On Summarization of Non-Linear Narratives

by

Shruti Gaur

A Thesis Presented in Partial Fulfillment
of the Requirements for the Degree
Master of Science

ARIZONA STATE UNIVERSITY

August 2011

ABSTRACT

Navigating within non-linear structures is a challenge for all users when the space is large but the problem is most pronounced when the users are blind or visually impaired. Such users access digital content through screen readers like JAWS which read out the text on the screen. However presentation of non-linear narratives in such a manner without visual cues and information about spatial dependencies is very inefficient for such users.

The NSDL Science Literacy StrandMaps are visual layouts to help students and teachers browse educational resources. A Strandmap shows relationships between concepts and how they build upon one another across grade levels.

NSDL Strandmaps are non-linear narratives which need to be presented to users who are blind in an effective way. A good summary of the Strandmap can give the users an idea about the concepts that are explained in it. This can help them decide whether to view the map or not. In addition, a preview-based navigation mechanism can help users decide which direction they want to take, based on a preview of upcoming content in each direction.

Given a non-linear narrative like a Strandmap which has both text and structure, and a word limit w, the goal of this thesis is to find the best way to create its summary. The following approaches are considered:

- Purely Text-based Approach using a Multi-document Text Summarizer
- Purely Structure-based Approach using PageRank
- Approaches Combining both Text and Structure
  - CUTS-Based Approach (Topic Segmentation)
  - PageRank with Content

Since no reference summaries for such structures were available, user studies were conducted to evaluate these algorithms. PageRank with Content approach performed the best.

Another important conclusion was that text and structure are intertwined in a Strandmap by design.

DEDICATION

To my family for their love and support

ACKNOWLEDGEMENTS

I take this opportunity to thank my Advisor Prof. K. Selçuk Candan for his immense help during the course of my research. I am very thankful for his insightful guidance, help and support. I would also like to express my gratitude to Prof.Hari Sundaram and Prof.Hasan Davulcu for their valuable guidance during MAISON meetings and for being on my thesis committee.

I would like to thank Xinxin, Renwei, Wei, Mijung, Mithila, Parth, Yan and Huiping who have been my friends and colleagues at EMITlab and have always helped and supported me during my research. I am also thankful to my friends and roommates Kanika, Sravani, Shivani, Vivek, Leena, Pranati, Madhurima, Preetika, Rohit, Manish, Hardik and Tejas for their help, understanding and encouragement and for being there during difficult times. I would like to thank Mike and all the people at HRC for their help and support. I am also very thankful to all the people who participated in my user studies thus making the evaluation of my research possible.

Finally I would like to thank my parents and family for always believing in me.

TABLE OF CONTENTS

Page

TABLE OF CONTENTS v

LIST OF TABLES ix

LIST OF FIGURES x

CHAPTER 1

1 INTRODUCTION 1

1.1 Motivation 1

1.2 NSDL Strandmaps 2

1.3 MAISON 4

1.4 Problem Statement 9

1.5 Organization of this Thesis 11

2 RELATED WORK 12

2.1 Text Summarization 12

Using Shallow Lexical Features 12

Modeling units of text 13

2.2 Graph Summarization 14

Finding Communities 14

Hierarchical Clustering 15

Max-flow Min-cut 17

Graph partitioning 17

Finding Bipartite Cores 18

Other Methods 19

Clustering 19

Graph Partitioning(Multilevel algorithms) 20

Structural Clustering 21

Simulated Annealing 22

Graph Compression 23

Visualization Techniques 25

Hierarchical Graph Partitioning 25

Graph layout 25

Chapter Page

Visualization of Features of a Graph 25

Web Navigation Systems 27

Information-Theoretic Approaches 27

Database Type approaches 29

Graph Anonymization 30

3 BACKGROUND 31

3.1 MEAD Summarizer 31

Feature Extraction 31

Centroid 31

Centroid Value 32

Positional Value 33

First Sentence Overlap 33

Sentence Length 33

Sentence Scoring 33

Re-Ranking 33

3.2 Multidimensional Scaling 34

3.3 PageRank 34

3.4 CUTS 36

Representation of Stream 36

Curve Construction 37

Curve Segmentation 38

Eliminating Over-Segmentation 40

Topic Development Patterns 40

3.5 Text Summarization Using CUTS 41

4 APPROACHES TO SUMMARIZING NON-LINEAR NARRATIVES 43

4.1 Representative Algorithms 43

4.2 Purely Text-Based Approach 43

4.3 Purely Structure-Based Approach - PageRank (without Content) 43

Uniform Connectivity 44

Biased Connectivity 45

Chapter Page
4.4 Combining Text and Structure . . . 45
Approach I : Topic-development Curve Analysis . . . 45
Constrained Multidimensional Scaling (CMDS) . . . 47
Summarization of Strandmaps with CUTS . . . 50
Approach II: PageRank With Content . . . 51
5 EVALUATION . . . 54
5.1 Evaluation Challenges . . . 54
5.2 User Study Design . . . 54
5.3 User Study . . . 56
5.4 Evaluation Measure . . . 57
6 RESULTS . . . 59
6.1 User Ratings . . . 59
6.2 User Agreement . . . 59
6.3 Experimental Setup . . . 59
6.4 Comparison among Approaches . . . 61
6.5 Graph Rewiring . . . 62
6.6 Conjectures . . . 63
Interdependence between Text and Structure in Strandmaps . . . 63
PageRank with Content . . . 64
Difference in Performance across Strandmaps . . . 64
7 CONCLUSION . . . 65
7.1 Future Work . . . 66
REFERENCES . . . 67
APPENDIX . . . 73
A USER STUDIES . . . 73
A.1 User Study Version 1 . . . 73
Lessons Learnt . . . 73
A.2 User Study Version 2 . . . 73
Lessons Learnt . . . 73
A.3 User Study Version 3 . . . 76

Chapter Page

Lessons Learnt 77

B SUMMARIZATION IN MAISON 78

B.1 Summary Preview for Navigation in NSDL StrandMaps 78

B.2 MAISON Web Annotation Plugin 78

LIST OF TABLES

Table Page

5.1 Strandmaps for Evaluation 56

6.1 Distribution of User Ratings 59

6.2 P-Values from ANOVA test for User Agreement 59

6.3 Different Configurations for CUTS-based Approach 60

6.4 Comparison between Approaches 61

6.5 Effect of Graph Rewiring 63

LIST OF FIGURES

Figure Page

1.1 NSDL Strandmap - Stars [1] 2

1.2 Features of an NSDL Strandmap 3

1.3 Strandmap Search Interface in NSDL 5

1.4 Search Results on NSDL site 6

1.5 MAISON Search Interface 7

1.6 Search Results on MAISON site 7

1.7 Creating a Summary Annotation 8

1.8 Navigating Strandmaps through Summary Annotations 8

1.9 Creating a Keyword Annotation 9

1.10 Navigating Strandmaps through Keyword Annotations 10

1.11 MAISON Web Annotations 10

2.1 Community Structure in a Graph 14

2.2 Dendogram [67] 15

2.3 Focused community crawling and the graph induced (a)The virtual source vertex (b)Vertices of seed websites (c)Vertices of websites one link away from any seed site (d)References to sites not in b or c and (e)The virtual sink vertex [28] 17

2.4 Bipartite Core [40] 18

2.5 Graph Partitioning [52] 20

2.6 Overview of Graph Layout Algorithms [32] 26

2.7 Logical Neighborhoods and the Summarization Process. Crossed circles denote entry points of neighborhoods. Each sub-neighborhood contains one entry point per its parent neighborhoods. Each parent neighborhood includes the entry points of its sub-neighborhoods.[19] 28

2.8 Graph Summarization by Aggregation [70] 29

3.1 Centroid Example 32

3.2 A Small Network 35

3.3 CUTS Curve [56] 38

3.4 Topic Development Patterns [56] 38

3.5 Adaptive Curve Segmentation [45] 39

Figure Page
3.6 Combining Curve Segments [56] 40
4.1 Sentence Graphs 46
4.2 Pagerank Example 52
4.3 Pagerank with Content Example 53
5.1 User Study Interface 57
6.1 Normalized User Overlap Graph 62
A.1 User Study Version 1 74
A.2 User Study Version 2 75
A.3 User Study Version 3 Screen 1 76
A.4 User Study Version 3 Screen 2 76
B.1 MAISON Web Annotation System 78

Chapter 1

INTRODUCTION

1.1 Motivation

Users who are blind access digital content through Screen Readers like JAWS [2], Window-Eyes [3] and Dolphin [4] which read out the text on the screen. However since Screen Readers present the information in a linear manner, such users cannot skim the page quickly and might potentially have to cover a large portion of the page before reaching the content they are looking for. Moreover individuals who are blind do not get information about spatial dependencies between elements on the page and other visual cues which help their sighted peers decide which parts of a page are more important than the others.

