INGENIERÍA INDUSTRIAL
UNIVERSIDAD DE CHILE

fcfm

FACULTAD DE CIENCIAS
FÍSICAS Y MATEMÁTICAS
UNIVERSIDAD DE CHILE

INSTITUTO MILENIO
SISTEMAS COMPLEJOS DE INGENIERÍA

Chapter 2

Web Data

PROFESSOR

Juan D. Velásquez

Outline

- Web's Operation
- The Information Behind the Clicks
- Session reconstruction process
  - Traditional Approach
  - New Approaches
    - Web User Session Reconstruction Using Integer Programming
    - Fast combinatorial algorithm for Web user session reconstruction
- Finding real sessions
- The information contained in a web page
- Web page content
- Web page links

Section 2.1

》》 Web's Operation

The web process
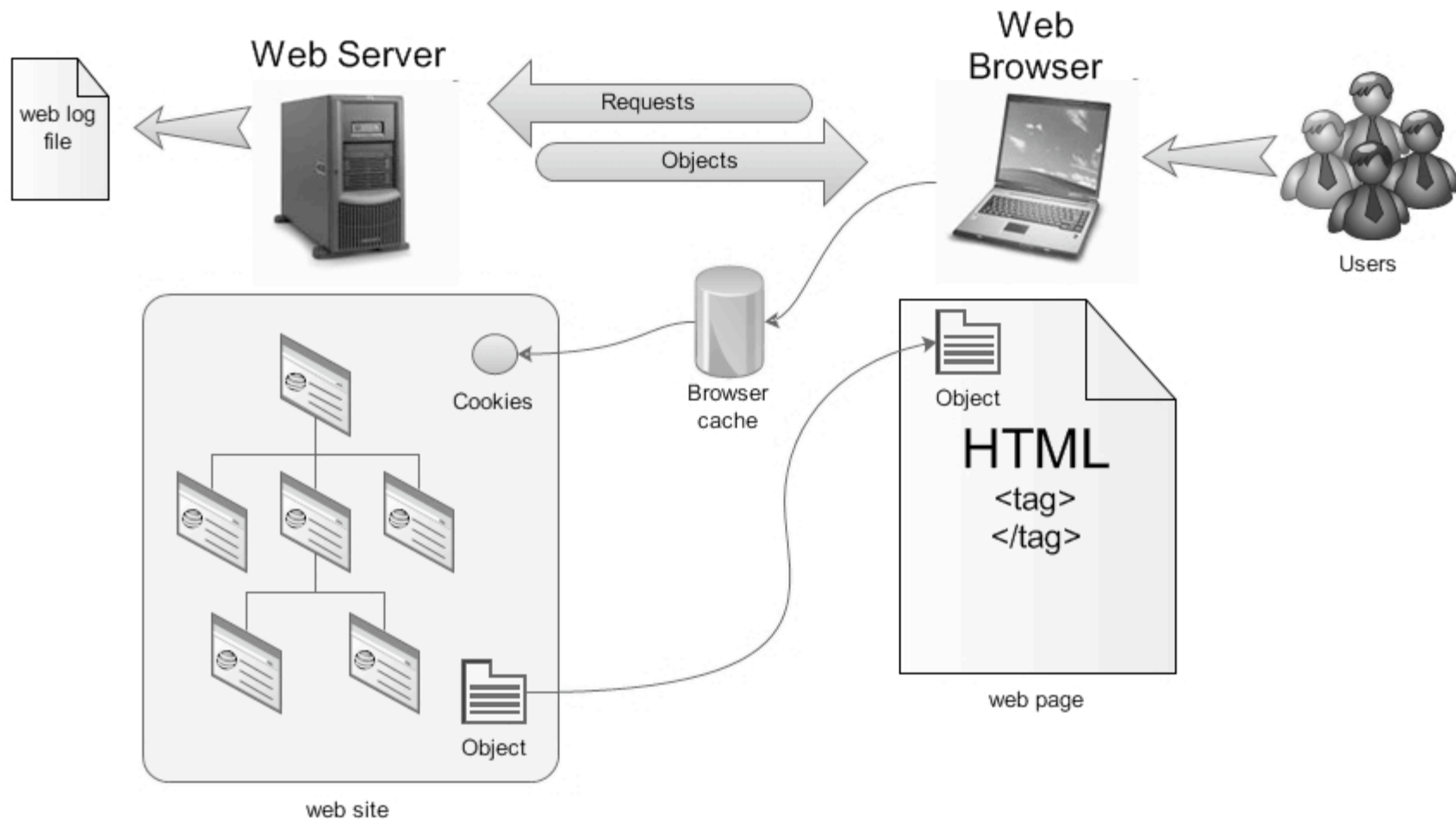
Web server and web browser

- Once the document has been **read** by the browser, the specific tags inside are **interpreted**.
- **When the browser interpreting the tags** find a reference about an object, for instance an image, the HTTP **gets it and transfers** it to the browser.

The process finishes when the last tag is interpreted and the page is shown to the visitor

The web process

- ‣ The transport operation use the **HTTP protocol.**
- ‣ Web pages are written in the **HTML language.**
- ‣ A web page contains tags that reference other object to ask to the server or to be download to the user browser.
- ‣ **Content** are usually more complex than they appears:
  - ‣ Applets
  - ‣ Javascripts
  - ‣ Dynamics HTML
  - ‣ Flash

Web server and web browser (2)

- The **web log** registers contain information about the **visitor browsing behaviour**, in particular, the **page navigation sequence** and the **time spent in each page visited**.
- When a web page is accessed, the HTML code, with **web page tags** referring to various web objects, is **interpreted in the browser**.
- **A register is created for the accessed page** as well as for each object referred in the page.
- Depending on the web activity, these logs can **contain millions of registers** and **most of them may not hold relevant information**.

The web Server

- A Server is a program not a machine.

- They usually serve **not only plain web pages** , they also serve **web applications with HTML front-end.**

- Web application architecture: **Multiple layer Model.**
  - **Interface Layer**: Perform html rendering.
  - **Logic Layer**: core business application.
  - **Data Layer**: Storage/Retrieval data process.

- Web server also maintain data logs about **user action** in the web: Some site could have of the order of **Gb/day of logs.**

The web Client

- The browser perform all the **required management of the connection with the server**.

- There are also allowed **to connect to proxy server** that are **cache server of statics pages**.

- Also **execute local program** (client side application)

- Maintain a repository of clientside data called **"Cookies",** that allows to web application:

  retrieve particular information about a particular client.

  maintain a session ID with the client.

- **These cookies also are present in the server** in order to identify the correct client.

Section 2.2

>> The Information behind the clicks

Understanding the visitor behavior in a web site

- Visitor browsing behaviour
  - Web logs.
- Visitor preferences
  - Web pages
- Problems:
  - Web logs contain **a lot of irrelevant data.**
  - A Web site is a huge **collection of heterogeneous, unlabelled, distributed, time variant, semi-structured and high dimensional data.**

The dream

"Transform the visitors into customers and retain the existing ones"

- Some solutions:
  - Continuous improvement of the web site structure and content.
  - Personalization of the relationship between the user and the web site.
  - Understanding the user behaviour in the web site.

The Server log File

- Usually this repository was used to perform **web server tuning** and other **system administration task**.

- But they acquire some **unexpected VALUE** to the **marketing researcher**.

- ¡THEY CONTAIN IN A IMPLICIT WAY ALL THE CLIENT BEHAVIOUR:

  - WE DON'T NEED A SURVEY, WE ALREADY HAVE THE INFORMATION!

- The Log file write **a line** with this precious information **for each request of a client browser**.

The Server Log File: an extract

#	IP	Id	Acces	Time	Method/URL/Protocol	Status	Bytes	Referer	Agent
1	165.182.168.101	-	-	16/06/2002:16:24:06	GET p1.htm HTTP/1.1	200	3821	out.htm	Mozilla/4.0 (MSIE 5.5; Wi
2	165.182.168.101	-	-	16/06/2002:16:24:10	GET A.gif HTTP/1.1	200	3766	p1.htm	Mozilla/4.0 (MSIE 5.5; Wi
3	165.182.168.101	-	-	16/06/2002:16:24:57	GET B.gif HTTP/1.1	200	2878	p1.htm	Mozilla/4.0 (MSIE 5.5; Wi
4	204.231.180.195	-	-	16/06/2002:16:32:06	GET p3.htm HTTP/1.1	304	0	-	Mozilla/4.0 (MSIE 6.0; Wi
5	204.231.180.195	-	-	16/06/2002:16:32:20	GET C.gif HTTP/1.1	304	0	-	Mozilla/4.0 (MSIE 6.0; Wi
6	204.231.180.195	-	-	16/06/2002:16:34:10	GET p1.htm HTTP/1.1	200	3821	p3.htm	Mozilla/4.0 (MSIE 6.0; Wi
7	204.231.180.195	-	-	16/06/2002:16:34:31	GET A.gif HTTP/1.1	200	3766	p1.htm	Mozilla/4.0 (MSIE 6.0; Wi
8	204.231.180.195	-	-	16/06/2002:16:34:53	GET B.gif HTTP/1.1	200	2878	p1.htm	Mozilla/4.0 (MSIE 6.0; Wi
9	204.231.180.195	-	-	16/06/2002:16:38:40	GET p2.htm HTTP/1.1	200	2960	p1.htm	Mozilla/4.0 (MSIE 6.0; Wi
10	165.182.168.101	-	-	16/06/2002:16:39:02	GET p1.htm HTTP/1.1	200	3821	out.htm	Mozilla/4.0 (MSIE 5.01; W
11	165.182.168.101	-	-	16/06/2002:16:39:15	GET A.gif HTTP/1.1	200	3766	p1.htm	Mozilla/4.0 (MSIE 5.01; W
12	165.182.168.101	-	-	16/06/2002:16:39:45	GET B.gif HTTP/1.1	200	2878	p1.htm	Mozilla/4.0 (MSIE 5.01; W
13	165.182.168.101	-	-	16/06/2002:16:39:58	GET p2.htm HTTP/1.1	200	2960	p1.htm	Mozilla/4.0 (MSIE 5.01; W
14	165.182.168.101	-	-	16/06/2002:16:42:03	GET p3.htm HTTP/1.1	200	4036	p2.htm	Mozilla/4.0 (MSIE 5.01; W
15	165.182.168.101	-	-	16/06/2002:16:42:07	GET p2.htm HTTP/1.1	200	2960	p1.htm	Mozilla/4.0 (MSIE 5.5; Wi
16	165.182.168.101			16/06/2002:16:42:08	GET C.gif HTTP/1.1	200	3423	p2.htm	Mozilla/4.0 (MSIE 5.01; W

The Server log File: Structure

- **IP Address**: Client IP.
- **Identity**.
- **Authuser**: Used when SSL is activated.
- **Time**: data and time of the request
- **Request**: The object requested by the browser.
- **Status**: Integer code of the status of the request.
- **Bytes**: The number of bytes returned.
- **Referrer**: text send by the client indicating the original source of a request.
- **User-Agent**: Name and version of the web browser used.

Session reconstruction: the need

- If we want to understand the user behaviour in a web site, web need to know his/her **real browsing behaviour.**

- The **quality of patterns** extracted by using a mining technique **depend on the input data.**

- Elements like **proxies servers**, **dynamic IP**, **missing references** and the **inability of servers to identify different users** make **difficult to reconstruct a real session.**

The Web Logs: some problems

- **The web log doesn't store the client id.**
- **Proxy and Firewall**: The IP are masked, then the IP number couldn't identify uniquely a client.
- **Web A synchronism**: Several user access simultaneously the server. Identification method like cookies or session reconstruction techniques are needed.
- **Web Crawlers or Spider Robots**: Google or Yahoo! use a automatic program that retrieve periodically each page. They have to be identified and eliminated.
  - http://www.robotstxt.org/wc/robots.html
- **Cache:** Sometime the browser use a web cache or a proxy cache that imply that the behaviour was not stored on the logs.

Section 2.3

≫ Session Reconstruction Process – A traditional approach

Session Reconstruction Process

- We want to identify the lines in the logs file that belong to a unique valid client.
- This process is called "**Sesionization**"
- **Usual assumption**:
  - each session has a maximum time duration.
- Strategies:
  - **Proactive strategies**: Identify users methods likes cookies. Privacy problem.
  - **Reactive strategies**: Non invasive privacy.
    - **Navigational Oriented Heuristics**: pages visited follows the hyperlink structure. If a page doesn't follows this order is a new session.
    - **Time Oriented Heuristics**: Using usually **30 min for maximum session time.**

Session Reconstruction Process

1. **Filtering**: Select **only the relevant log register**, relevant to web pages. Eliminating request for pictures, videos or errors code.

2. **Grouping**: **IP** and **Agents** can be a good selector of client sessions.

3. **Discriminating sessions by time stamps**: Register are selected by **time windows of 30 minutes.**

4. **Identifying irregular session**: **robot or spider** that could be detected looking at the Agent field.

