Vortrag im Seminar:

"Lernende Agenten im WWW"

Wintersemester 1998 / 99

Universität Dortmund, FB Informatik, LS Künstliche Intelligenz, Katharina Morik

Adaptive Web Sites:
Automatische Erzeugung von Web Seiten

(Mike Perkowitz, Oren Etzioni)

Frank Wächter [waechter@icd.de]

Datum des Vortrags: 04.12.1998

Version 1.01

Literatur: Perkowitz, M. and Etzioni, O. (1998) Adaptive Web Sites: Automatically Synthessizing Web Pages; AAAI 1998

Gliederung

- Motivation
- Möglichkeiten und Ziele
- Praxisbeispiel
- Das Index-Seiten Synthese Problem
- Die Besuchs-Zusammenhangs-Annahme
- Das Cluster Mining Verfahren
- Der PageGather Algorithmus
- Experimentelle Validierung
- Zukünftige Verbesserungen

Motivation

- Web Sites sollen ihre Informationen auch preisgeben
- **Probleme:**

Verschiedene Besucher	↔	Andere Ziele
Gleiche Besucher, andere Zeiten	↔	Andere Ziele
Angedachter Gebrauch	↔	Reale Benutzung

- Web Sites lernen vom **<u>Benutzerverhalten</u>** und passen ihre Organisation und Präsentation selbst an

Anpassungsmöglichkeiten

Maßschneiderung:

- Angepaßt an die Bedürfnisse jedes **Einzelnen**
- Viele Versionen der gleichen Site müssen vorhanden sein
- Sammlung von **Benutzerdaten** $\leftrightarrow$ Privatssphäre

Optimierung:

- Lernen durch Interaktion mit allen Benutzern
- Verbesserung für **alle Besucher** und für zukünftige Besucher

Verfolgte Ziele

- Benutzer sollen **keine Fragen** beantworten müssen
- **Alle** sollen die Site einfacher benutzen können
- Auf **keinerlei Meta-Informationen** aufbauen, die noch nicht verbreitet sind, z.B. XML
- Umformungen bei gleichzeitigem **Strukturerhalt** der Site:
  - Links zu Seiten der Site hinzufügen
  - Bestehende Links hervorheben
  - Promotion von Links auf der Titel-Seite
  - **Erstellung von neuen Index-Seiten mit passenden Links zu einzelnen Themen**

Praxisbeispiel - Web Site

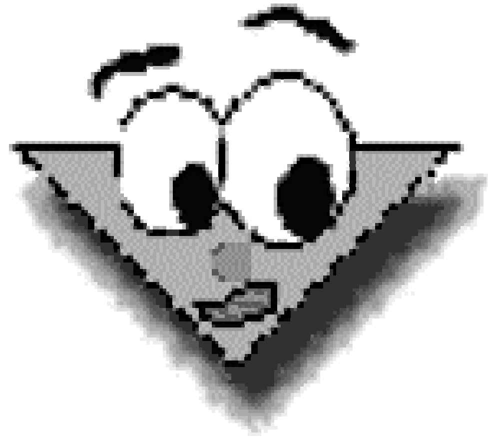
Music Machines

http://www.hyperreal.org/music/machines

- Site für elektronische Musikinstrumente aller Art,
  mit Bildern, Tests, Handbücher und Beispiel-Samples
- Gruppiert nach Herstellern
- 2.500 verschiedene Seiten; 1200 Besucher;
  10.000 Zugriffe / Tag

Praxisbeispiel - Agent "Winky"

- Enthält einen Agenten: **"Winky"**



- Winky listet auf jeder Seite eine kleine Sammlung von **"passenden" Links** zu automatisch erstellten Index-Seiten auf

- Winky kann man auch **Feedback** geben und ihn somit verbessern
  - "Ich fand, was ich suchte"
  - "Ich fand es nicht"
  - "Ich surfte nur so herum"

Praxisbeispiel - Fallbeispiel

- Viele Benutzer interessieren sich für Keyboards
- Viele klicken nacheinander mehrere Hersteller von Keyboards an und hören sich die Beispiel-Samples an
- Benutzerinteraktionen werden in **Access-Log** Datei festgehalten:

	IP-Adresse	URL	Zeit
▶	111.11.11.11	.../yamaha/dx7/samples/dx7.wav	00:00:00
	222.22.22.22	.../x/xx/xxx.htm	00:00:00
▶	111.11.11.11	.../roland/jx/samples/jx.wav	00:05:00
▶	111.11.11.11	.../korg/kr55/samples/kr55.wav	00:10:00
	333.33.33.33	.../y/yy/yyy/yyy.zip	00:10:00

Praxisbeispiel - Screenshot

- home
- gear
- links
- search
- guide
- email

Music Machines

Winky suggests:

- /categories/drum-machines/samples/
- /manufacturers/ARP/Odyssey/samples/
- /manufacturers/Moog/MiniMoog/samples/
- /Analogue-Heaven/
- /manufacturers/Roland/TR-909/samples/

Help MM Improve! Choose one: [I found what I wanted] [Didn't find it] [Just browsing]

samples

ARP Odyssey	Moog MiniMoog	Roland SH-101
Akai MPC	Moog Rogue	Roland SH-synths
Boss DR-110	Music-and-More VF11	Roland System-100-100m
Boss DR-55	Paia Fatman	Roland TR-606
Casio CZ	Paia Vocoder	Roland TR-808
Casio SK	Rhodes Polaris	Roland TR-909
Kawai SX-synths	Roland CompuRhythm	Sequential Tom
Korg KPR-77	Roland JX	Siel DK-600
Korg KR-55	Roland Juno	Univox Micro-Rhythmer-12
Linn LinnDrum	Roland MC-303	Yamaha TX-81z

Das Index-Seiten Synthese Problem

- **Gegeben:** Web Site und Access-Log
- **Gesucht:** Index-Seiten mit Menge von Links zu **<u>verwandten</u>** aber noch **<u>nicht verbundenen</u>** Seiten

1. Welche Links kommen auf die Index-Seite
2. Wie werden die Links sortiert
3. Welchen Titel hat die Index-Seite
4. Welche Beschriftung haben die Links
5. Paßt das Layout zum Rest der Site
6. Wo kann man die Index-Seite aufrufen

Die Besuch-Zusammenhangs-Annahme

- Das einzig verwendbare Mittel ist die Access-Log Datei
- Ein **<u>Besuch</u>** ist die geordnete Sequenz von Seiten, welche ein Besucher in einer Sitzung besucht hat
- **Annahme:** **Die Seiten, die ein Benutzer während einer Sitzung besucht, tendieren dazu, konzeptionell verwandt zu sein**
- **Probleme:**
  - Besucher verfolgt vielleicht pro Sitzung mehrere Ziele
  - Besucher kann nicht immer durch die IP-Adresse identifiziert werden (Proxy-Server)
- Viele Daten sammeln und nach globalen Trends suchen

Clustering

- Bei der Gruppierung (**<u>Clustering</u>**) von Dokumenten werden diese in einem N-dimensionalen Raum repräsentiert (N = Anzahl der Medikamente, z.B. Wörter eines Vokabulars); jedes Dokument wird als Vektor repräsentiert
- Die **<u>Ähnlichkeit</u>** zwischen Dokumenten wird als Cosinus des Winkels zwischen den Vektoren definiert
- Eine Gruppe (**<u>Cluster</u>**) ist eine Kollektion von Dokumenten, die eine hohe Ähnlichkeit zueinander besitzen, aber eine relativ geringe Ähnlichkeit zu Dokumenten aus anderen Gruppen vorweisen

Cluster Mining (Gruppen-Gewinnung)

- Traditionelle Clustering-Verfahren partitionieren die Menge der Dokumente in disjunkte Gruppen, jedes Dokument paßt in genau eine Gruppe

- **<u>Cluster Mining</u>** sucht nach kleiner Zahl von sich überlappenden Gruppen, Dokumente können verschiedenen Gruppen angehören