There have been efforts to make structural and navigational information accessible to users who are blind. A transformer system [37] extracts all the links and titles in the page and makes them available at the top of the page to make it easier to access links. It also adds textual tags for html elements like Images, Buttons, Radiobuttons etc. and links on the page.

For information represented in the form of Concept maps, Website maps, Food-webs, E-R diagrams etc, the content itself is visually rich and non-linear.To present such information using screen readers would be a huge challenge. Even if the users can listen to the text contained in such structures and their relationships efficiently, keeping everything in memory and making sense of it would be a cognitive overload. Since individuals who are blind tend to construct linear memory structures [29], it would be more difficult for them to comprehend such structures as a whole.

Our goal is to make such non-linear narratives accessible to individuals who are blind. This thesis focuses on exploring ways to create summaries of such structures which can then be used to enhance search and navigation for users who are blind.

Hovy and Lin [33] define a summary as "*a text that is produced out of one or more(possibly multimedia) texts, that contains(some of) the same information of the original text(s), and that is no longer than half of the original text(s).*" In this thesis we seek

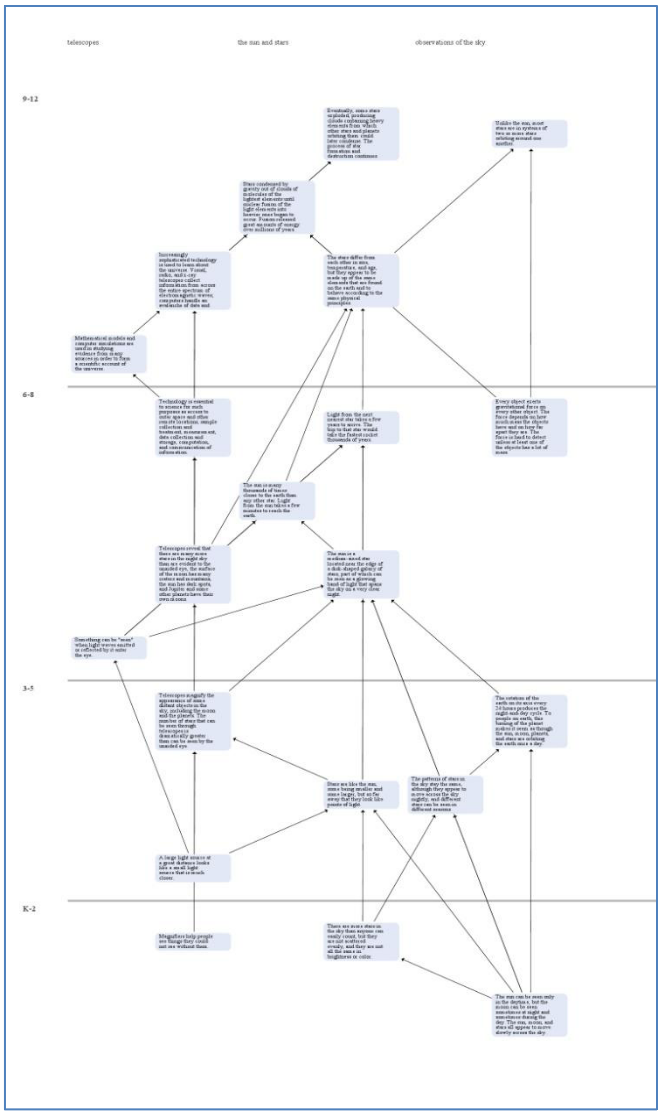

telescopes
the sun and stars
observations of the sky
9-12
6-8
3-5
K-2
Eventually, some stars exploded, producing clouds containing heavy elements from which other stars and planets orbiting them could later condense. The process of star formation and destruction continues.
Unlike the sun, most stars are in systems of two or more stars orbiting around one another.
Stars condensed by gravity out of clouds of molecules of the lightest elements until nuclear fusion of the light elements into heavier ones began to occur. Fusion released great amounts of energy over millions of years.
Increasingly sophisticated technology is used to learn about the universe. Visual, radio, and x-ray telescopes collect information from across the entire spectrum of electromagnetic waves; computers handle an avalanche of data and
The stars differ from each other in size, temperature, and age, but they appear to be made up of the same elements that are found on the earth and to behave according to the same physical principles.
Mathematical models and computer simulations are used in studying evidence from many sources in order to form a scientific account of the universe.
Technology is essential to science for such purposes as access to outer space and other remote locations, sample collection and treatment, measurement, data collection and storage, computation, and communication of information.
Light from the next nearest star takes a few years to arrive. The trip to that star would take the fastest rocket thousands of years.
Every object exerts gravitational force on every other object. The force depends on how much mass the objects have and on how far apart they are. The force is hard to detect unless at least one of the objects has a lot of mass.
The sun is many thousands of times closer to the earth than any other star. Light from the sun takes a few minutes to reach the earth.
Telescopes reveal that there are many more stars in the night sky than are evident to the unaided eye, the surface of the moon has many craters and mountains, the sun has dark spots, and Jupiter and some other planets have their own moons.
The sun is a medium-sized star located near the edge of a disk-shaped galaxy of stars, part of which can be seen as a glowing band of light that spans the sky on a very clear night.
Something can be "seen" when light waves emitted or reflected by it enter the eye.
Telescopes magnify the appearance of some distant objects in the sky, including the moon and the planets. The number of stars that can be seen through telescopes is dramatically greater than can be seen by the unaided eye.
The rotation of the earth on its axis every 24 hours produces the night-and-day cycle. To people on earth, this turning of the planet makes it seem as though the sun, moon, planets, and stars are orbiting the earth once a day.
Stars are like the sun, some being smaller and some larger, but so far away that they look like points of light.
The patterns of stars in the sky stay the same, although they appear to move across the sky nightly, and different stars can be seen in different seasons.
A large light source at a great distance looks like a small light source that is much closer.
Magnifiers help people see things they could not see without them.
There are more stars in the sky than anyone can easily count, but they are not scattered evenly, and they are not all the same in brightness or color.
The sun can be seen only in the daytime, but the moon can be seen sometimes at night and sometimes during the day. The sun, moon, and stars all appear to move slowly across the sky.

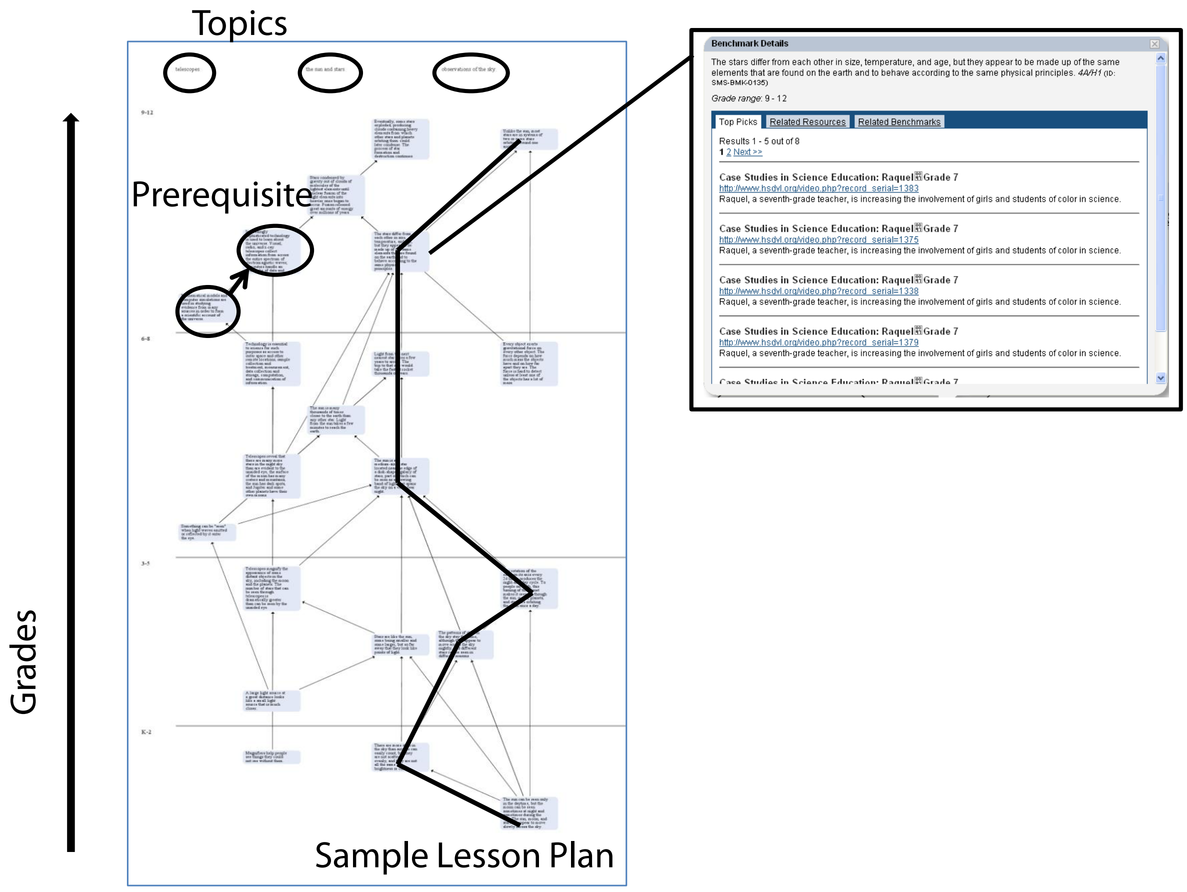
Figure 1.1: NSDL Strandmap - Stars [1]

to find approaches to create efficient summaries of Strandmaps.