5. **Real session conditioning**

RDBMS could help with the indexing to the efficiency of the calculations.

Sesionization process

IP	Agent
165.182.168.101	MSIE 5.01
165.182.168.101	MSIE 5.01
165.182.168.101	MSIE 5.01
165.182.168.101	MSIE 5.5
165.182.168.101	MSIE 5.5
165.182.168.101	MSIE 5.5
165.182.168.101	MSIE 5.5
204.231.180.195	MSIE 6.0
204.231.180.195	MSIE 6.0
204.231.180.195	MSIE 6.0
204.231.180.195	MSIE 6.0
204.231.180.195	MSIE 6.0

	Date
......	16−Jun−02 16:39:02
......	16−Jun−02 16:39:58
	16−Jun−02 16:42:03
	16−Jun−02 16:24:06
	16−Jun−02 16:26:05
......	16−Jun−02 16:42:07
	16−Jun−02 16:58:03
......	16−Jun−02 16:32:06
	16−Jun−02 16:34:10
	16−Jun−02 16:38:40
	16−Jun−02 17:34:20
......	16−Jun−02 17:35:45

IP	Agent	Date	Sess
165.182.168.101	MSIE 5.01	16−Jun−02 16:39:02	1
165.182.168.101	MSIE 5.01	16−Jun−02 16:39:58	1
165.182.168.101	MSIE 5.01	16−Jun−02 16:42:03	1
165.182.168.101	MSIE 5.5	16−Jun−02 16:24:06	2
165.182.168.101	MSIE 5.5	16−Jun−02 16:26:05	2
165.182.168.101	MSIE 5.5	16−Jun−02 16:42:07	2
204.231.180.195	MSIE 6.0	16−Jun−02 16:32:06	3
204.231.180.195	MSIE 6.0	16−Jun−02 16:34:10	3
204.231.180.195	MSIE 6.0	16−Jun−02 16:38:40	3
204.231.180.195	MSIE 6.0	16−Jun−02 17:34:20	4
204.231.180.195	MSIE 6.0	16−Jun−02 17:35:45	4

Real session condition

- L set of log register, $r_{ij} \in L$

- $R=\{r_1,...,r_n\}$ the set of session, where $r_i=(r_{ij})$

- C1: r_{ij}.timestamp $> r_{ij\text{-}1}$.timestamp

- C2: $\bigcup\{r_{ij}\} = L$, completeness

- C3: $\exists! \; i' \neq i, j' / r_{ij}=r_{i'j'}$, each object in L belong to a only one session.

Enforcing these condition we obtain a much more consistent set and also better behaviour descriptions.

Some problems [Berendt01, Cooley99]

- **Single IP address/Multiple Server Sessions.**
  - Proxy server
- **Multiple IP addresses/Single Server Sessions.** For privacy reasons or ISP configuration, it is possible to assign a **random IP** address to a visitor request.
- **Multiple IP address/Single Visitor.** A visitor that accesses a web site from **different machines**, but has the same behaviour each time.
- **Multiple Agent/Single User.** As before, when a visitor uses different machines that may have **different agents.**

User and Session Identification Issues

- Distinguish among different users to a site
- Reconstruct the activities of the users within the site
- Proxy servers and anonymizers
- Rotating IP addresses connections through ISPs
- Missing references due to caching
- Inability of servers to distinguish among different visits

Some solutions

- **Remote Agent**
  - A remote agent is implemented in Java Applet
  - It is loaded into the client only once when the first page is accessed
  - The subsequent requests are captured and send back to the server

- **Modified Browse**
  - The source code of the existing browser can be modified to gain user specific data at the client side

- **Dynamic page rewriting**
  - When the user first submit the request, the server returns the requested page rewritten to include a session specific ID
  - Each subsequent request will supply this ID to the server

- **Heuristics**
  - use a set of assumptions to identify user sessions and find the missing cache hits in the server log

WUM – Heuristics

- Timeout
  - if the time between pages requests exceeds a certain limit, it is assumed that the user is starting a new session

- IP/Agent
  - Each different agent type for an IP address represents a different sessions

- Referring page
  - If the referring page file for a request is not part of an open session, it is assumed that the request is coming from a different session.

- Same IP-Agent/different sessions (Closest)
  - Assigns the request to the session that is closest to the referring page at the time of the request.

- Same IP-Agent/different sessions (Recent)
  - In the case where multiple sessions are same distance from a page request, assigns the request to the session with the most recent referrer access in terms of time

WUM – Heuristics (2)

- **The path completion heuristics**
  - If the referring page file of a session is not part of the previous page file of that session, the user must have accessed a cached page
  - The "back" button method is used to refer a cached page.
  - Assigns a constant view time for each of the cached page file

Sessionization – Example

Time	IP	URL	Ref	Agent
0:01	1.2.3.4	A	-	IE5;Win2k
0:09	1.2.3.4	B	A	IE5;Win2k
0:10	2.3.4.5	C	-	IE4;Win98
0:12	2.3.4.5	B	C	IE4;Win98
0:15	2.3.4.5	E	C	IE4;Win98
0:19	1.2.3.4	C	A	IE5;Win2k
0:22	2.3.4.5	D	B	IE4;Win98
0:22	1.2.3.4	A	-	IE4;Win98
0:25	1.2.3.4	E	C	IE5;Win2k
0:25	1.2.3.4	C	A	IE4;Win98
0:33	1.2.3.4	B	C	IE4;Win98
0:58	1.2.3.4	D	B	IE4;Win98
1:10	1.2.3.4	E	D	IE4;Win98
1:15	1.2.3.4	A	-	IE5;Win2k
1:16	1.2.3.4	C	A	IE5;Win2k
1:17	1.2.3.4	F	C	IE4;Win98
1:25	1.2.3.4	F	C	IE5;Win2k
1:30	1.2.3.4	B	A	IE5;Win2k
1:36	1.2.3.4	D	B	IE5;Win2k

Based on examples in tutorial
PKDD-2002 by Berendt et. al.

Sessionization – Example

Sort the users (IP+Agent)

Time	IP	URL	Ref	Agent
0:01	1.2.3.4	A	-	IE5;Win2k
0:09	1.2.3.4	B	A	IE5;Win2k
0:10	2.3.4.5	C	-	IE4;Win98
0:12	2.3.4.5	B	C	IE4;Win98
0:15	2.3.4.5	E	C	IE4;Win98
0:19	1.2.3.4	C	A	IE5;Win2k
0:22	2.3.4.5	D	B	IE4;Win98
0:22	1.2.3.4	A	-	IE4;Win98
0:25	1.2.3.4	E	C	IE5;Win2k
0:25	1.2.3.4	C	A	IE4;Win98
0:33	1.2.3.4	B	C	IE4;Win98
0:58	1.2.3.4	D	B	IE4;Win98
1:10	1.2.3.4	E	D	IE4;Win98
1:15	1.2.3.4	A	-	IE5;Win2k
1:16	1.2.3.4	C	A	IE5;Win2k
1:17	1.2.3.4	F	C	IE4;Win98
1:26	1.2.3.4	F	C	IE5;Win2k
1:30	1.2.3.4	B	A	IE5;Win2k
1:36	1.2.3.4	D	B	IE5;Win2k

0:01	1.2.3.4	A	-	IE5;Win2k
0:09	1.2.3.4	B	A	IE5;Win2k
0:19	1.2.3.4	C	A	IE5;Win2k
0:25	1.2.3.4	E	C	IE5;Win2k
1:15	1.2.3.4	A	-	IE5;Win2k
1:26	1.2.3.4	F	C	IE5;Win2k
1:30	1.2.3.4	B	A	IE5;Win2k
1:36	1.2.3.4	D	B	IE5;Win2k

0:10	2.3.4.5	C	-	IE4;Win98
0:12	2.3.4.5	B	C	IE4;Win98
0:15	2.3.4.5	E	C	IE4;Win98
0:22	2.3.4.5	D	B	IE4;Win98

0:22	1.2.3.4	A	-	IE4;Win98
0:25	1.2.3.4	C	A	IE4;Win98
0:33	1.2.3.4	B	C	IE4;Win98
0:58	1.2.3.4	D	B	IE4;Win98
1:10	1.2.3.4	E	D	IE4;Win98
1:17	1.2.3.4	F	C	IE4;Win98

Sessionization – Example

Sessionize using heuristics (h1 with 30 min)

0:01	1.2.3.4	A	-	IE5;Win2k
0:09	1.2.3.4	B	A	IE5;Win2k
0:19	1.2.3.4	C	A	IE5;Win2k
0:25	1.2.3.4	E	C	IE5;Win2k
1:15	1.2.3.4	A	-	IE5;Win2k
1:26	1.2.3.4	F	C	IE5;Win2k
1:30	1.2.3.4	B	A	IE5;Win2k
1:36	1.2.3.4	D	B	IE5;Win2k

0:01	1.2.3.4	A	-	IE5;Win2k
0:09	1.2.3.4	B	A	IE5;Win2k
0:19	1.2.3.4	C	A	IE5;Win2k
0:25	1.2.3.4	E	C	IE5;Win2k

1:15	1.2.3.4	A	-	IE5;Win2k
1:26	1.2.3.4	F	C	IE5;Win2k
1:30	1.2.3.4	B	A	IE5;Win2k
1:36	1.2.3.4	D	B	IE5;Win2k

The h1 heuristic (timeout=30 min) will result in the two sessions

Sessionization – Example

Sessionize using heuristics (with href)

0:22	1.2.3.4	A	-	IE4;Win98
0:25	1.2.3.4	C	A	IE4;Win98
0:33	1.2.3.4	B	C	IE4;Win98
0:58	1.2.3.4	D	B	IE4;Win98
1:10	1.2.3.4	E	D	IE4;Win98
1:17	1.2.3.4	F	C	IE4;Win98

By using the reffer-based heuristics, we have only a single session

Sessionization – Example

Path completion

0:22	1.2.3.4	A	-	IE4;Win98
0:25	1.2.3.4	C	A	IE4;Win98
0:33	1.2.3.4	B	C	IE4;Win98
0:58	1.2.3.4	D	B	IE4;Win98
1:10	1.2.3.4	E	D	IE4;Win98
1:17	1.2.3.4	F	C	IE4;Win98
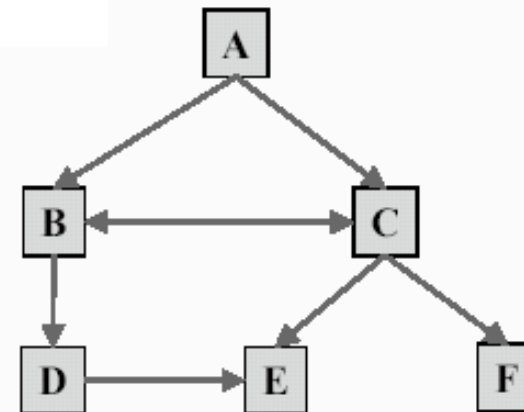
A=>C , C=>B , B=>D , D=>E , C=>F

Need to look for the shortest backwards path from E to C based on the site topology. Note, however, that the elements of the path need to have occurred in the user trail previously.

E=>D, D=>B, B=>C

Final comments

- What happen if the web page content is changed during the study period?

- A➔B, B➔D but there are two versions of D.

- If we want study the user behaviour, it is necessary to consider to maintain a change register.