Traditionelle Verfahren Klassifikation	⟷	**Cluster Mining** Regeln für Struktur

Der PageGather Algorithmus

1. Wandle Access-Log in Besuche um
2. Errechne **Häufigkeiten** des "gemeinsamen Vorkommens" (co-occurence frequency) und erzeuge hieraus eine Ähnlichkeits-Matrix
3. Erzeuge aus der Matrix einen Graphen, finde **Cliquen** (als schnellere Alternative: suche nur nach **Zusammenhangskomponenten**)
4. Erzeuge aus jeder gefundenen Gruppen eine Index-Seite mit Links zu Elementen der Gruppe

PageGather - Begriffsklärung

Clique:

Teilgraph, in dem jedes Paar von Knoten durch eine **direkte Kante** miteinander verbunden ist

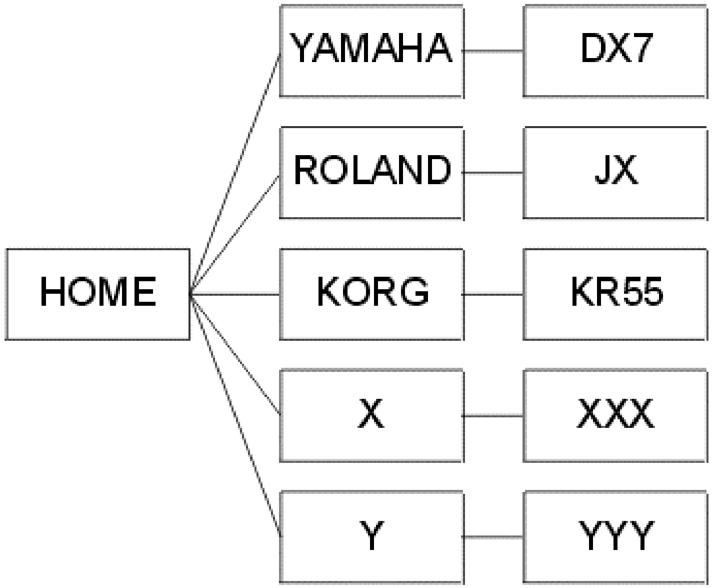
➔ Cluster, in dem jeder Paar von Seiten häufig gemeinsam besucht wird

Zusammenhangskomponente:

Teilgraph, in dem jeder Paar von Knoten durch einen **Pfad** von Kanten miteinander verbunden ist

➔ Cluster, in dem jede Seite ähnlich zu mindestens einer weiteren Seite im Cluster ist

Berechnung der Häufigkeiten und Konstruktion der Ähnlichkeitsmatrix



DX7	0				
JX	8	0			
KR55	9	9	0		
XXX	4	3	2	0	
YYY	2	1	4	8	0
	DX7	**JX**	**KR55**	**XXX**	**YYY**

PageGather - Berechnung Schritt 2

- Errechne für jedes Paar Seiten S_1 und S_2:

  ***Pr*$_{1,2}$ ($S_1|S_2$)**, die Wahrscheinlichkeit, daß der Besucher S_1 besucht, wenn er S_2 besucht hat und ***Pr*$_{2,1}$ ($S_2|S_1$)**

- Nimm **<u>Minimum</u>** von $Pr_{1,2}$ und $Pr_{2,1}$, um Fehler zu minimieren

  Beispiel: S_1 ist Übersichtsseite und Start für viele Suchen;