1.2 NSDL Strandmaps

National Science Digital Library(NSDL) is an online library supported by the National Science Foundation. It contains resources for science, technology, engineering, and mathematics(STEM) education and research. These resources are selected by specialists from high quality scientific and technical material and cover the curriculum from K through 12.

The NSDL Science Literacy StrandMaps [1] are visual layouts which serve as



Benchmark Details

The stars differ from each other in size, temperature, and age, but they appear to be made up of the same elements that are found on the earth and to behave according to the same physical principles. *4A/H1* (ID: SMS-BMK-0135)

Grade range: 9 - 12

Top Picks | Related Resources | Related Benchmarks

Results 1 - 5 out of 8
1 2 Next >>

Case Studies in Science Education: Raquel Grade 7
http://www.hsdvl.org/video.php?record_serial=1383
Raquel, a seventh-grade teacher, is increasing the involvement of girls and students of color in science.

Case Studies in Science Education: Raquel Grade 7
http://www.hsdvl.org/video.php?record_serial=1375
Raquel, a seventh-grade teacher, is increasing the involvement of girls and students of color in science.

Case Studies in Science Education: Raquel Grade 7
http://www.hsdvl.org/video.php?record_serial=1338
Raquel, a seventh-grade teacher, is increasing the involvement of girls and students of color in science.

Case Studies in Science Education: Raquel Grade 7
http://www.hsdvl.org/video.php?record_serial=1379
Raquel, a seventh-grade teacher, is increasing the involvement of girls and students of color in science.

Figure 1.2: Features of an NSDL Strandmap

primary interfaces to help students and teachers browse NSDL educational resources. A Strandmap about a specific science or math topic shows relationships between concepts belonging to the topic and how they build upon one another across grade levels. A Strandmap for "Stars" is shown as an example in Figure 1.1 and Figure 1.2 highlights its important features.

- Each box describes a concept using a set of benchmark statements defined by AAAS Project 2061 [55]. Clicking on a box shows the NSDL resources relevant to the concept and information about related AAAS Project 2061 Benchmarks and National Science Education Standards.

- Grade levels increase from bottom to top thus implying an increase in complexity of concepts on moving upwards.

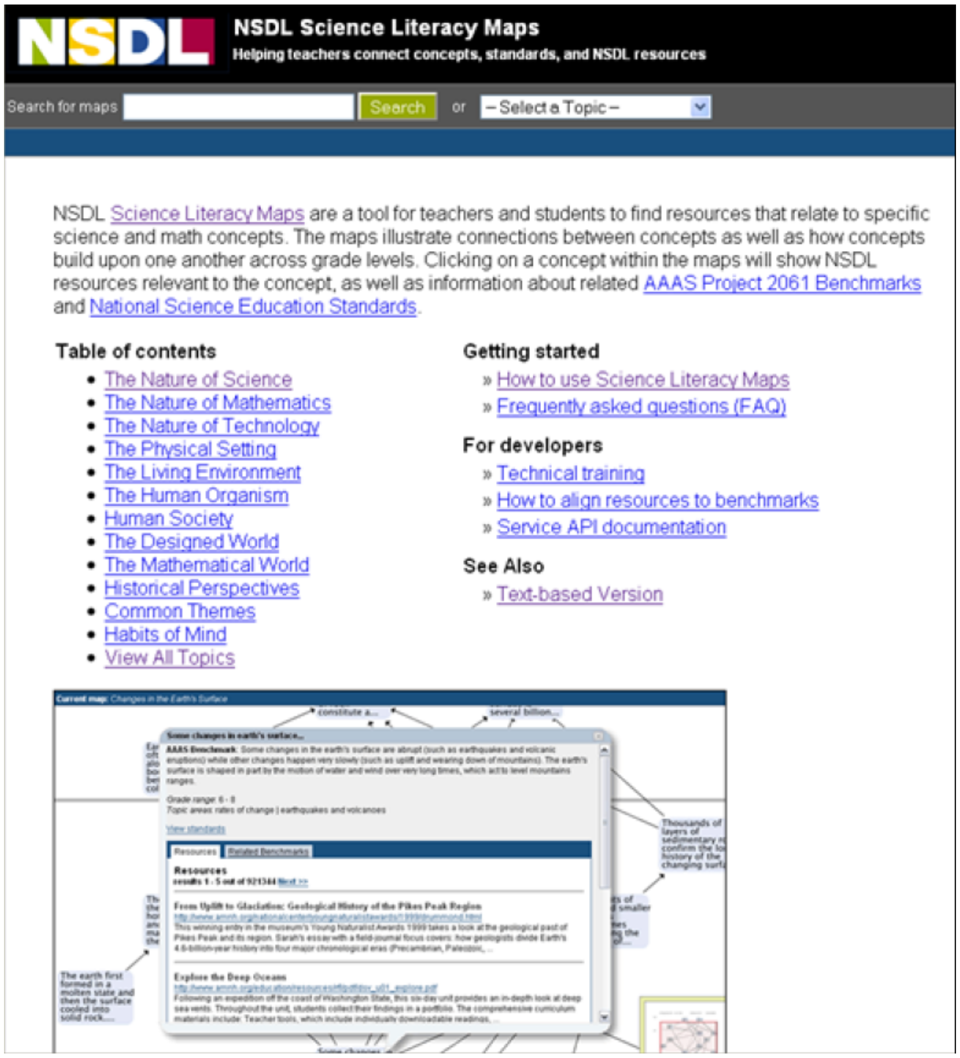
- Groups of concepts, called topics or strands are listed on the top of the map. These are the main topics being talked about in the map.
- The arrows between the concepts depict how concepts build upon each other.Their direction signifies a "Prerequisite" relationship which means that knowing the concept from where the arrow starts is required to be able to understand the concept where the arrow points to.

Strandmaps can be thought of as directed graphs in which concepts are nodes and edges between them determine the order of learning. Grades increase from bottom to top thus dividing the Strandmap into Grade Levels horizontally. Topic strands vary vertically. Thus teachers can design lesson plans which start from one concept and follow arrows along the path leading to another concept. Clicking on each node along this lesson plan path will give them the necessary information and resources to teach that concept. Students can also use the Strandmap to find resources included in the curriculum for their grade-level.

1.3 MAISON

MAISON: Middleware for Accessible Information Spaces on NSDL [5, 12] is a middleware service to help minimize the extraneous load on teachers and students who are blind, while they search and access materials from the collection of resources available on NSDL. Such users interact through screen readers such as JAWS [2], Window-Eyes [3] and Dolphin [4].

MAISON provides an accessible text-based interface to NSDL Strandmaps. The middleware is built on top of the Strand map service (SMS), which can be accessed through a Concept Space Interchange Protocol (CSIP) service API [6] provided by NSDL. MAISON middleware provides a new CSIP-A protocol that enables development of user interfaces that provide contextually informed strand map search and navigation.In addition to specifying the keyword to search, a context keyword can also be specified which further refines search results. Figure 1.3 shows the NSDL interface for searching Strandmaps. Search results are displayed by the interface shown in Figure 1.4. The MAISON interface


NSDL
NSDL Science Literacy Maps
Helping teachers connect concepts, standards, and NSDL resources
Search for maps
Search
or
– Select a Topic –

NSDL Science Literacy Maps are a tool for teachers and students to find resources that relate to specific science and math concepts. The maps illustrate connections between concepts as well as how concepts build upon one another across grade levels. Clicking on a concept within the maps will show NSDL resources relevant to the concept, as well as information about related AAAS Project 2061 Benchmarks and National Science Education Standards.

Table of contents

- The Nature of Science
- The Nature of Mathematics
- The Nature of Technology
- The Physical Setting
- The Living Environment
- The Human Organism
- Human Society
- The Designed World
- The Mathematical World
- Historical Perspectives
- Common Themes
- Habits of Mind
- View All Topics

Getting started

» How to use Science Literacy Maps
» Frequently asked questions (FAQ)

For developers

» Technical training
» How to align resources to benchmarks
» Service API documentation

See Also

» Text-based Version

Current map: Changes in the Earth's Surface

Some changes in earth's surface...