- Proposal solution LOGML [Punin WEBKDD'01]

Mechanisms for session identification [Berendt 2002]

Method	Description	Privace Concerns	Advantages	Disadvantages
IP Adress + Agent	Assume each unique IP address/Agent pair is a unique user	Low	Always available. No Additional technology required.	Not guaranteed to be unique. Defeated by rotating Ips.
Embedded Sessions Ids	Use dinamically generated pages to associate ID with every hyperlink	Low to Medium	Always available. Independent of IP address	Cannot capture repeat visitors. Additional overhead for dynamic pages
Registration	User explicity logs into the site	Medium	Can track individuals not just browsers	Many users won't register. Not available before registration
Cookie	Save ID on the client machine	Medium to High	Can track repeat visit from same browser	Can be turned off by users
Software Agents	Program loaded into browser and sends back usage data	High	Accurrate usage data for a single site	Likely to be rejected by users

New Approaches for Session Reconstruction Process

≫ Web User Session Reconstruction Using Integer Programming

Pablo E. Román proman@ing.uchile.cl

Robert F. Dell dell@nps.edu

Juan D. Velásquez jvelasqu@dii.uchile.cl

Preliminary: Properties of sessions

Traditional Sessionization

$y = -2{,}1748x + 8{,}454$

$R^2 = 0{,}9181$
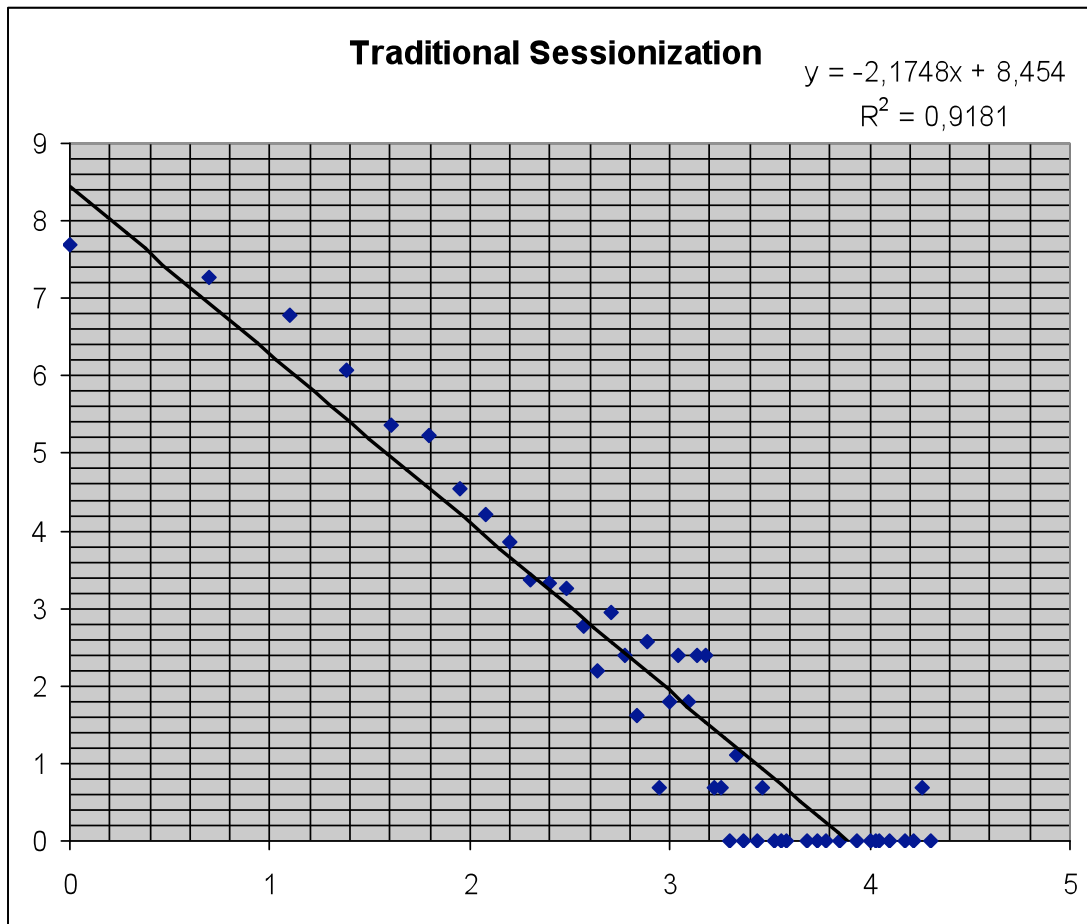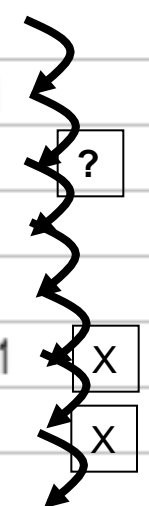
- Session length distribution.
- Approx power law.
- Brownian motion reaching a threshold.
- Very good fit in real cases.
- Is a universal property.

The Problem

- We need to recover individual user sessions
  - Using the registers from a web log
  - Using the time precedence of the registers
  - Using the web page links
  - Should show the distribution behavior of session size.

Finding Sessions (Heuristic)

#	IP	Time	Method/URL/Protocol	Status	Bytes	Agent
1	"165.182.168.101"	16/04/2008:16:24:06	GET index.htm HTTP/1.1	200	3821	Mozilla/4.0
2	"165.182.168.101"	16/04/2008:16:24:07	GET /informacion/academicos/index.htm HTTP/1.1	200	1345	Mozilla/4.0
3	"165.182.168.101"	16/04/2008:16:24:08	GET /informacion/academicos/Profesores_Titulares/index.htm HTTP/1.1	200	2567	Mozilla/4.0
4	"190.20.216.76"	16/04/2008:16:24:09	GET /images/borde_titulos.jpg HTTP/1.1	200	4678	Mozilla/4.0
5	"190.44.161.57"	16/04/2008:16:24:10	/ HTTP/1.1	200	3821	Mozilla/4.0
6	"165.182.168.101"	16/04/2008:16:24:11	GET index.htm HTTP/1.1	200	3821	Mozilla/4.0
7	"165.182.168.101"	16/04/2008:16:24:12	GET informacion/academicos/index.htm HTTP/1.1	200	1345	Mozilla/4.0
8	"165.182.168.101"	16/04/2008:16:24:13	GET index.htm HTTP/1.1	200	3821	Mozilla/4.0
9	"165.182.168.101"	16/04/2008:16:24:14	GET index.htm HTTP/1.1	200	3821	Mozilla/4.0
10	"165.182.168.101"	16/04/2008:16:24:15	GET /exalumnos_empresas/index.htm HTTP/1.1	200	4563	Mozilla/4.0
11	"165.182.168.101"	16/04/2008:16:24:16	GET /proyectos_investigacion/index.htm HTTP/1.1	200	1224	Mozilla/4.0
12	"165.182.168.101"	16/04/2008:16:24:17	GET /publicaciones/index.htm HTTP/1.2	200	2543	Mozilla/4.0
13	"165.182.168.101"	16/04/2008:16:24:18	GET /~cea/ index.html	200	1924	Mozilla/4.0

It is a combinatorial problem!!
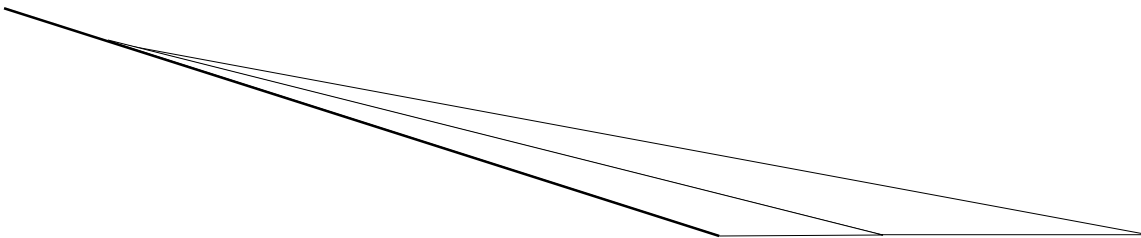
Integer Program to construct sessions: Decision Variables

- X_{ros} : 1 if log register "r" is assigned as the "o-th" request during session "s" and zero otherwise.

Log register

Index r	IP	Time	Method/URL/Protocol	Status	Bytes	Agent
1	"190.182.168.10"	16/04/2008:16:24:06	GET index.htm HTTP/1.1	200	3821	Mozilla/4.0
2	"65.182.168.101"	16/04/2008:16:24:07	GET /informacion/academicos/index.htm HTTP/1.1	200	1345	Mozilla/4.0
3	"165.182.168.1"	16/04/2008:16:24:08	GET /informacion/academicos/Profesores_Titulares/index.htm HTTP/1	200	2567	Mozilla/4.0
4	"190.20.216.76"	16/04/2008:16:24:09	GET /images/borde_titulos.jpg HTTP/1.1	200	4678	Mozilla/4.0
5	"190.44.161.57"	16/04/2008:16:24:10	/ HTTP/1.1	200	3821	Mozilla/4.0
6	"165.182.168.101"	16/04/2008:16:24:11	GET index.htm HTTP/1.1	200	3821	Mozilla/4.0
7	"65.182.168.1"	16/04/2008:16:24:12	GET informacion/academicos/index.htm HTTP/1.1	200	1345	Mozilla/4.0
8	"65.182.168.10"	16/04/2008:16:24:13	GET index.htm HTTP/1.1	200	3821	Mozilla/4.0
9	"65.182.168.10"	16/04/2008:16:24:14	GET index.htm HTTP/1.1	200	3821	Mozilla/4.0
10	"65.182.168.10"	16/04/2008:16:24:15	GET /exalumnos_empresas/index.htm HTTP/1.1	200	4563	Mozilla/4.0
11	"65.182.168.10"	16/04/2008:16:24:16	GET /proyectos_investigacion/index.htm HTTP/1.1	200	1224	Mozilla/4.0
12	"65.182.168.10"	16/04/2008:16:24:17	GET /publicaciones/index.htm HTTP/1.2	200	2543	Mozilla/4.0
13	"165.182.168.101"	16/04/2008:16:24:18	GET /~cea/ index.html	200	1924	Mozilla/4.0

Sessions

s	1	2	3	4	5	.	.	.
o								
1	1	5	6	8	9			
2	2		7	12	13			
3	3		10					
4			11					
.								
.								
.								
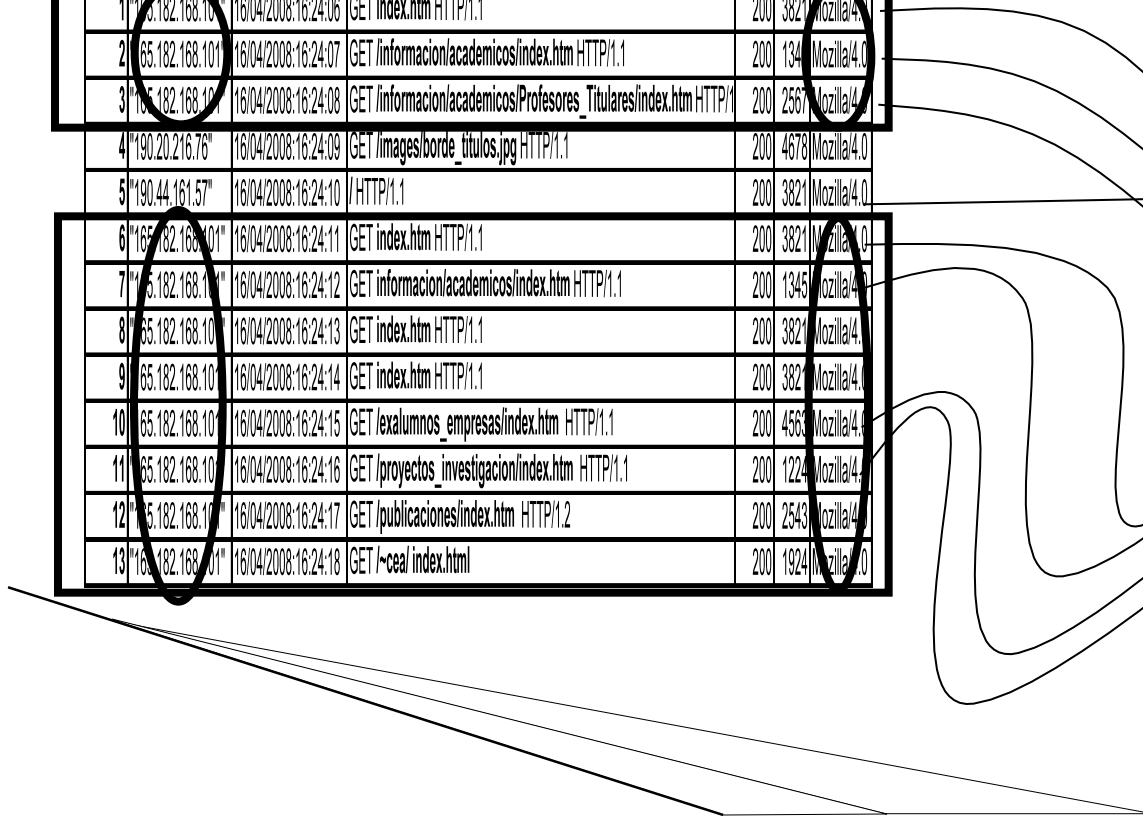
Integer Program

Maximize $\sum_{ros} C_o X_{ros}$

Subject to:

r used almost 1

$$\sum_{os} X_{ros} \leq 1 \quad \text{s have almost 1 r in o} \quad \forall r \qquad (1)$$

$$\sum_{r} X_{ros} \leq 1 \quad \text{ordering property in s} \quad \forall o, s \qquad (2)$$

$$X_{r,o+1,s} \leq \sum_{r' \in bpage_r} X_{r',o,s} \quad \forall r, o, s \qquad (3)$$

$$X_{ros} \in \{0,1\} \forall r, o, s,$$
$$X_{ros} = 0, \forall r \in first, o > 1, s$$

Which Co?

‣ Plausibility Argument: Approximating an entropy function.

