)

...empfiehlt Kunden zufällig Items aus Cluster k1 oder k2, wenn sie durch eine Crossedge repräsentiert werden. Andernfalls entscheidet VRC wie VOC.

Pseudocode:

```
cluster VRC(kunde) {

  sample1 = samplesArray[kunde][1];
  sample2 = samplesArray[kunde][2];

  cluster1 = C(sample1);
  cluster2 = C(sample2);

  if(cluster1 != cluster2)
       return randomCluster1Or2() ;
  else return cluster1;           //=cluster2
}
```

Interessant ist dieser Algorithmus besonders darum, weil er keine Information mehr darüber benötigt, welcher der beiden Cluster „beliebter" ist.

Die Autoren beweisen in Ihrem Artikel, dass zwischen VRC und OPT das folgende Verhältnis besteht:

$$\Pi(VRC,p) / \Pi(OPT,p) = 2/\sqrt{k} + 1$$

Der von den Autoren angestrebte Algorithmus kann später maximal dieses Verhältnis erreichen.

5 MAX

Um der Frage nachzugehen, wie sich die Menge der Samples pro Kunde auf die Empfehlungsqualität auswirkt, wird ein weiterer Algorithmus eingeführt – MAX.

In diesem Fall wird ein Kunde nun nicht mehr durch nur eine Kante repräsentiert. MAX empfiehlt einen solchen Cluster indem möglichst viele Samples dieses Kunden zu finden sind.
Dabei wird es relevant, wie sehr der Kunde mit seinen Präferenzen p von denen der Masse abweicht. Je geringer die Abweichung, desto besser wird Π (MAX,p) gegenüber Π (VRC,p).

Pseudocode:

```
cluster MAX(kunde) {

// wie viel Samples pro Kunde u. Cluster
samplesInCluster[];

forall cluster
  forall sample
  {
     samplesInCluster[C(samplesArray[kunde][sample])]++;
  }

sample1 = samplesArray[kunde][1];
sample2 = samplesArray[kunde][2];

cluster1 = C(sample1);
cluster2 = C(sample2);

if(samplesInCluster[cluster1]>samplesInCluster[cluster2])
     return cluster1;
elseif(samplesInCluster[cluster1]<samplesInCluster[cluster2])
     return cluster2;
else
     return randomCluster1Or2();
}
```

V Algorithmen bei unbekanntem p(e)

Im letzten Teil stellen die Autoren zwei Algorithmen – NEIGHBOR und VOTING vor, die auch ohne Kenntnis der Nutzpräferenzverteilung p ähnlich gute Empfehlungen abgeben.

Sie zeigen, dass unter der bisherigen Annahme von 2 Clustern und 2 Samples pro Kunde, zwischen dem erwarteten Gesamtnutzen von NEIGHBOR und OPT folgendes Verhältnis besteht:

Π (NEIGHBOR, p) >= 0.7 * Π (OPT,p)

Dies kann durch Verwendung von VOTING sogar noch verbessert werden. Da beide Algorithmen keine Clusterinformation mehr haben, werden Items emfohlen.

1 NEIGHBOR

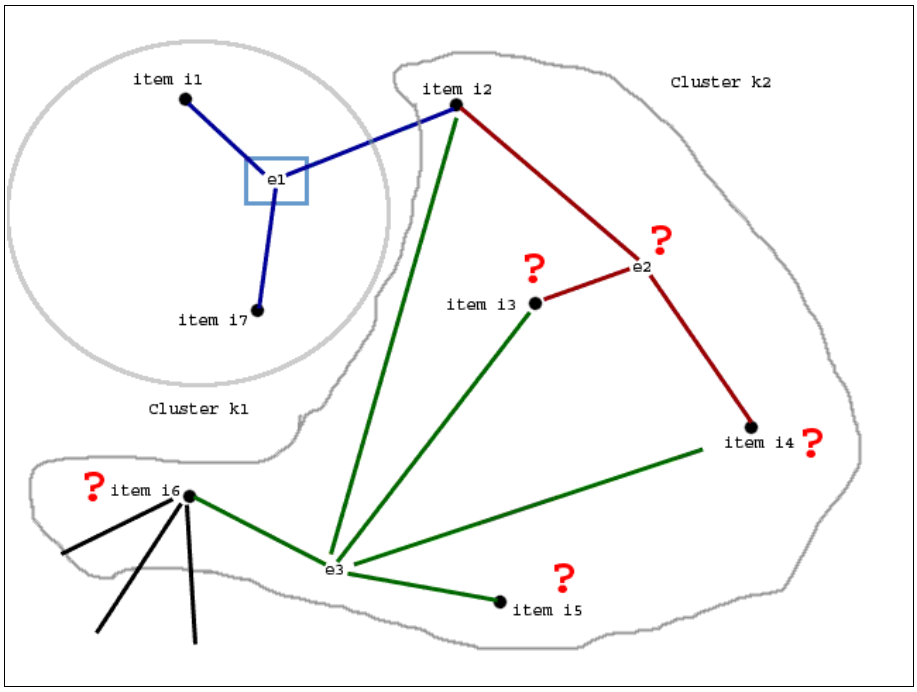
...empfiehlt einem Kunden – repräsentiert durch eine ihrer Kante – zufällig eines der benachbarten Items - in der Grafik kommen für e1 alle „?“-Kanten in Frage.