AAAS Benchmark: Some changes in the earth's surface are abrupt (such as earthquakes and volcanic eruptions) while other changes happen very slowly (such as uplift and wearing down of mountains). The earth's surface is shaped in part by the motion of water and wind over very long times, which act to level mountains ranges.

Grade range: 6 - 8
Topic areas: rates of change | earthquakes and volcanoes

View standards

Resources | Related Benchmarks

Resources
results 1 - 5 out of 921344 Next >>

From Uplift to Glaciation: Geological History of the Pikes Peak Region
http://www.amnh.org/nationalcenter/youngnaturalistawards/1999/drummond.html
This winning entry in the museum's Young Naturalist Awards 1999 takes a look at the geological past of Pikes Peak and its region. Sarah's essay with a field-journal focus covers: how geologists divide Earth's 4.6-billion-year history into four major chronological eras (Precambrian, Paleozoic, ...

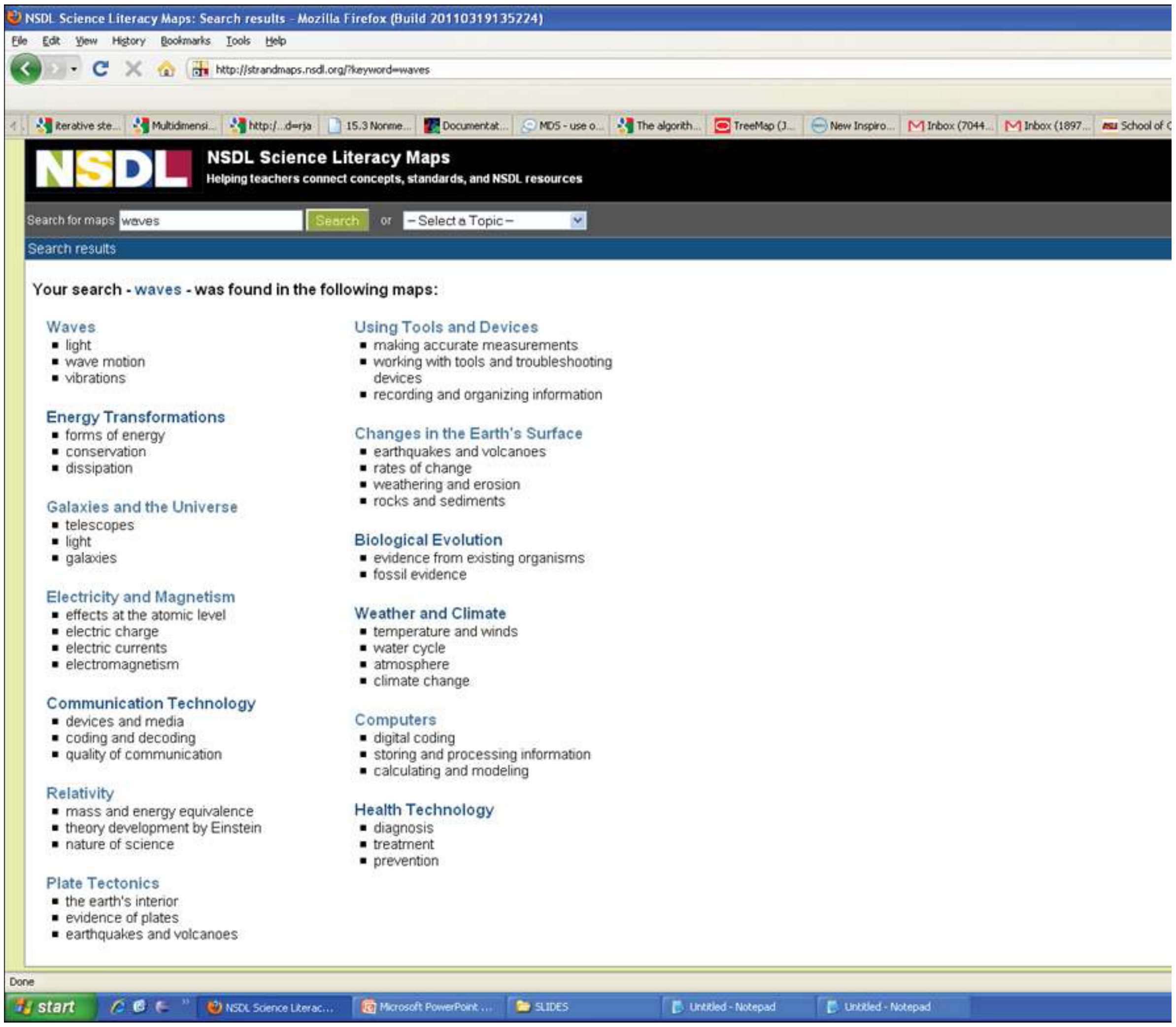
Explore the Deep Oceans
http://www.amnh.org/education/resources/rfl/pdf/dsv_u01_explore.pdf
Following an expedition off the coast of Washington State, this six-day unit provides an in-depth look at deep sea vents. Throughout the unit, students collect their findings in a portfolio. The comprehensive curriculum materials include: Teacher tools, which include individually downloadable readings, ...

constitute a...
several billion...
Thousands of layers of sedimentary r confirm the lo history of the changing surfa
The earth first formed in a molten state and then the surface cooled into solid rock....
Some changes...

Figure 1.3: Strandmap Search Interface in NSDL

for searching these Strandmaps is shown in Figure 1.5 and the search results are shown in Figure 1.6. MAISON interfaces are easily accessible by screen readers and customizable by the user. Search can be enhanced using context keywords. There is an option to organize Strandmap nodes based on grades or strands which can give a clustered map focused on either of these properties.In addition to the above features, MAISON provides a personalized experience by providing options for bookmarking and history navigation for the user.

The NSDL site shows the Strandmap to users in the form of an image. Since such an element is inaccessible by screen-readers, we created an accessible interface for


NSDL Science Literacy Maps: Search results - Mozilla Firefox (Build 20110319135224)
File Edit View History Bookmarks Tools Help
http://strandmaps.nsdl.org/?keyword=waves
NSDL
NSDL Science Literacy Maps
Helping teachers connect concepts, standards, and NSDL resources
Search for maps
waves
Search
or
-- Select a Topic --
Search results
Done
start

Your search - waves - was found in the following maps:

Waves
- light
- wave motion
- vibrations

Energy Transformations
- forms of energy
- conservation
- dissipation

Galaxies and the Universe
- telescopes
- light
- galaxies

Electricity and Magnetism
- effects at the atomic level
- electric charge
- electric currents
- electromagnetism

Communication Technology
- devices and media
- coding and decoding
- quality of communication

Relativity
- mass and energy equivalence
- theory development by Einstein
- nature of science

Plate Tectonics
- the earth's interior
- evidence of plates
- earthquakes and volcanoes

Using Tools and Devices
- making accurate measurements
- working with tools and troubleshooting devices
- recording and organizing information

Changes in the Earth's Surface
- earthquakes and volcanoes
- rates of change
- weathering and erosion
- rocks and sediments

Biological Evolution
- evidence from existing organisms
- fossil evidence

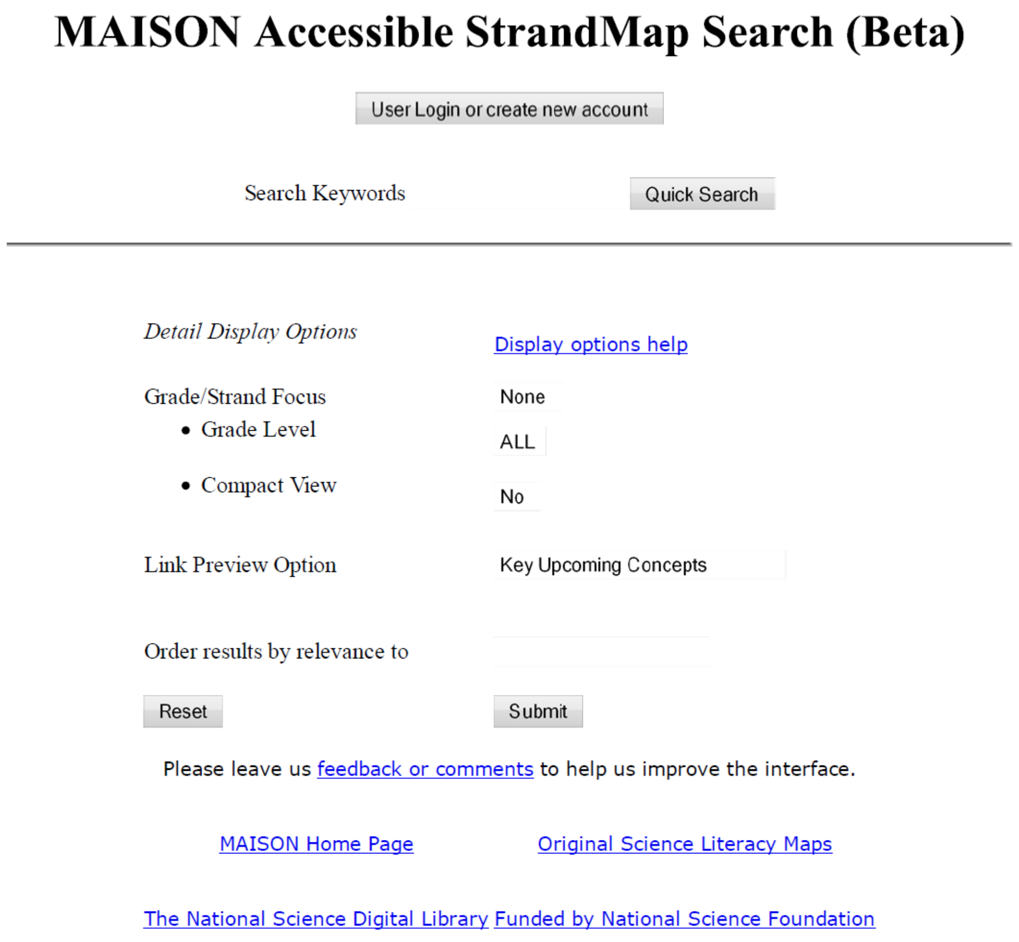
Weather and Climate
- temperature and winds
- water cycle
- atmosphere
- climate change