‣ Using $Co=Log(1/Po)$

‣ Po : the probability of having a session of size o

‣ Empirical result: Inverse Gaussian, or nearly asymptotic power law is measured on several site.
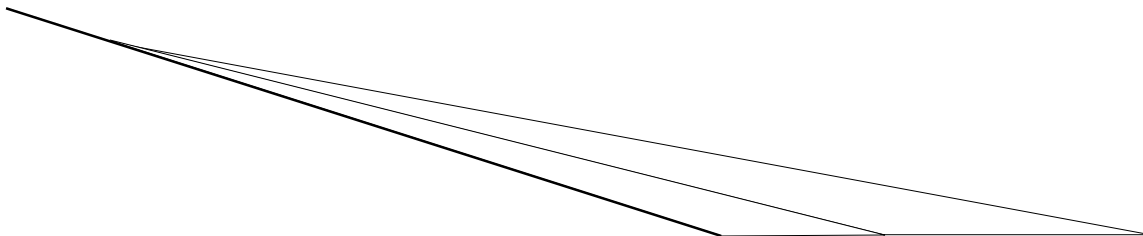
The experiment

The Web Site: http://www.dii.uchile.cl
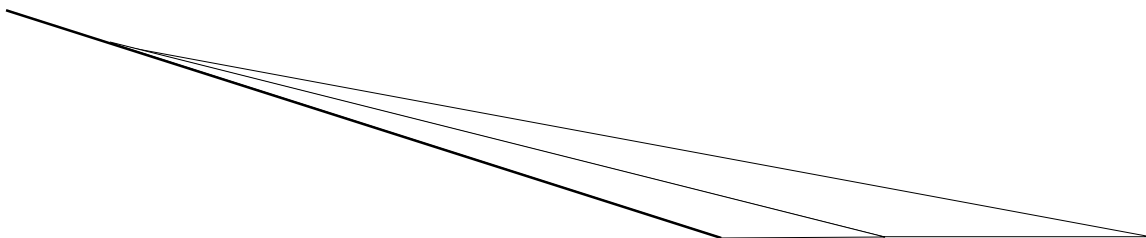
- Department of industrial engineering
- Departmental information
- Project pages
- Research group pages
- Personal pages
- Student organizations
- Web mail
- Over 3,500,000 raw registers per month

Partial view of web page links for DII site
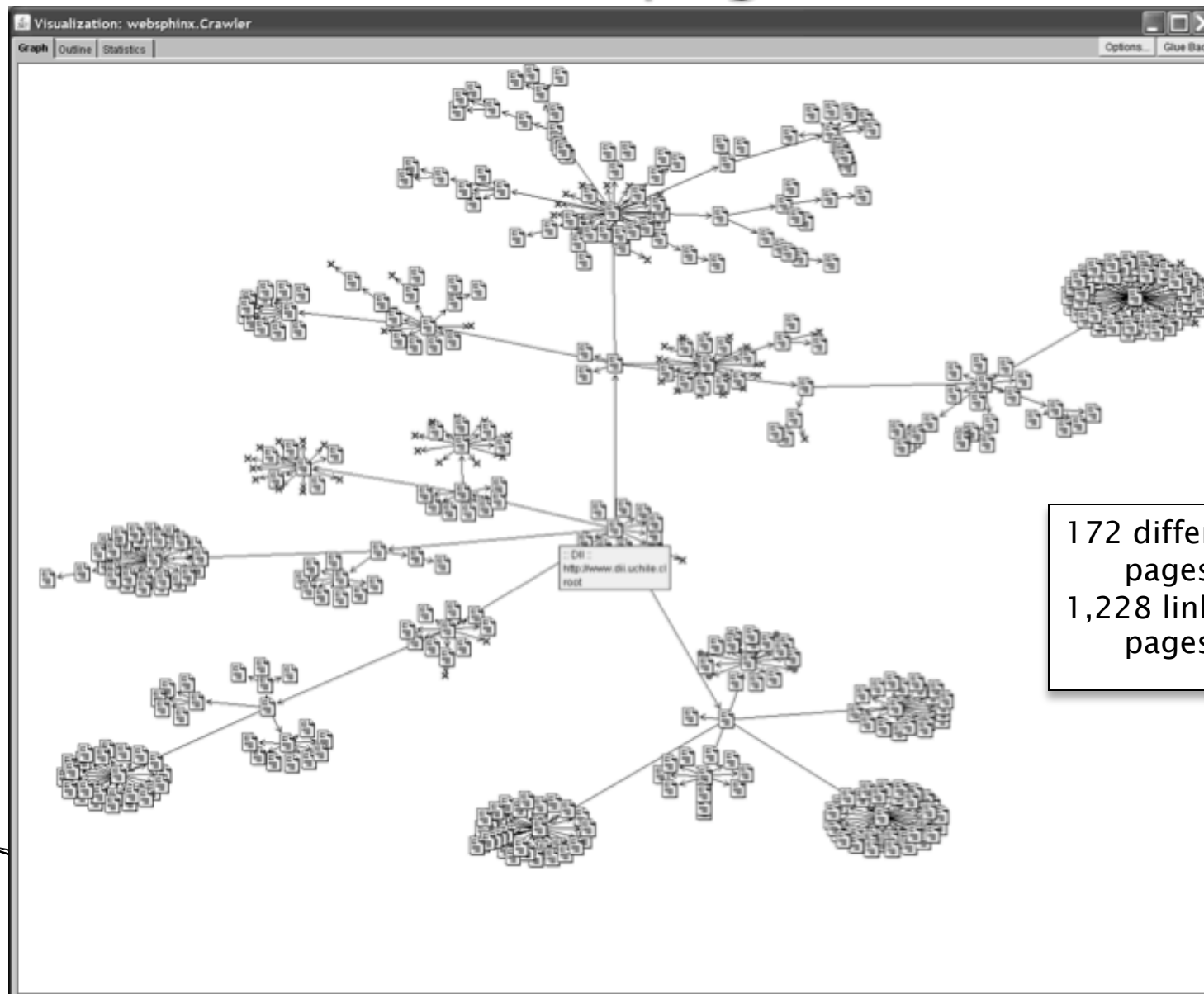
172 different text pages with links.
1,228 links between pages.

Test Data (one month)

▸ 3,756,006 registers including text pages, images, binary documents, video clips, robot spiders, web mail access, server errors and even hacking attempts.

- 302,503 clean registers
- 172 different pages
- 16,985 different IP addresses; 98% have less than 50 pages visited. 84% have 3 or less different pages visited.
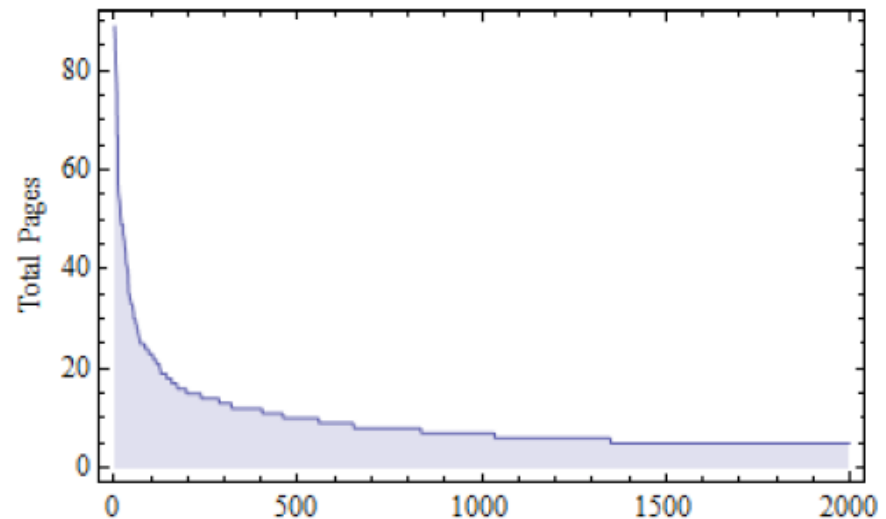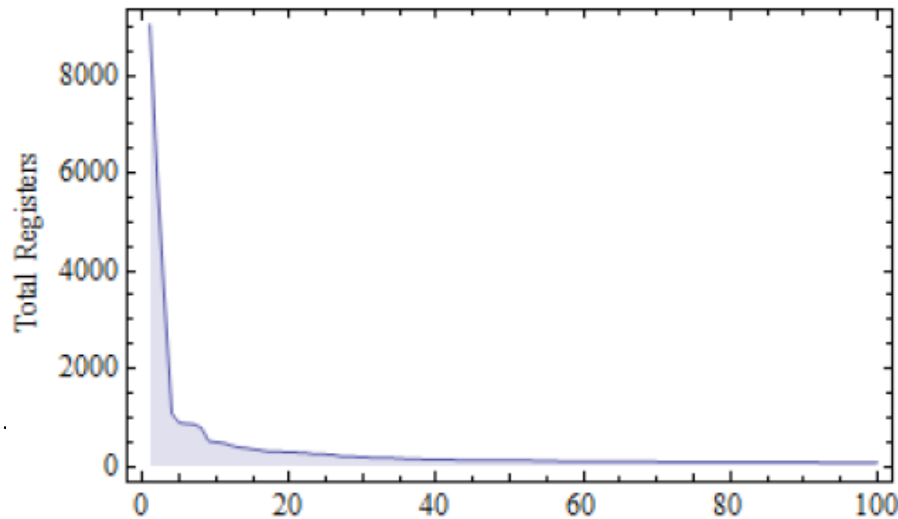
Restricting to interesting session

$$S \;=\; \sum_p f_p Log_N(1/f_p)$$

Log Number of Registers
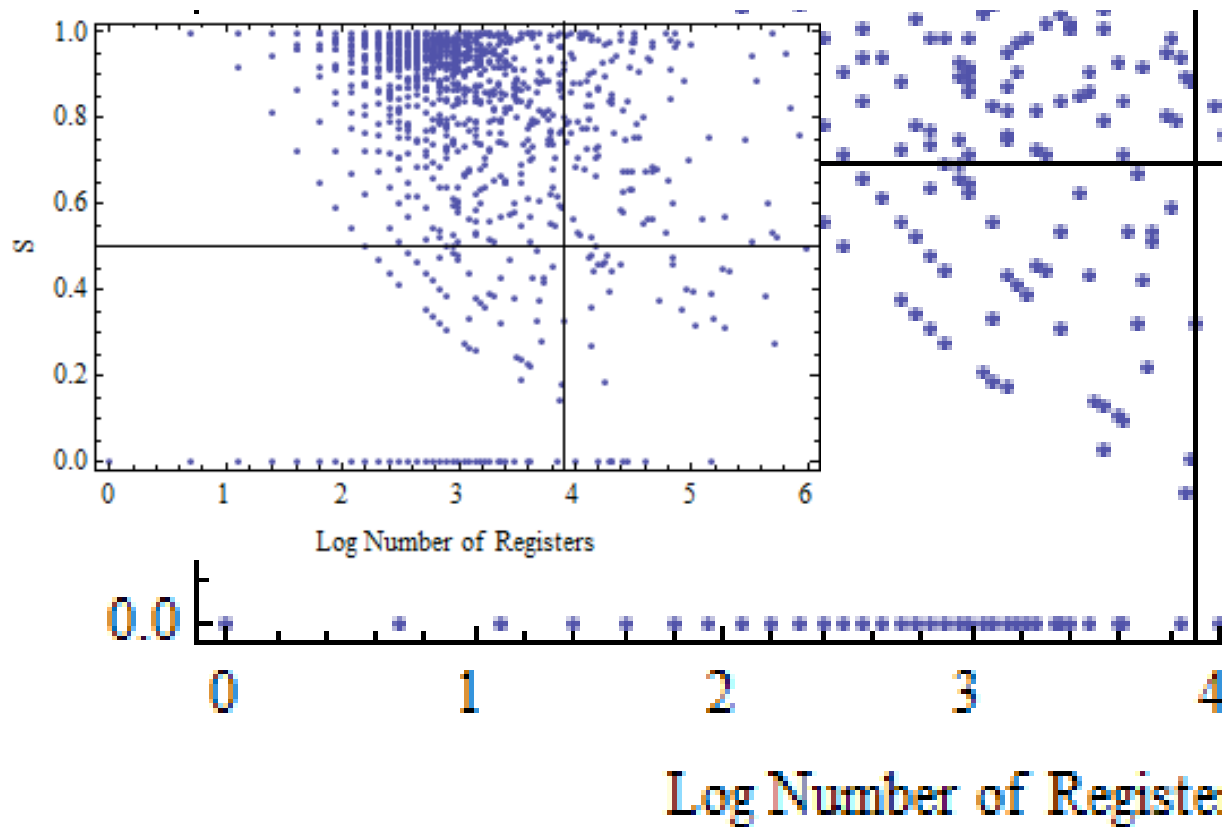
Log Number of Register
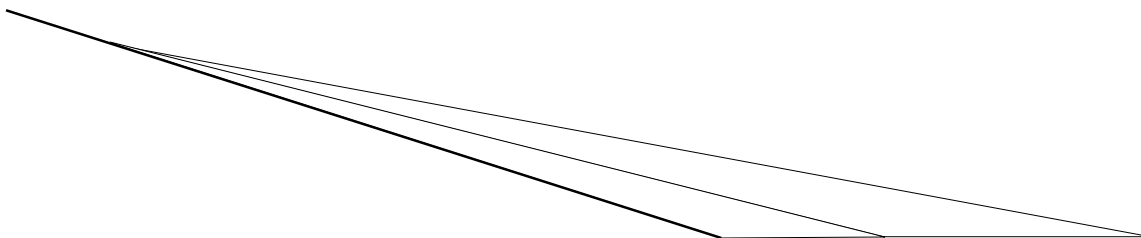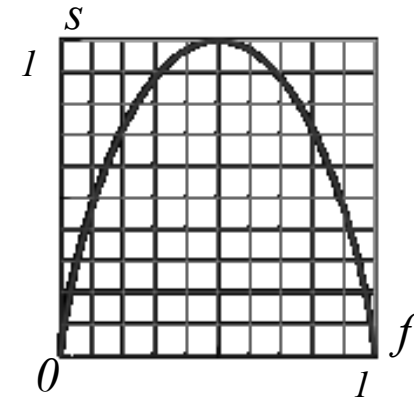
Reducing complexity

‣ We estimate the number of binary variables of 10^{10} -> unsolvable!