  S_2 ist spezielle Seite

  ➔ $Pr_{1,2}$ hoch, aber $Pr_{2,1}$ gering

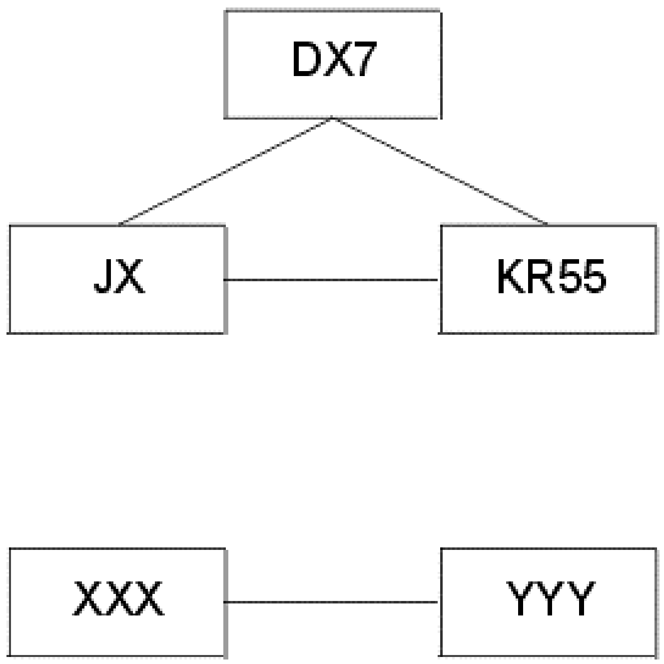
- Alle schon miteinander verbundenen Seiten bekommen Eintrag Null, denn nur noch nicht verbundene Seiten werden gesucht

PageGather - Berechnung Schritt 3

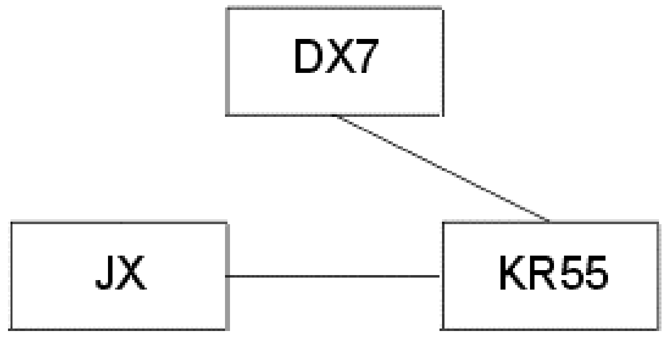
- Erstelle Graphen aus Ähnlichkeitsmatrix
- Wende **Schwellwert** an und entferne alle Kanten, die unterhalb des Schwellwertes liegen
  - ➔ Graph wird kleiner und schmaler für effizientere Berechnung
- Finde **Cliquen**, maximal einer konstanten Größe (sonst NP-vollständig) bzw. **Zusammenhangskomponenten**
- Zusammenhangskomponenten sind schneller zu berechnen, aber Cliquen bilden genauere Gruppen von Seiten, da alle paarweise miteinander häufig aufgerufen werden