Computers
- digital coding
- storing and processing information
- calculating and modeling

Health Technology
- diagnosis
- treatment
- prevention

Figure 1.4: Search Results on NSDL site

navigating within the Strandmap for users who are blind. Starting with the nodes(concepts) at the lowest level, there are link annotations for each possible path which give a preview of what kind of concepts are reachable on that path. Link annotations to aid navigation can be of three types: summary of upcoming nodes, keywords from upcoming concepts or simply the text descriptions. The summary annotation on an edge presents a text summary of the contents of the nodes reachable from that edge within a certain distance. This is further explained in Figure 1.7. A snapshot of the Strandmap annotated this way is shown in Figure 1.8. Similarly for keyword annotations, the most important keywords are extracted from the upcoming nodes and included in the annotation of the edge leading to them. As illustrated in Figure 1.9, the keywords or summary



MAISON Accessible StrandMap Search (Beta)

User Login or create new account

Search Keywords

Quick Search

Detail Display Options

Display options help

Grade/Strand Focus	None
• Grade Level	ALL
• Compact View	No
Link Preview Option	Key Upcoming Concepts
Order results by relevance to	

Reset

Submit

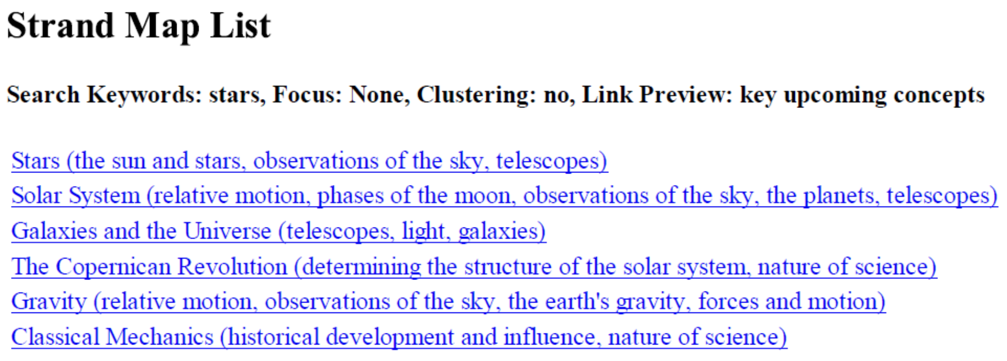
Please leave us feedback or comments to help us improve the interface.

MAISON Home Page

Original Science Literacy Maps

The National Science Digital Library Funded by National Science Foundation

Figure 1.5: MAISON Search Interface



Strand Map List

Search Keywords: stars, Focus: None, Clustering: no, Link Preview: key upcoming concepts

Stars (the sun and stars, observations of the sky, telescopes)
Solar System (relative motion, phases of the moon, observations of the sky, the planets, telescopes)
Galaxies and the Universe (telescopes, light, galaxies)
The Copernican Revolution (determining the structure of the solar system, nature of science)
Gravity (relative motion, observations of the sky, the earth's gravity, forces and motion)
Classical Mechanics (historical development and influence, nature of science)

Figure 1.6: Search Results on MAISON site

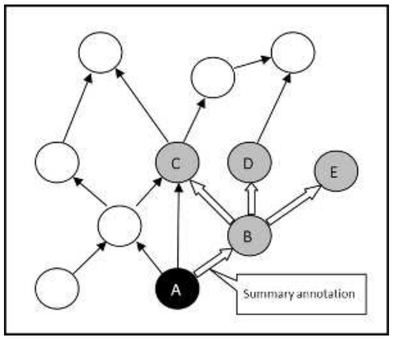
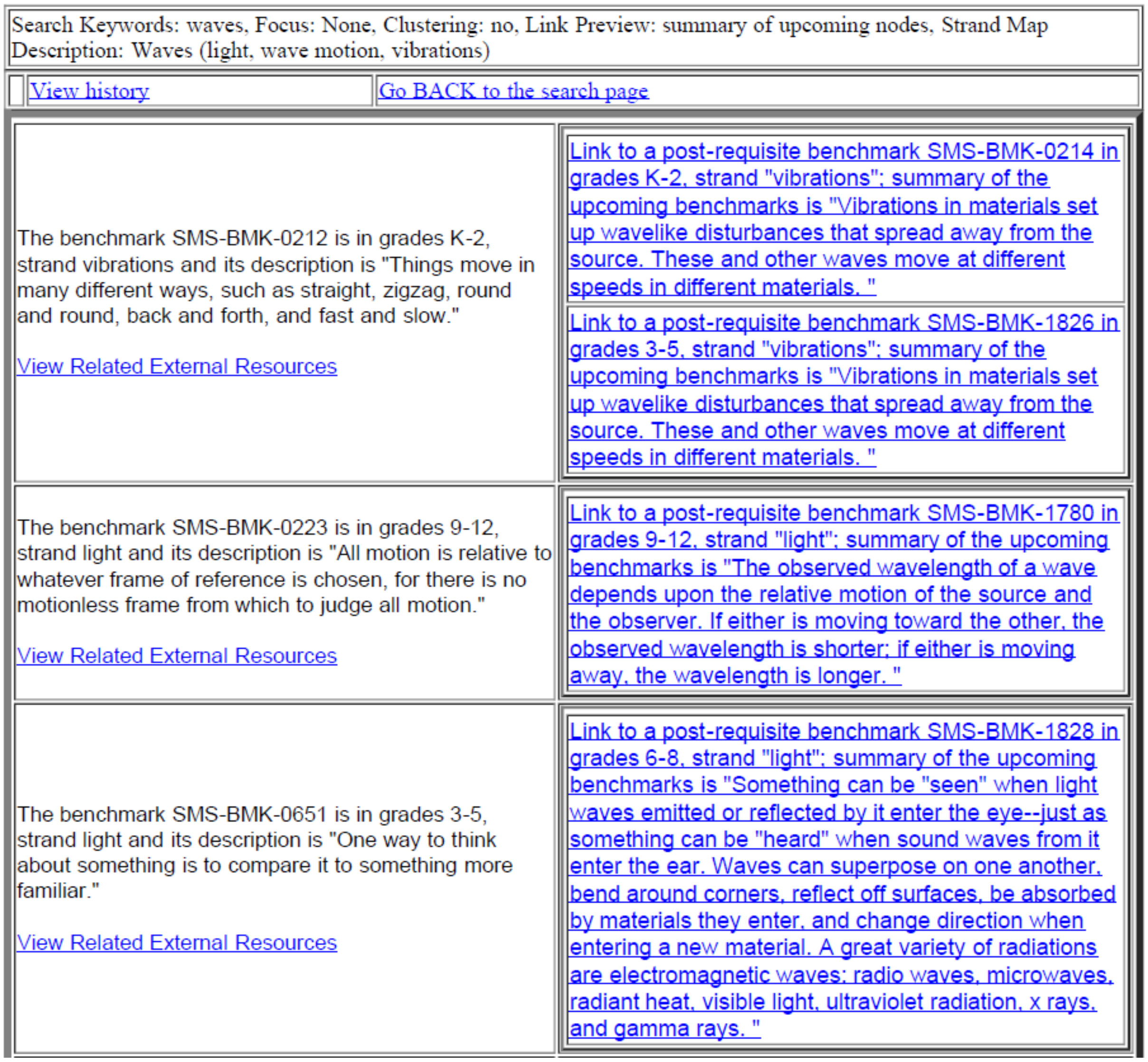