‣ Fortunately we can separate the problem in small problems (chunks) by considering a maximum time between registers of 1500 second.

‣ Each chunks are restricted to have more than 50 register in order to avoid chunks proliferation.

‣ 403 chunks were identified
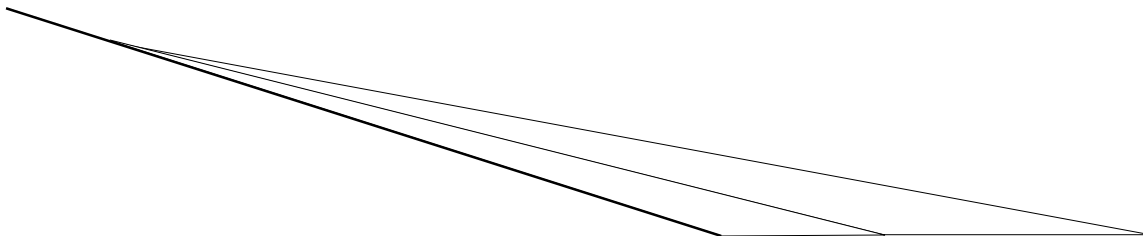
Sessionization Results

- Used IP registers with high diversity of pages and high hits per page (17.3% of the totals).
  - 403 different instances or chunks (divided by IP address and time).
  - Integer programs varying between 4,000 to 292,000 binary variables and 8,000 to 281,000 constraints.
- Generated by GAMS and solved by CPLEX v10.1.0 using a 1.6 Ghz Pc.
- We set the gap to 1% and the maximum time to 300s.
  - 85% of the instance solved to within 1%, the other 15% reached the 300s limit (we used the best solution found by the limit).
  - Around 3hrs to solve the instances.

The quality of the resulting session

- We don't know the real sessions.
- An empirical power rule for session size has been observed in the literature [Huberman et al. 1998] [Vasquez et al. 2006].
- The correlation coefficient and standard error of fitting to a power law gives us a sense of the quality of the sessions.
- Our correlation coefficient is 0.9784 and our standard error is 0.3817. A common heuristic has a correlation coefficient of 0.9181 and a standard error of 0.6401.
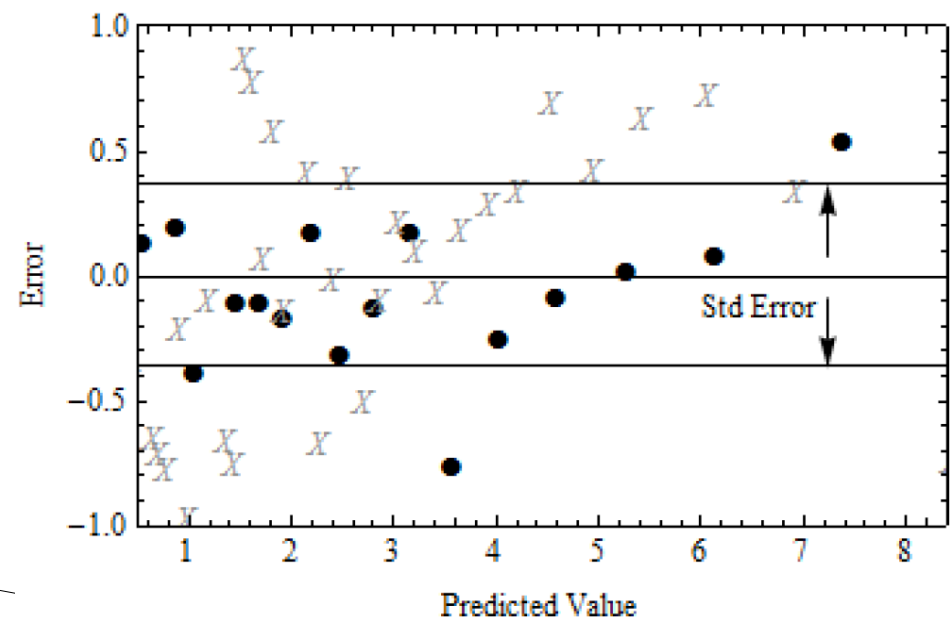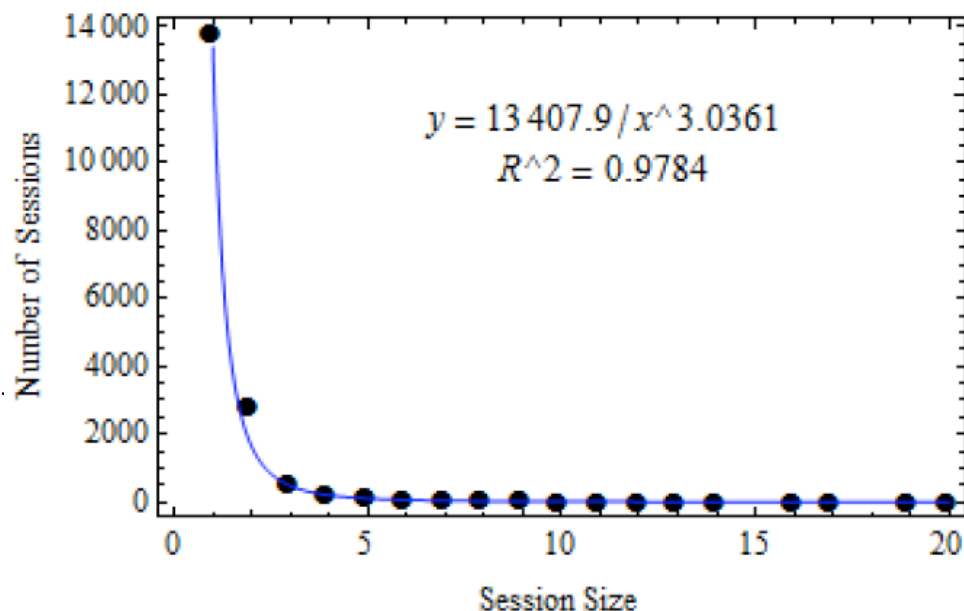
The coefficient

|	$C_{ro} =$	R^2	StdErr	Total Sessions
1	$1/\sqrt{o}$	0.88	1.10	12, 502
2	$Log(o)$	0.93	0.66	12, 403
3	$3/2Log(o) + (o-3)^2/12o$	0.94	0.59	12, 403
4	o	0.93	0.63	12, 409
5	o^2	0.92	0.72	12, 410
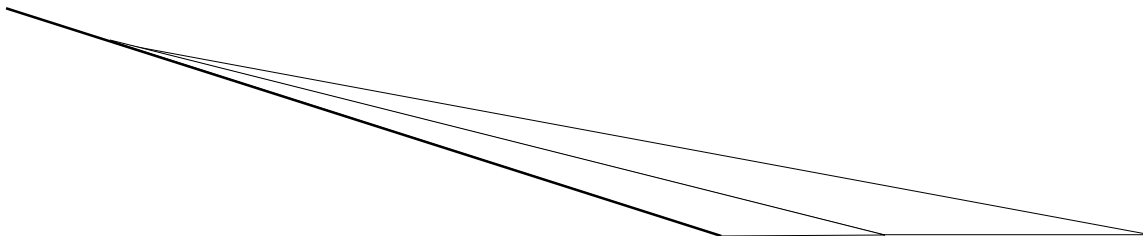
Flexibility of MIP scheme: Additional experiment

▸ Modifying the objective function coefficient to find the maximum number of session of a given size.

◦ fix $C_o=1$ for the desired size=o and zero otherwise.