Pseudocode:

```
item NEIGHBOR(kunde) {

 // Kanten des Graphs repräsentiert durch Nicht-0 in Matrix
 graph[kunde][knoten1][knoten2];

 //angrenzende Kanten bzw. Items finden
 forall andererKunde != kunde {
   forall item != knoten1 {
      if(graph[andererKunde][knoten2][item] != 0)
        return item;
      else if(graph[andererKunde][item][knoten2] != 0 )
        return item
   }
   forall item != knoten2 {
      if(graph[andererKunde][knoten1][item] != 0)
        return item;
      else if(graph[andererKunde][item][knoten1] != 0 )
        return item;
   }
  }
}
```



Dieser Algorithmus hat eine erwartete Gesamtnützlichkeit von mindestens **0.7 * Π (OPT,p)**.

2 VOTING

...empfiehlt immer die benachbarten Items, welche durch eine größere Anzahl Kanten mit der aktuellen Kundenkante verbunden sind - in der Grafik für e1, die „!“-Kante.

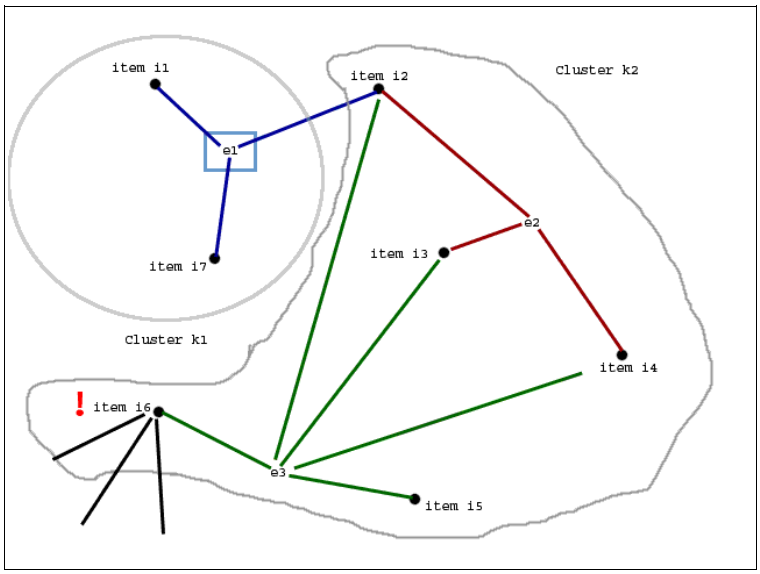
Pseudocode:

```
item VOTING(kunde) {

 // Kanten des Graphs repräsentiert durch Nicht-0 in Matrix
 graph[kunde][knoten1][knoten2];

 multiplicity[item];  // wie oft ist Item in Kante

 //angrenzende Kanten bzw. Items finden
 forall andererKunde != kunde {
   forall item != knoten1 {
      if(graph[andererKunde][knoten2][item] != 0)
        multiplicity[item]++;
      else if(graph[andererKunde [item][knoten2] != 0 )
        multiplicity[item]++;
   }
   forall item != knoten2 {
      if(graph[andererKunde]knoten1][item] != 0)
        multiplicity[item]++;
      else if(graph[andererKunde [item][knoten1] != 0 )
        multiplicity[item]++;
   }
 }
 multiplicity.sort();
 return multiplicity[first];     //den besten rausholen
```




```
}
```

Erwartungsgemäß verbessert sich die Aussagekraft von Empfehlungen hier mit steigender Kundenzahl. Die Autoren beweisen, dass für m-> ∞, VOTING so gut wird wie OPT_W.

VI Erweiterbarkeit und Fazit

Das vorgestellte Modell ist in seiner vorgestellten Form nur recht eingeschränkt einsetzbar, da unrealistische Vereinfachungen vorgenommen wurden. Diese werden im folgenden kurz genannt.

1. ***Cluster werden als homogen betrachtet.***
   Wie bei Kategorisierungen wäre es auch hier sinnvoll unterschiedliche Stufen und Granularitäten zu berücksichtigen

2. ***Clusterwahl statt Itemwahl.***
   Im allgemeinen kann nicht davon ausgegangen werden, dass alle Items in einem Cluster auf dasselbe Kundeninteresse stoßen.

3. ***Globales Optimierungsziel***
   Anstatt den Algorithmus auf seine Gesamtnützlichkeit zu maximieren, wäre es auch möglich die minimale Nützlichkeit pro Kunde zu maximieren.

4. ***Kunden werden als gleich wichtig betrachtet.***
   Es wäre realistischer, die Samples der Kunden stärker zu gewichten, die häufiger und in größere Mengen einkaufen.

5. ***Einseitige Nützlichkeitsfunktion***
   Besser wäre die Hinzunahme eines negativen Wertebereichs für die Darstellung von Abneigung gegenüber Produkten.

Die Autoren heben hervor, dass durch den modularen Aufbau, also die Trennung wichtiger Aspekte des Problems in unabhängige aber kompatible Teile, eine spätere Erweiterung kein Problem darstellt.
Trotz oder gerade wegen dieser Vereinfachung ist das Modell allgemeingültig und leicht verständlich. Außerdem handelt es sich bei der Arbeit um das erste vollständige und vor allem mathematisch fundierte Modell für Collaboratives Filtern. Somit sind die vorliegenden Forschungsergebnisse eher als solides Fundament für ein noch junges Gebiet der Informatik zu sehen. Mit dieser Vorarbeit wird es in Zukunft leichter sein, hochwertige Recommendation Systeme zu entwickeln.

[Fragen und Anmerkungen zur Ausarbeitung an robert.soesemann@student.uni-tuebingen.de]