Figure 1.7: Creating a Summary Annotation



Search Keywords: waves, Focus: None, Clustering: no, Link Preview: summary of upcoming nodes, Strand Map Description: Waves (light, wave motion, vibrations)

View history | Go BACK to the search page

The benchmark SMS-BMK-0212 is in grades K-2, strand vibrations and its description is "Things move in many different ways, such as straight, zigzag, round and round, back and forth, and fast and slow." View Related External Resources	Link to a post-requisite benchmark SMS-BMK-0214 in grades K-2, strand "vibrations"; summary of the upcoming benchmarks is "Vibrations in materials set up wavelike disturbances that spread away from the source. These and other waves move at different speeds in different materials. " Link to a post-requisite benchmark SMS-BMK-1826 in grades 3-5, strand "vibrations"; summary of the upcoming benchmarks is "Vibrations in materials set up wavelike disturbances that spread away from the source. These and other waves move at different speeds in different materials. "
The benchmark SMS-BMK-0223 is in grades 9-12, strand light and its description is "All motion is relative to whatever frame of reference is chosen, for there is no motionless frame from which to judge all motion." View Related External Resources	Link to a post-requisite benchmark SMS-BMK-1780 in grades 9-12, strand "light"; summary of the upcoming benchmarks is "The observed wavelength of a wave depends upon the relative motion of the source and the observer. If either is moving toward the other, the observed wavelength is shorter; if either is moving away, the wavelength is longer. "
The benchmark SMS-BMK-0651 is in grades 3-5, strand light and its description is "One way to think about something is to compare it to something more familiar." View Related External Resources	Link to a post-requisite benchmark SMS-BMK-1828 in grades 6-8, strand "light"; summary of the upcoming benchmarks is "Something can be "seen" when light waves emitted or reflected by it enter the eye--just as something can be "heard" when sound waves from it enter the ear. Waves can superpose on one another, bend around corners, reflect off surfaces, be absorbed by materials they enter, and change direction when entering a new material. A great variety of radiations are electromagnetic waves: radio waves, microwaves, radiant heat, visible light, ultraviolet radiation, x rays, and gamma rays. "

Figure 1.8: Navigating Strandmaps through Summary Annotations

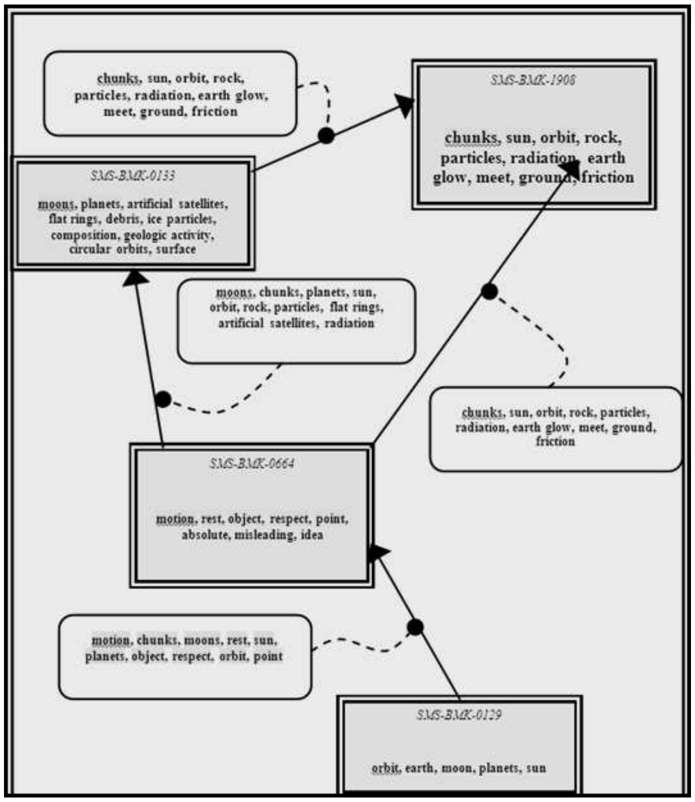

chunks, sun, orbit, rock, particles, radiation, earth glow, meet, ground, friction
SMS-BMK-1908
chunks, sun, orbit, rock, particles, radiation, earth glow, meet, ground, friction
SMS-BMK-0133
moons, planets, artificial satellites, flat rings, debris, ice particles, composition, geologic activity, circular orbits, surface
moons, chunks, planets, sun, orbit, rock, particles, flat rings, artificial satellites, radiation
chunks, sun, orbit, rock, particles, radiation, earth glow, meet, ground, friction
SMS-BMK-0664
motion, rest, object, respect, point, absolute, misleading, idea
motion, chunks, moons, rest, sun, planets, object, respect, orbit, point
SMS-BMK-0129
orbit, earth, moon, planets, sun

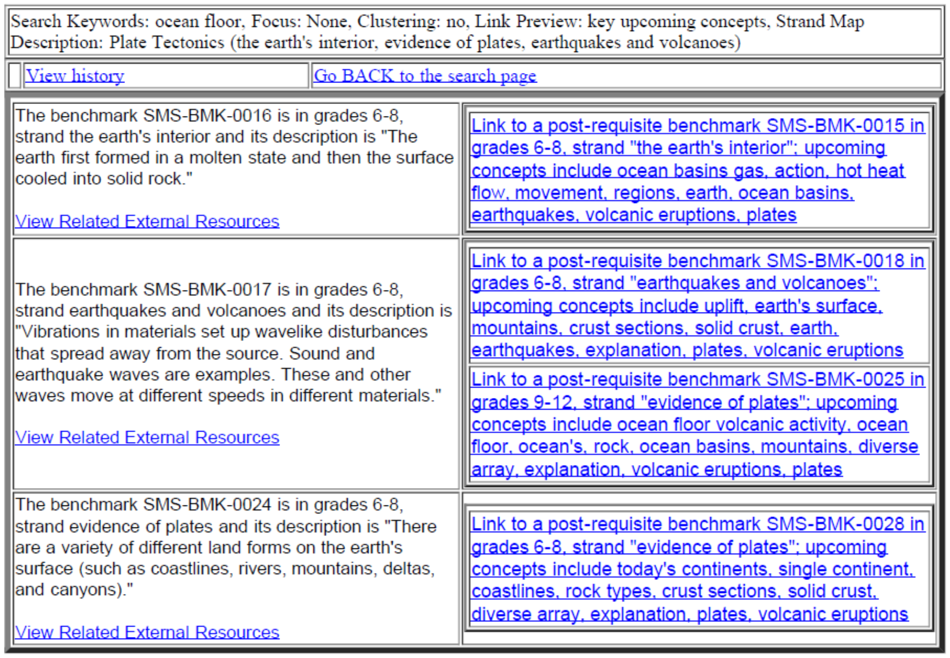
Figure 1.9: Creating a Keyword Annotation

annotations are not only generated from the immediate neighbors but also from nodes that are present further along that path, if they are deemed important enough. The screenshot in Figure 1.10 shows how keywords are used to annotate edges of a Strandmap. More implementation details can be found in Appendix B.1.

MAISON also provides a Web Annotation Plugin which annotates the links in a webpage to help people who are blind have a better navigation experience by getting a preview of the content to which the link points. The plugin provides either keyword or text summary annotations on pressing hotkey combinations from the keyboard. Figure 1.11 shows a sample link and the keyword and text annotations generated by the plugin for it. Implementation details can be found in the Appendix B.2.