◦ The results

Size	Num. Sessions
2	3,000
3	1,509
4	755
5	435
6	257
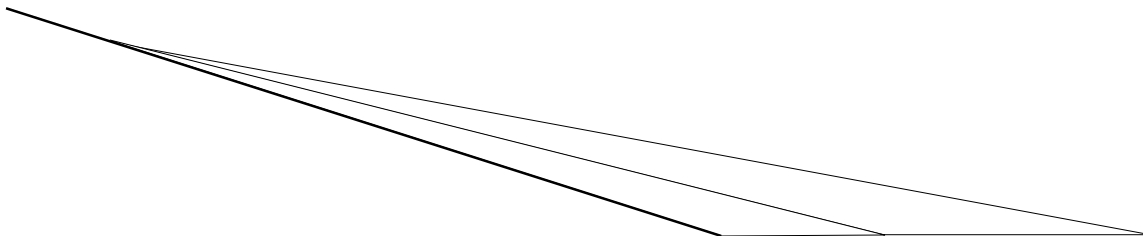
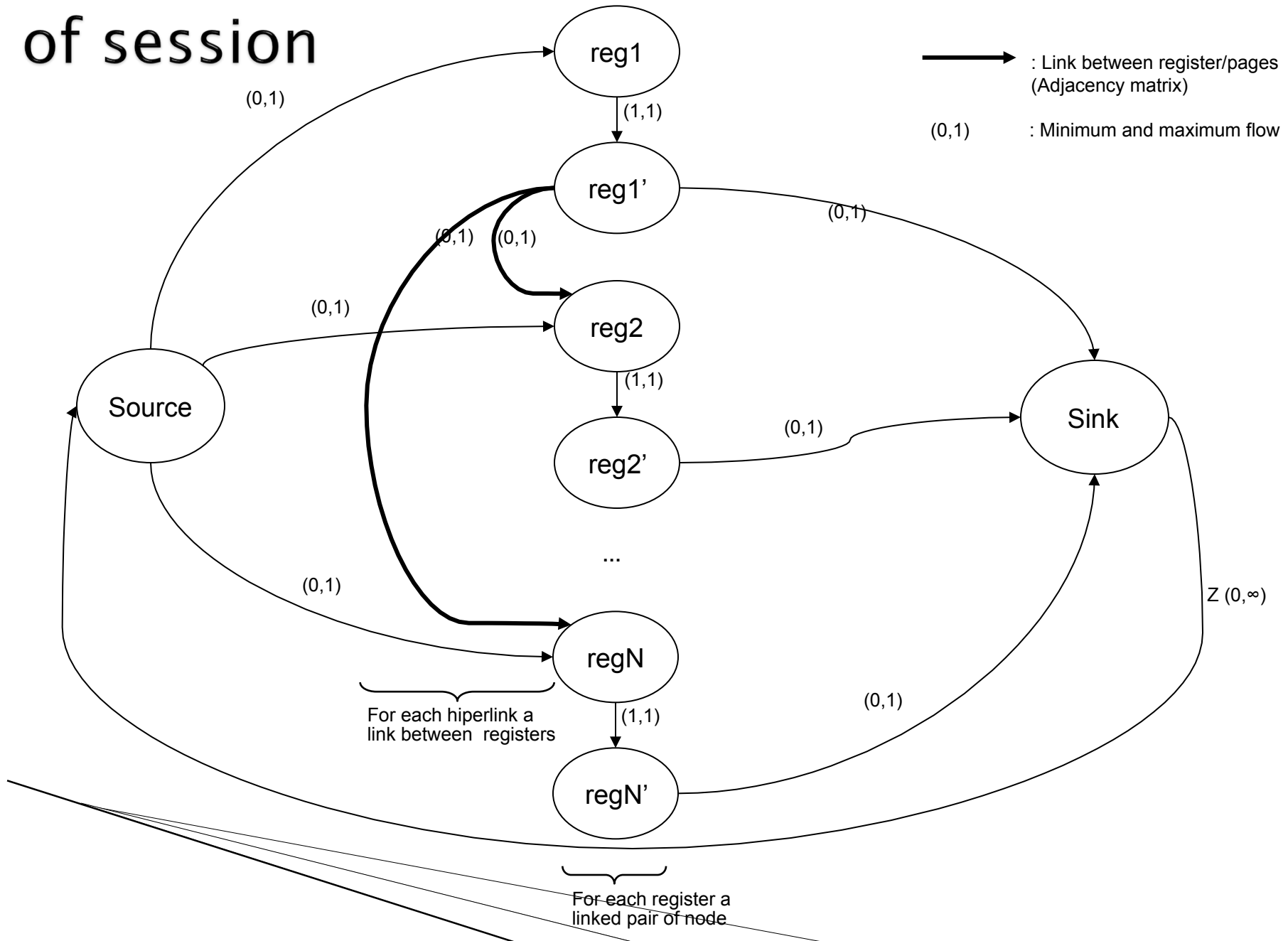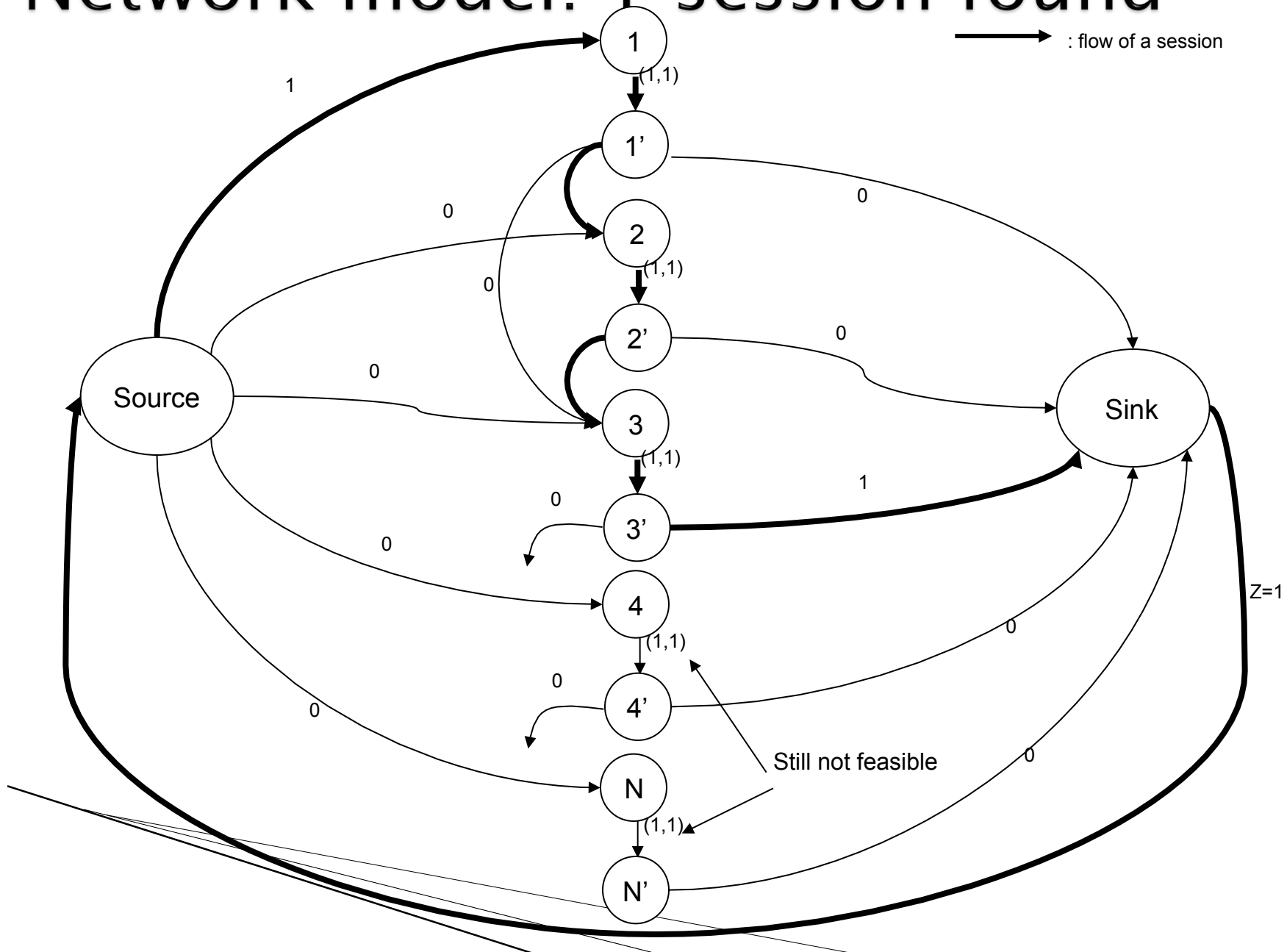# The problem: The time for solving IP
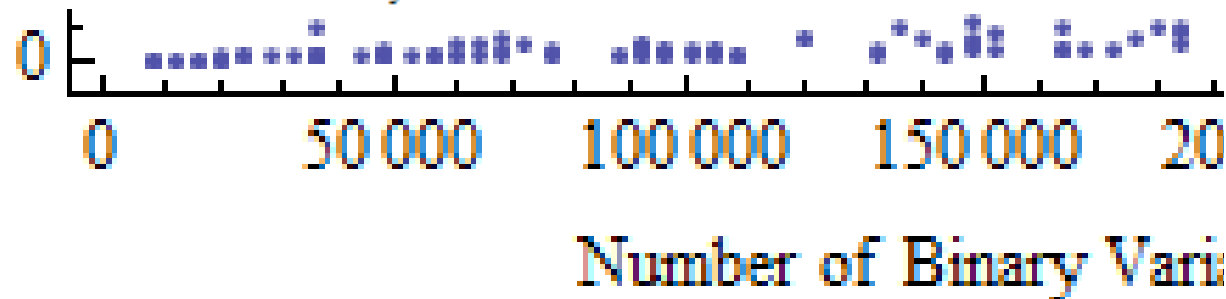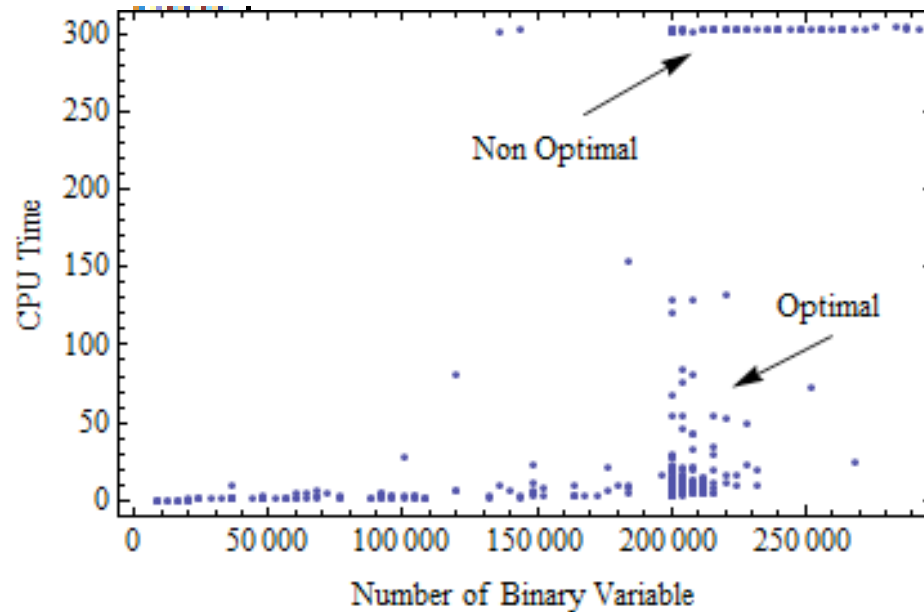
Network Flow Model

- ‣ Solutions prove to be solved in polynomial time (i.e. Ford–Fulkerson)
- ‣ Same linear restriction converted to build an network (links, time ordering).
- ‣ Minimize the flow: Number of session
- ‣ Each session is a unit of flow.
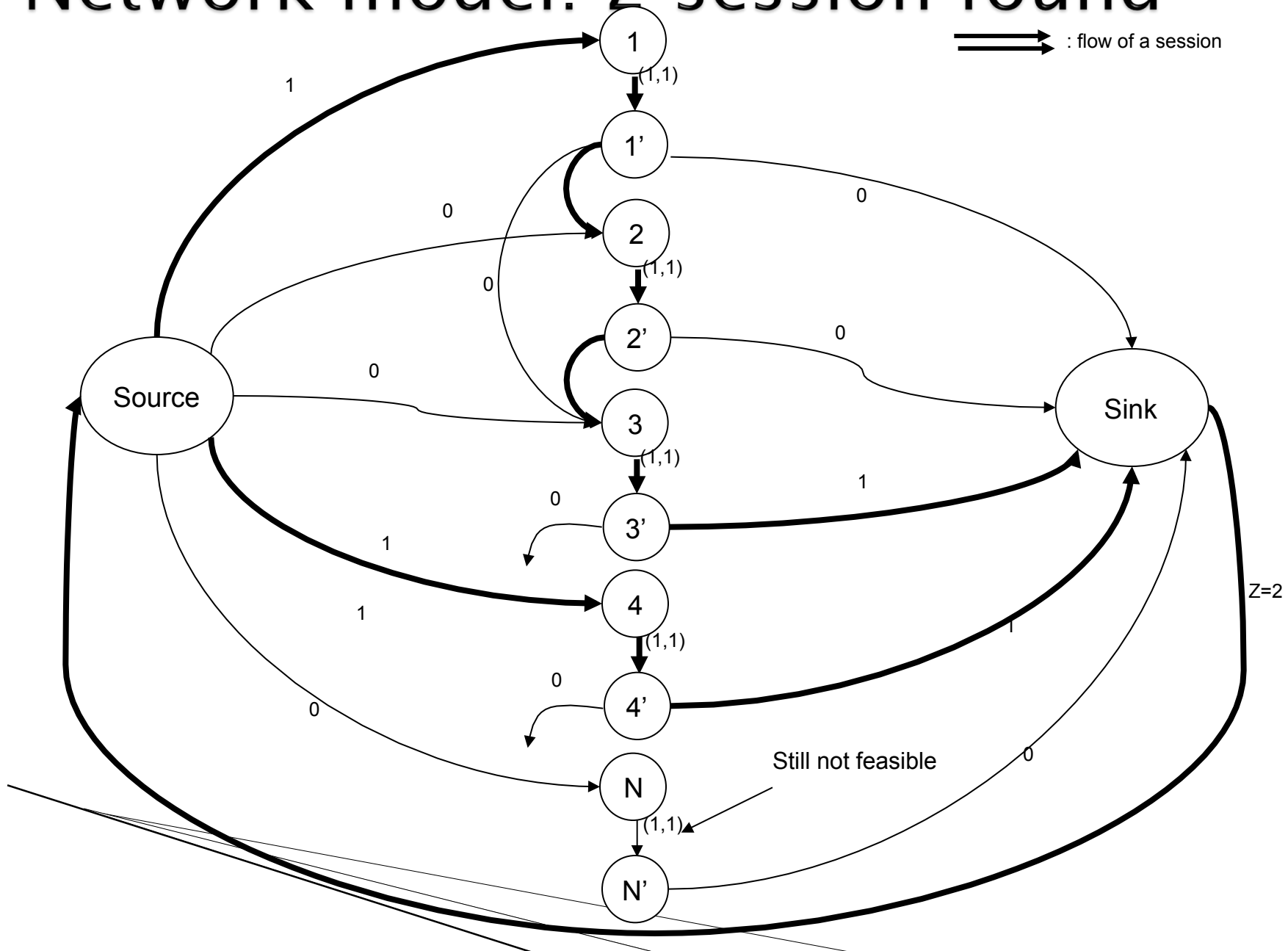
Network model to minimize the number of session

reg1

(0,1)

(1,1)

reg1'