PageGather - Beispiel Schritt 3

Erstellung des Graphen und Finden der Cliquen bzw. Zusammenhangskomponenten



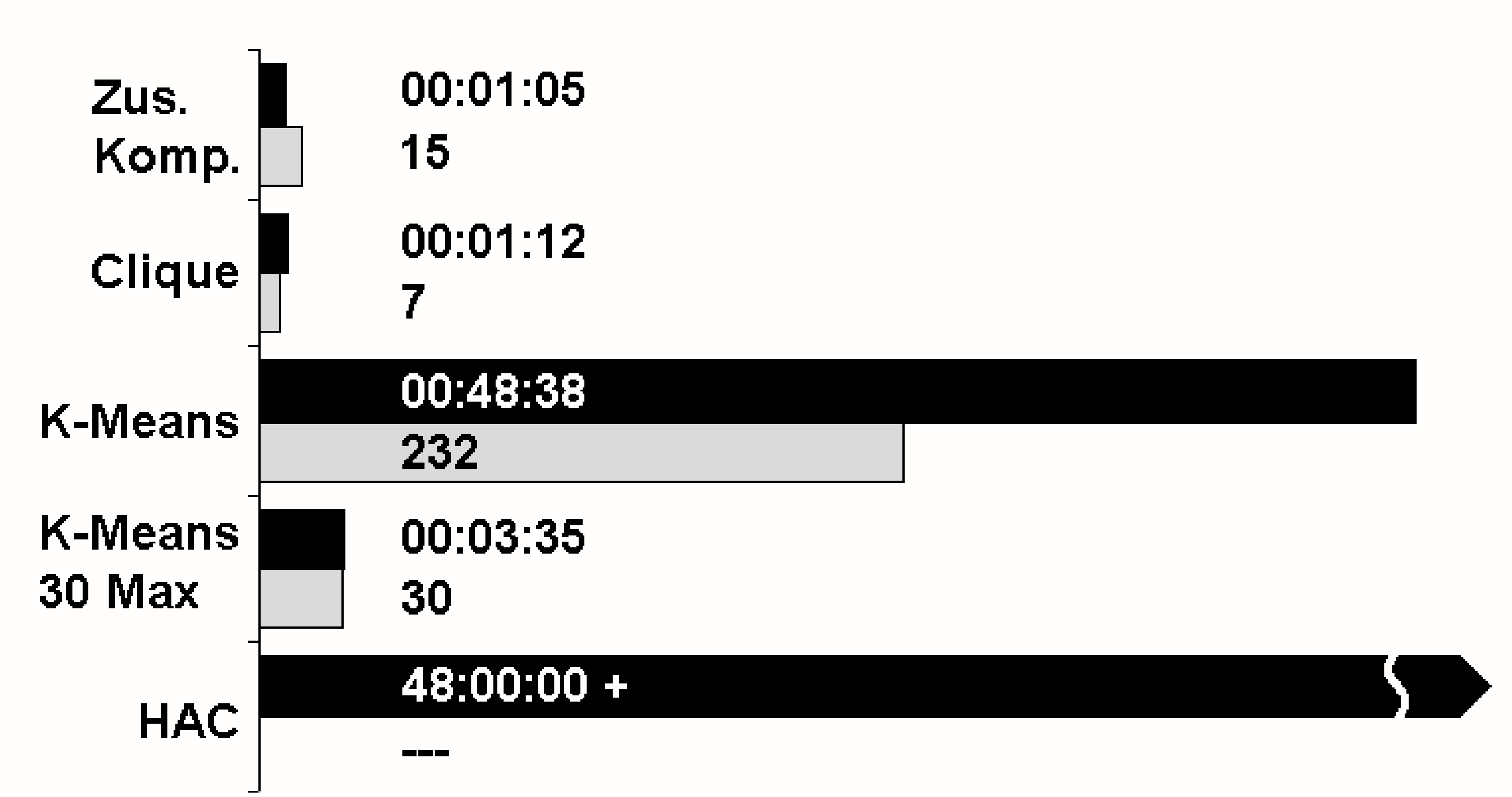
Schwellwert = 8



Schwellwert = 9

Experimentelle Validierung - Laufzeit

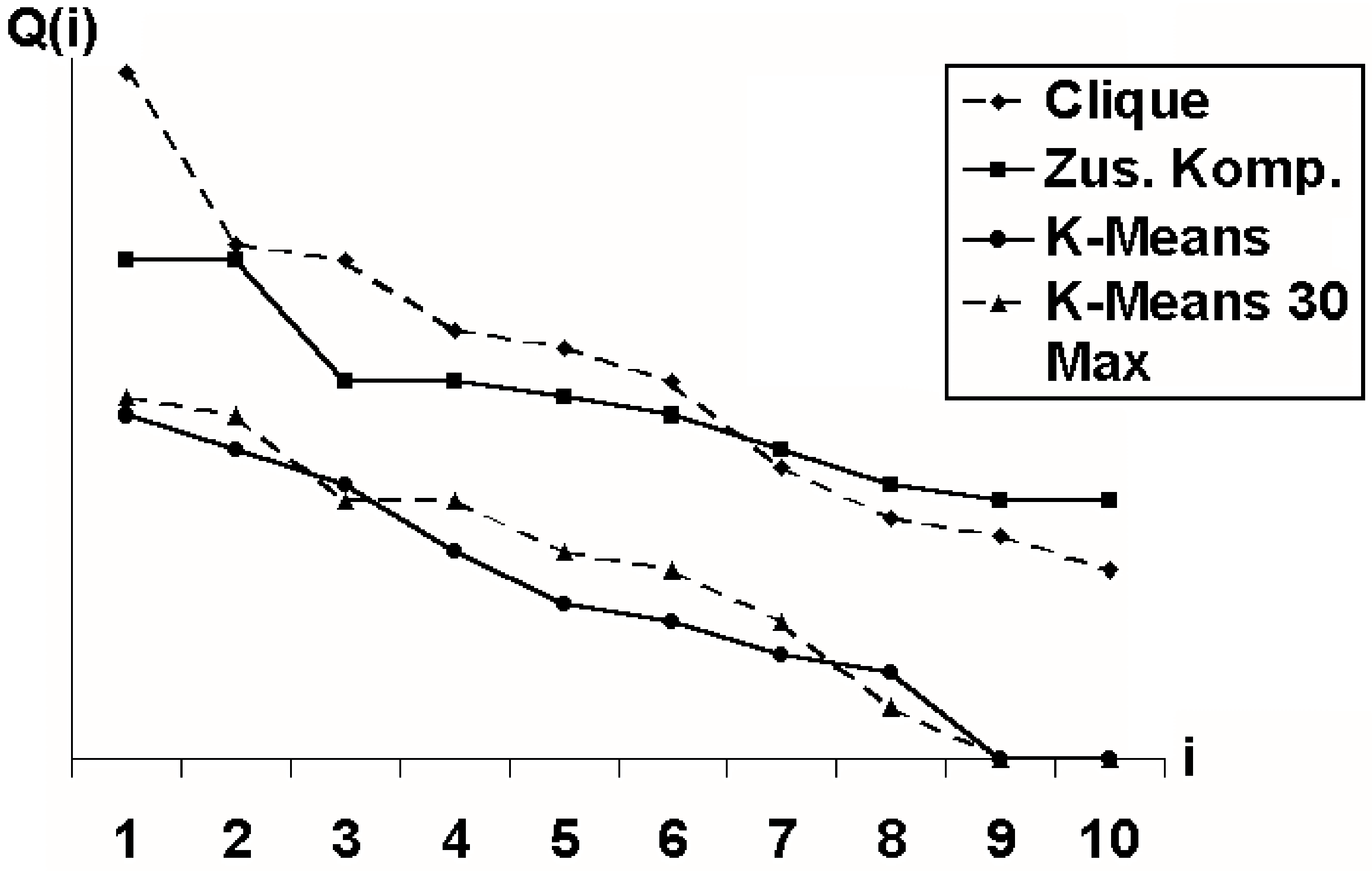
- Zwei Vergleichsalgorithmen: **HAC** und **K-Means**
- Alle Algorithmen sollen 20 Gruppen bilden



Experimentelle Validierung - Qualität

- Wie mißt man die Qualität der gefundenen Gruppen?
- **Maß:** Wenn ein Besucher eine Seite der Gruppe besucht, wie wahrscheinlich ist es, daß dieser noch andere Seiten dieser Gruppe besucht?
- **Problem:** Je größer die Gruppe ist, desto höher ist die Wahrscheinlichkeit, daß noch andere Seiten der Gruppe innerhalb eines Besuches besucht wurden
- Ermittle für die 10 besten Gruppen jedes Algorithmus Qualität

Experimentelle Validierung - Qualität (2)



- Trotz größerer Gruppen bei K-Means ist PageGather besser (und dabei noch schneller)

Experimentelle Validierung - Ergebnisse

- **Beispiele:**
  - Fast alle Audio-Samples
  - Keyboards verschiedener Hersteller
  - Software zum Download
- Andere Beispiele faßten eher einen einzelnen Abschnitt der Web Site zusammen
- **Selbst wenn der Algorithmus nur 2 gute Ideen aus 10 Vorschlägen produziert, können sie für den Webmaster trotzdem nützlich sein**

Zukünftige Verbesserungen

- Algorithmus kann zwar viele verwandte Seiten gruppieren, ob aber **alle Seiten** gefunden werden, ist fraglich
  ➔ Gruppen als Trainingsmenge zum Lernen nutzen, um Regeln für Gruppenzugehörigkeit zu finden
- **Irrtümliche Gruppierung** verschiedenartiger Seiten, gewollte Hierarchie der Site wird abgeflacht
  ➔ Durch Ausnutzung von Meta-Informationen (z.B. Keywords) werden homogenere Index-Seiten erzeugt