1.4 Problem Statement

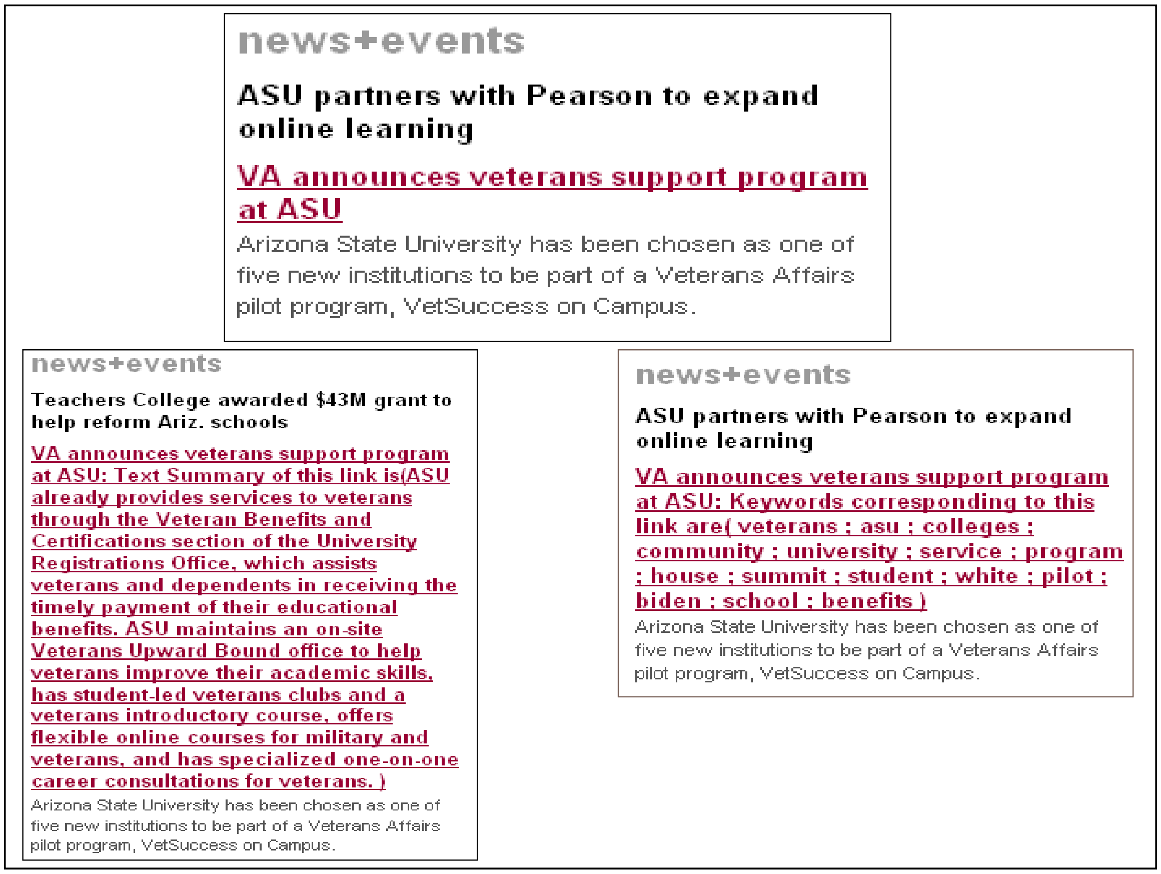
Navigating within non-linear narratives is a challenge for all users when the space is large but the problem is most pronounced when the users are blind or visually impaired.



Search Keywords: ocean floor, Focus: None, Clustering: no, Link Preview: key upcoming concepts, Strand Map Description: Plate Tectonics (the earth's interior, evidence of plates, earthquakes and volcanoes)

View history	Go BACK to the search page
The benchmark SMS-BMK-0016 is in grades 6-8, strand the earth's interior and its description is "The earth first formed in a molten state and then the surface cooled into solid rock." View Related External Resources	Link to a post-requisite benchmark SMS-BMK-0015 in grades 6-8, strand "the earth's interior"; upcoming concepts include ocean basins gas, action, hot heat flow, movement, regions, earth, ocean basins, earthquakes, volcanic eruptions, plates
The benchmark SMS-BMK-0017 is in grades 6-8, strand earthquakes and volcanoes and its description is "Vibrations in materials set up wavelike disturbances that spread away from the source. Sound and earthquake waves are examples. These and other waves move at different speeds in different materials." View Related External Resources	Link to a post-requisite benchmark SMS-BMK-0018 in grades 6-8, strand "earthquakes and volcanoes"; upcoming concepts include uplift, earth's surface, mountains, crust sections, solid crust, earth, earthquakes, explanation, plates, volcanic eruptions Link to a post-requisite benchmark SMS-BMK-0025 in grades 9-12, strand "evidence of plates"; upcoming concepts include ocean floor volcanic activity, ocean floor, ocean's, rock, ocean basins, mountains, diverse array, explanation, volcanic eruptions, plates
The benchmark SMS-BMK-0024 is in grades 6-8, strand evidence of plates and its description is "There are a variety of different land forms on the earth's surface (such as coastlines, rivers, mountains, deltas, and canyons)." View Related External Resources	Link to a post-requisite benchmark SMS-BMK-0028 in grades 6-8, strand "evidence of plates"; upcoming concepts include today's continents, single continent, coastlines, rock types, crust sections, solid crust, diverse array, explanation, plates, volcanic eruptions

Figure 1.10: Navigating Strandmaps through Keyword Annotations



news+events

ASU partners with Pearson to expand online learning

VA announces veterans support program at ASU

Arizona State University has been chosen as one of five new institutions to be part of a Veterans Affairs pilot program, VetSuccess on Campus.

news+events

Teachers College awarded $43M grant to help reform Ariz. schools

VA announces veterans support program at ASU: Text Summary of this link is(ASU already provides services to veterans through the Veteran Benefits and Certifications section of the University Registrations Office, which assists veterans and dependents in receiving the timely payment of their educational benefits. ASU maintains an on-site Veterans Upward Bound office to help veterans improve their academic skills, has student-led veterans clubs and a veterans introductory course, offers flexible online courses for military and veterans, and has specialized one-on-one career consultations for veterans.)

Arizona State University has been chosen as one of five new institutions to be part of a Veterans Affairs pilot program, VetSuccess on Campus.

news+events

ASU partners with Pearson to expand online learning

VA announces veterans support program at ASU: Keywords corresponding to this link are(veterans ; asu ; colleges ; community ; university ; service ; program ; house ; summit ; student ; white ; pilot ; biden ; school ; benefits)

Arizona State University has been chosen as one of five new institutions to be part of a Veterans Affairs pilot program, VetSuccess on Campus.

Figure 1.11: MAISON Web Annotations

NSDL Strandmaps are non-linear narratives which need to be presented to users who are blind, in an effective way. A good summary of the Strandmap can give the users an idea about the concepts that are explained in that map and help them decide whether they want to view the map or not. In addition, a preview-based navigation mechanism can help users decide which path they want to take from a certain node, based on a preview of nodes further reachable from that path.

Given a non-linear narrative like a Strandmap which has both text and structure, and a word limit w, our goal is to find the best way to create its summary.

There are several approaches which can be used to create a summary of a Strandmap.

- **Purely Text-Based Approach:** A text-only approach can be used to summarize the textual content of the nodes in the Strandmap without giving any weight to their structural relationships.
- **Purely Structure-Based Approach:** A purely structure-based approach will summarize the Strandmap based on the link-based importance of each node without taking content into account.
- **Approaches Combining Text and Structure:** Purely text-based and purely structure-based approaches are two ends of the spectrum and we can find effective ways of combining both together to create a summary.

1.5 Organization of this Thesis

Chapter 2 contains a summary of techniques for text and graph summarization. Chapter 3 describes the MEAD Summarizer [60], the CUTS [56] algorithm for segmentation of text streams, the method of Multidimensional Scaling [71] and PageRank [17, 53]. Chapter 4 contains representative algorithms from the approaches we discussed in Section 1.4 to summarize Strandmaps. Chapter 5 describes the methods to evaluate summaries generated by various approaches and Chapter 6 contains results and analysis. The Appendix contains all versions of user studies that were conducted for evaluation and summarization in MAISON [5].

Chapter 2

RELATED WORK

2.1 Text Summarization

Text Summarization techniques strive to obtain a concise piece of text that is representative and contains the same information as the original text.

Summaries are of two kinds - Abstracts and Extracts. An Abstract is a new set of sentences constructed after semantic analysis of the input text. This is the way humans naturally create summaries. An Extract on the other hand simply picks up parts of the text it deems most important to convey the meaning of the input text using some heuristics.

The techniques used to achieve extraction-based summaries can be broadly classified as follows.

Using Shallow Lexical Features

These techniques basically use features of the text like - word frequency, location of the word, sentence length and cue words as important criteria to determine the importance of a sentence. Then they rank sentences on the basis of these features or their combination and pick up the highest ranked sentences from them.

One of the earliest work in text summarization using features was done by Luhn [46]. It first finds out the significance of individual words in a text. The significance of a sentence is a combination of the significance of words in it and their positions within it. This is based on the intuition that highly significant words at a close distance will be most representative of the text. Yih et al [68] score each word by a combination of its frequency and position in the document, favoring words that occur near the beginning. These features are combined using an ML based technique and sentence selection is treated as an optimization problem.

Teufel and Moens [69] mainly use positive and negative cue words to determine abstract-worthy material, in addition to using location, sentence length and word frequency. Hovy and Lin [33] also use cue words.

Modeling units of text

These techniques model the units of text-words, sentences and paragraphs, and the similarities between them, to generate summaries.

Mitra et al [47] create a text relationship map in which the paragraphs of text are represented as nodes. Paragraphs are treated as term-frequency vectors and vector similarity between every pair of paragraphs is calculated. For those that exceed some threshold, a link is put between them in the map. Paragraphs with highest number of links are considered best for creating the summary.