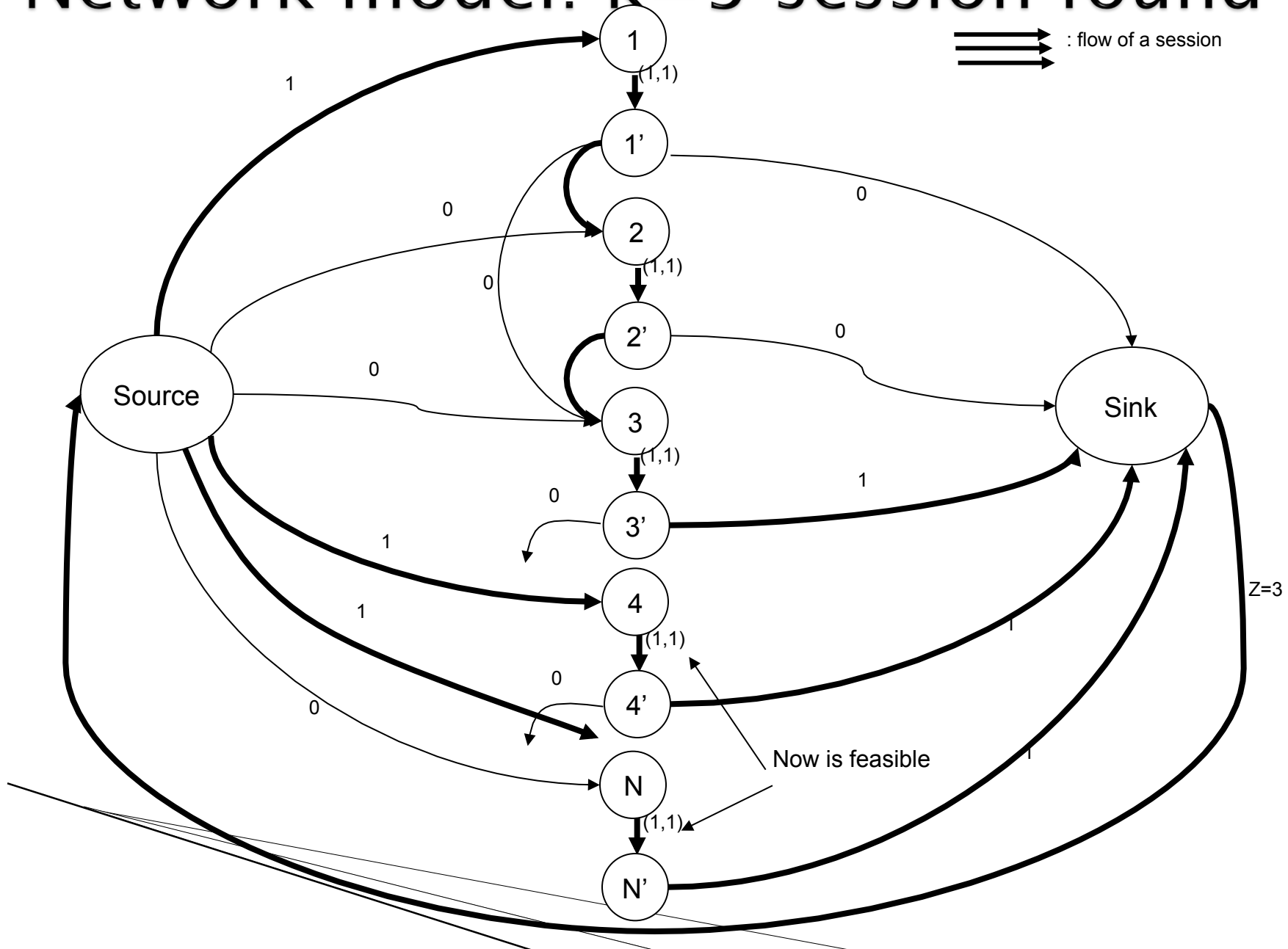(0,1)

(0,1) (0,1)

(0,1)

Source

reg2

(1,1)

reg2'

(0,1)

Sink

...

(0,1)

regN

For each hiperlink a
link between registers

(1,1)

Z (0,∞)

(0,1)

regN'

For each register a
linked pair of node

: Link between register/pages
(Adjacency matrix)

(0,1)   : Minimum and maximum flow
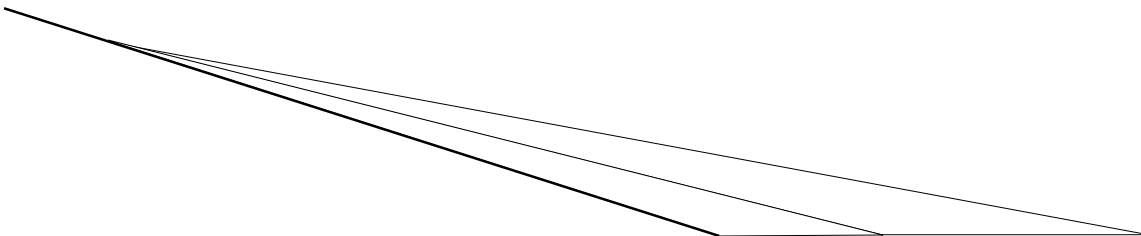
Network model: 1 session found

Network model: 2 session found

: flow of a session

1

(1,1)

0

0

0

0

0

0

0

(1,1)

0

0

1

Source

0

0

1

1

Sink

(1,1)

0

1

(1,1)

0

1

0

Still not feasible

Z=2

(1,1)

1 1' 2 2' 3 3' 4 4' N N'

Network model: k=3 session found

A network model to minimize the number of sessions

- Takes 5min to solve for the 403 chunks
  - Overall it finds results similar to the Integer Program.
  - A total of 12,367 sessions vs 12,403 sessions found by the Integer program (0,3% Gap).
  - Network model can also find the maximize number of sessions possible for a given sequence of pages.
  - Network model Can't find maximum number of sessions of a given size such as the Integer Program.

Others variations: Maximum likelihood

‣ Finding the maximum number of copies of a given session (network).

‣ Maximizing the number of session of a given size (MIP).

‣ Maximizing the number of sessions with a given web page in a given order (MIP).

Maximum number of sessions
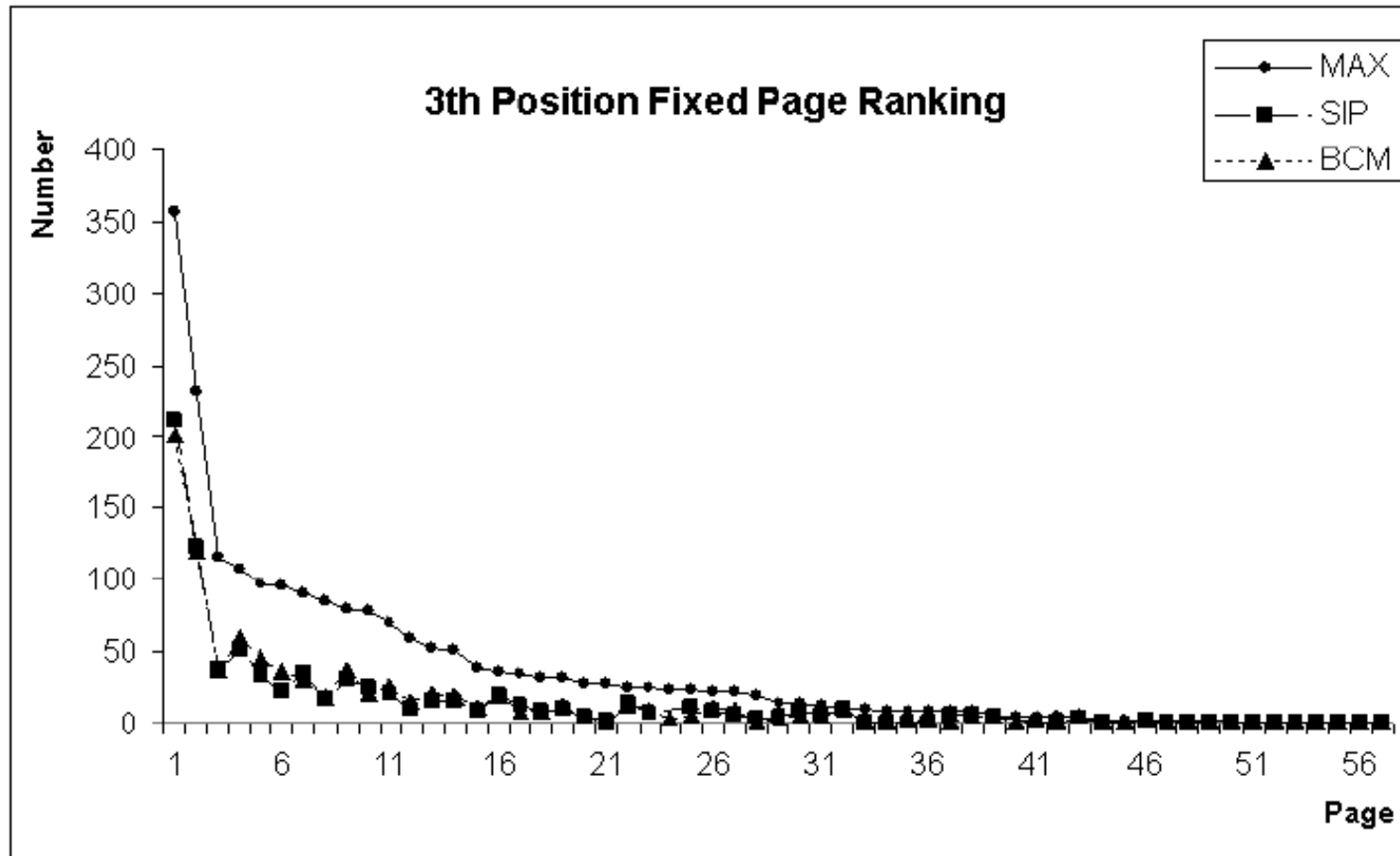
29 hrs of resolution.

Conclusions

- The Integer program and network find sessions that have a better fit with an empirical distribution than a commonly used heuristic.

- The Integer Program can also find the maximum number of sessions of a given size.

- The network model can also find the maximum number of sessions of a specific sequence of pages.
  - Applied to a recommender system

References

[Berent et al. 1999] B. Berendt, A. Hotho, and G. Stumme. Data preparation for mining world wide web browsing patterns. *Journal of Knowlegde and Information Systems*, 1(1):5-32, 1999.

[Berend et al. 2001] B. Berendt, B. Mobasher, M. Spiliopoulou, and J. Wiltshire. Measuring the accuracy of sessionizers for web usage analysis. In *Proc.of the Workshop on Web Mining, First SIAM Internat.Conf. on Data Mining*, pages -14, 2001.

[Huberman et al. 1998] B. Huberman, P. Pirolli, J. Pitkow, and R. M. Lukose. Strong regularities in world wide web sur ng. *Science*, 280(5360):95-97, 1998.

[Velasquez & Palade 2008] J.D.Velásquez and V. Palade. *Adaptive Web Sites: A Knowledge Extraction from Web Data Approach*. IOS Press, Amsterdam, NL, 2008.

[Spiliopoulou et al. 2003] M. Spiliopoulou, B. Mobasher, B. Berendt, and M. Nakagawa. A framework for the evaluation of session reconstruction heuristics in web-usage analysis. *INFORMS Journal on Computing*, 15(2):171-190, 2003.

[Vasquez et al. 2006] A. Vazquez, J. G. Oliveira, Z. Dezso, K.-I. Goh, I. Kondor, and A.-L. Barabasi. Modeling bursts and heavy tails in human dynamics. *PHYSICAL REVIEW E*, 73(3):036127, 2006.

Section 2.3

>> The Information contained in a web page

Web page content

- **Another dimension of the business** is the **content presented by the web pages** to the client.

- **Web content can be represented by an object**, that could be of diferent types: **Multimedia** or **HTML Text**.

- Analysis of web page content usually are made on the **text content**.

- **Example:**
  - ¿Which word or concepts are more important to the user?

Web text content

- From different web page content, **special attention** receive the **free text**.

- For the moment, a **searching** is performed by using **key words**.

- It is necessary to represent the text information in a **feature vector**, before to apply a **mining process**.

- **The representation must consider that the words in the web page don't have the same importance.**

Vector Space Model

● In Information Retrieval a set of document (X) are represented by a matrix $M=(m_{ij})=M(X)$

● This matrix represent **the representativity** of a set of words for a document.

● The index **i** represent the different **"relevant" word** that appears on the all the document in X.

● The index **j** represent the different **document** that appears on X.

● The entry $\mathbf{m_{ij}}$ of the matrix M represent the "**importance**" or weight of this **word i** on the **document j**.

● **Then a Document is represented by a column vector of this numeric matrix.**

Web page: vectorial representation

- Its vectorial representation would be a matrix of RxQ.
- Q is the number of pages in the web site and R is the number of different words in web page.
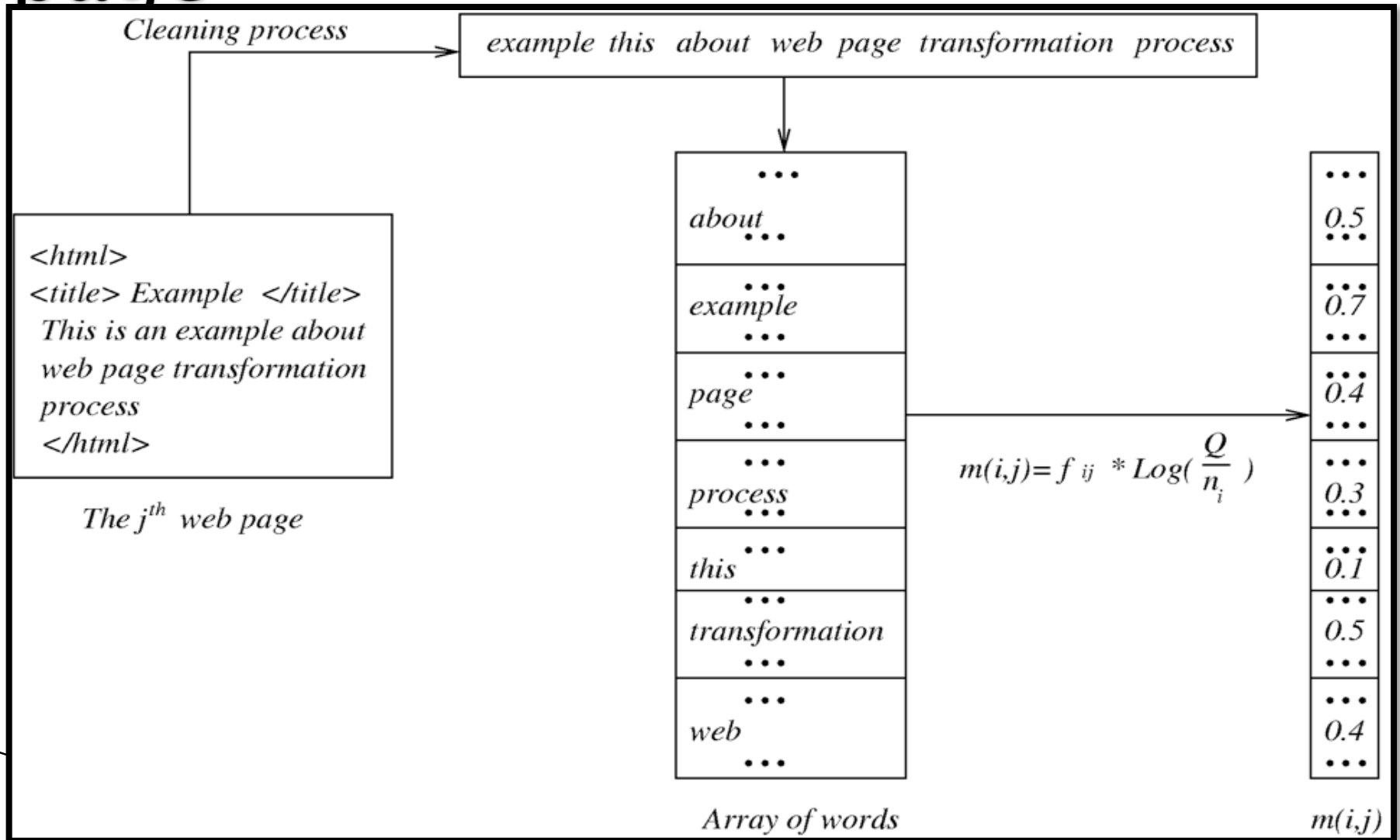- Each entry of the matrix correspond to the number of word found at the respective web page.

	Word	1	2	…	Q
1	advise	1	0	…	1
2	business	0	1	…	0
.	…	.	.	…	.
.	…	.	.	…	.
.	…	.	.	…	.
.	…	.	.	…	.
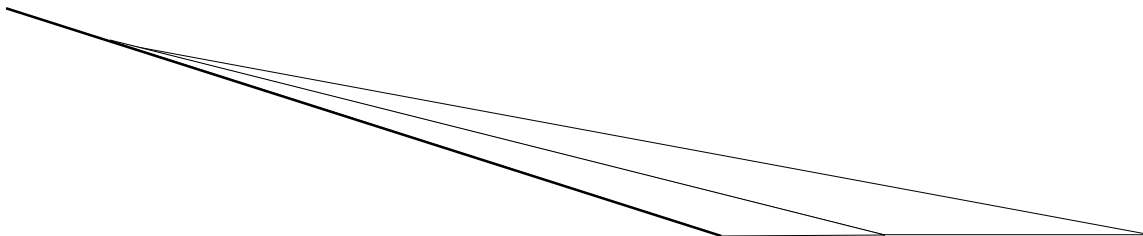
Different measure of representativeness

- Q: number of documents

- **Inverse Document Frecuency (IDF)**: $m_{ij} = log(Q/n_i)$

- **Term Frequency Inverse Document Frecuency (TFIDF)**: $m_{ij} = f_{ij}*log(Q/n_i)$, where f_{ij} is the number of ocurrences of the word i in the document j.

Vector representation of a web page

Cleaning process

example this about web page transformation process

```
<html>
<title> Example  </title>
  This is an example about
  web page transformation
  process
</html>
```

The jth web page

... about ...

... example ...

... page ...

... process ...

... this ...

... transformation ...

... web ...

Array of words

$$m(i,j)= f_{ij} * Log(\frac{Q}{n_i})$$

... 0.5 ...

... 0.7 ...

... 0.4 ...

... 0.3 ...

... 0.1 ...

... 0.5 ...

... 0.4 ...

$m(i,j)$

Vector allows us to: Compare web pages (cosine)

TFIDF:
$$M = (m_{ij}) = f_{ij} * \log(\frac{Q}{n_i})$$

Representative web page vector i y j.

$$p_i = (m_{1i},...,m_{Ri})$$

$$p_j = (m_{1j},...,m_{Rj})$$

$$dp(p_i, p_j) = \cos\theta = \frac{\sum_{k=1}^{R} m_{ki} m_{kj}}{\sqrt{\sum_{k=1}^{R} (m_{ki})^2} \sqrt{\sum_{k=1}^{R} (m_{kj})^2}}$$

Similarity function
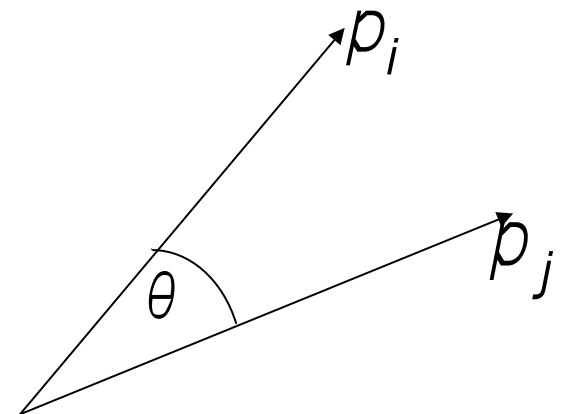
Vector Space Model: The Process (to see in details in further chapters)

- **Cleaning**: Text extraction from HTML.
- **Tokenization**: Use of syntactic rules, filtering stop words.
- **Stemming**: (Porter Algorithm
http://tartarus.org/martin/PorterStemmer/)
  translating the word to its semantic root. (ex: writing -> write)
- **Vectorization**: The calculation of the matrix M with the
Appropriate weight function.

Pros & Cons of Vector Model

- **Advantages:**
  - term–weighting improves quality of the answer set
  - partial matching allows retrieval of docs that approximate the query conditions
  - cosine ranking formula sorts documents according to degree of similarity to the query
- **Disadvantages:**
  - assumes independence of index terms; not clear if this is a good or bad assumption
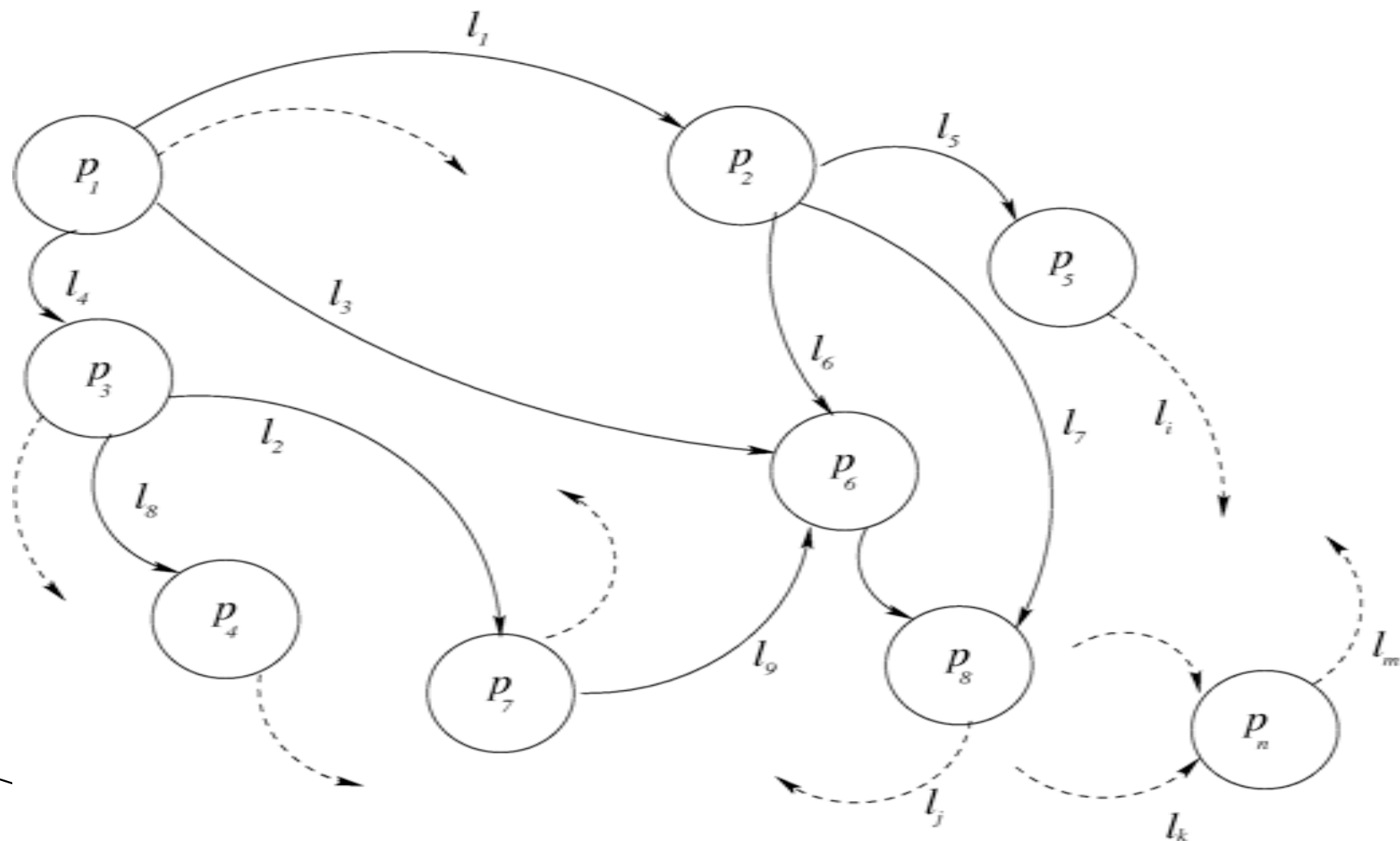
Web hyperlinks structure

»

Web hyperlinks structure

- Why a web page point to another one?

- **Link analysis:**

  - use link structure to determine credibility.

- If a web page is pointed by other ones, maybe it is because the page contains **relevant information.**

- We can **understand the formation of a web community.**

- We can **improve our web site.**

Web page links: Structure

The hyperlink: The data

- Present on HTML tag property "href" and others like the event OnClick
- ** Buy **
- Obtaining the data:
  - Crawl a web site: obtain recursively all the pages (text) that follow by hyperlink.
 - Store the relation q->p

Processing the Hyperlinks structure

- Identifying which pages contain more relevant information that others [Kleinberg99]:

  - **Authorities**. A natural information repository for the community. x will be a vector of weight for authorities.

  - **Hub**. These concentrate links to authorities web pages, for instance, "my favourite sites". y will be a vector of weight for hubs.

$$x_p = \sum_{q: \exists q \to p} y_q$$

$$y_p = \sum_{q: \exists p \to q} x_q$$

Later we will develop further in the HITS algorithm.
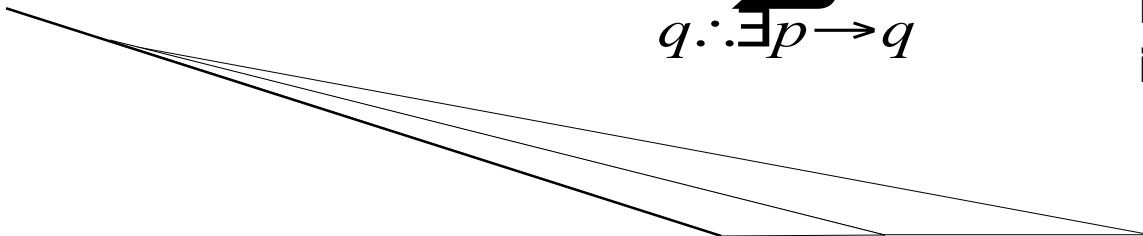
Processing the Hyperlinks structure

- **Hits & Page Rank algorithm**
  - Generate a "ranking" search result for search engine.
    - Google, Yahoo!

- **AdWord mechanism**
  - Give to Google a large amount of utility, based on this kind of structure mining.

Web Data: Summary

- Our **FOCUS** is mining an **e-business datawarehouse.**
- In order to **analyze the data** we need first to **known the process** that generate it.
  - And the **sources** of them:
- **Web Server Logs Files**
  - behaviour of the client.
- **Web Content**
  - The text that the client see.
- **Web Link**
  - The way that the client could browse the content.