MEAD by Radev et al [59] treats each document as a term-frequency vector and computes the centroid vector of the cluster of documents. This centroid signifies the central theme of the documents. It then uses cosine similarity with the centroid vector along with other features like similarity with first sentence or title, sentence length, position and query overlap in scoring sentences for extraction.

LexRank Radev et al [58] creates a graph of sentences from the given document(s) and treats it like a social network. The edge weight values in the adjacency matrix of the graph are given by pairwise intra-sentence cosine similarity. The idea is to perform random walk on this graph and use eigenvector centrality as a measure of sentence importance. The intuition is that a well-connected or more central sentence will be more important to the summary.

Chains of related words contribute to lexical cohesion. It occurs over a sequence of a number of related words, called lexical chains. Morris and Hirst [48] uses lexical chains as relationships between semantically similar words in a text. To find semantic relationships, a thesaurus is used. Barzilay and Elhadad [14] segment the text and then construct lexical chains. Then they identify the strong ones and extract the most important sentences.

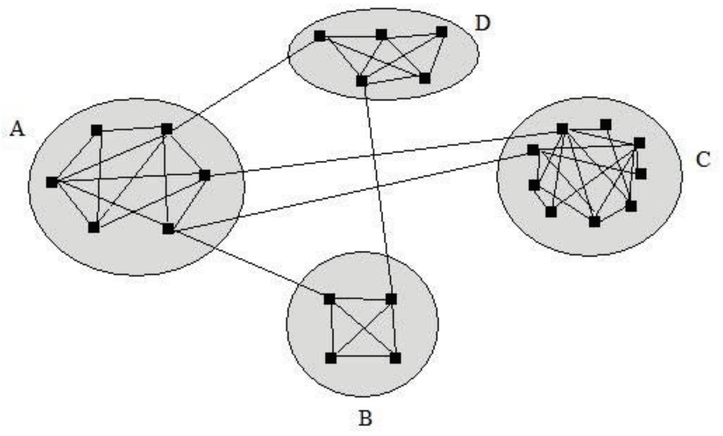


Figure 2.1: Community Structure in a Graph

2.2 Graph Summarization

These techniques use the link structure of the graph to summarize it. The approaches for Graph or Web summarization based on link structure can be broadly divided into the following categories.

Finding Communities

Communities are defined as a set of nodes in a graph which are closer to each other, than to nodes outside the graph. Newman and Girvan [51] describes community structure as "natural divisions of network nodes into densely connected subgroups". An example of communities has been shown in Figure 2.1 Since subgroups A, B, C and D are strongly connected to nodes in their own groups and not so well connected with nodes in other groups, we can say that they represent community structures in the graph.

Community structure is a pattern found in most graphs [21, 50]. In social networks, it is intuitive that people with same jobs, hobbies, interests or those belonging to the same organizations tend to be closer to each other. This can also be seen explicitly in social networks on the web. In the same way, the link structure of the web has similar pages pointing to each other more often.

Finding communities is an important step towards summarization of graphs as they tend to identify the structure of the web as clusters of similar pages. It is also an

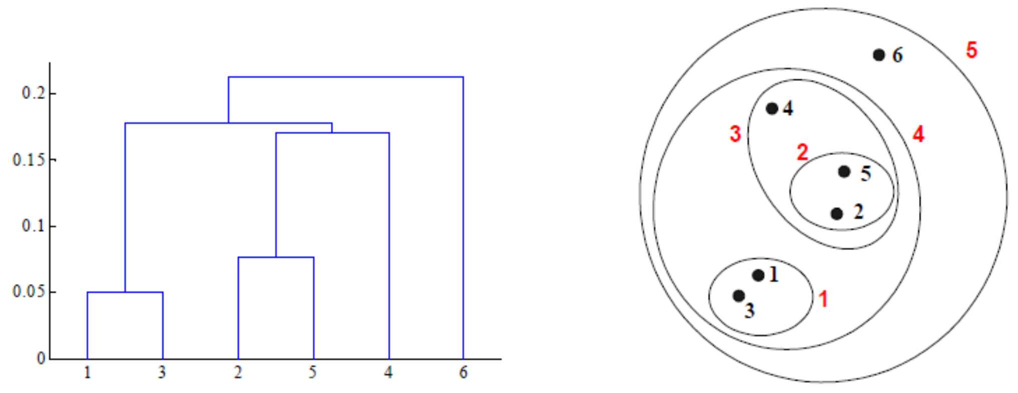


Figure 2.2: Dendogram [67]

important technique used for visualization of large graphs.

There are various techniques for finding communities. They can be categorized into the following methods

Hierarchical Clustering

The idea behind this method is to iteratively group those nodes together that have a high measure of similarity between them. Hierarchical clustering is of two kinds as described below.

- **Agglomerative:** This is a bottom-up approach which starts with individual nodes and a similarity measure between every pair of nodes. Then we keep on grouping the most similar nodes together, forming new nodes(communities). This forms a dendrogram (Figure 2.2) kind of structure which can be cut at any level to yield the required number of communities. Thus, we start with nodes and keep on adding edges between them till the required number of edges have been added or the required number of communities(connected-groups) have been found. Clauset et al [23] follows such an approach. However, Newman and Girvan [51] describe that these kind of methods tend to find only the central cores of communities, while leaving out peripheral nodes.

- **Divisive:** This is a top-down approach which starts with all the nodes and edges and keeps on deleting edges that are between least similar nodes. The edge-deletion can be stopped at any stage when we get the required number of communities.

Newman and Girvan [51] follow an approach which is similar in methodology but defines a different criterion on which the edges are deleted. They do not use the idea of similarity but define betweenness which signifies whether the edge falls in-between many paths between any two pair of nodes. This stems from the observation that two communities are linked by very few inter-community edges. So, all paths between nodes of different communities will pass thorough these small number of edges. Thus finding such edges, through which large number of paths pass, can be a way for identifying such in-between edges. Then they can be removed and individual communities can be isolated. The authors define three measures of betweenness which are explained below.

– **Shortest-path betweenness:** For each edge, its Shortest-path betweenness is given by the total number of shortest-paths between all pairs of nodes in the graph, that run through that edge.

– **Random-walk betweenness:** For every edge, its Random-walk betweenness is measured as the sum total of the number of times a random-walk, passes through it, where random-walks are executed between all pairs of nodes in the graph.

– **Current-flow betweenness:** The graph is regarded as a circuit with each edge having a unit resistance and current flowing from source to sink. Every pair of nodes in the graph is taken as the source-sink pair and the sum-total of current, over all source-sink pairs, for an edge is its Current-flow betweenness.

Out of these measures, the authors recommend Shortest-path betweenness for reasonable results and fast calculation. An important feature of this algorithm is the recalculation step after each edge removal i.e. after each edge has been assigned its betweeness score and the edge with maximum betweenness has been removed, the betweeness for all edges is recalculated. This seems to yield good results.

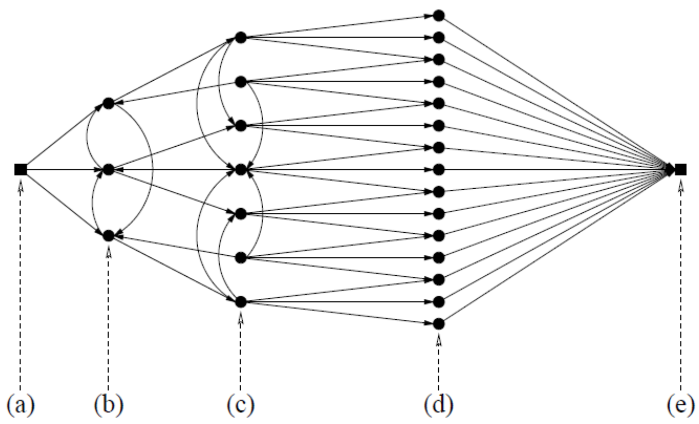


Figure 2.3: Focused community crawling and the graph induced (a)The virtual source vertex (b)Vertices of seed websites (c)Vertices of websites one link away from any seed site (d)References to sites not in b or c and (e)The virtual sink vertex [28]

Max-flow Min-cut

Flake et al [28] defines a community as "a set of web pages that link (in either direction) to more web pages in the community than to pages outside of the community". They make use of this definition to find communities using a maximum flow framework in graphs. Max-flow or Min-cut can be used to disconnect connected components that have larger cuts within and smaller cuts outside. So, they use a focused crawler to calculate maximum flow, starting from source(seeds), to the nodes connected to them and so on with the virtual sink. This is shown in Figure 2.3. Since hubs and authorities [38] signify nodes of a particular community,they use these as the starting points or seeds of their algorithm. They also use expectation-maximization to bootstrap the seeds.

Graph partitioning

This is one of the most common method for finding communities. It consists of recursively partitioning a graph into communities. There are various ways to get partitions. This is elaborated in the Clustering Section.

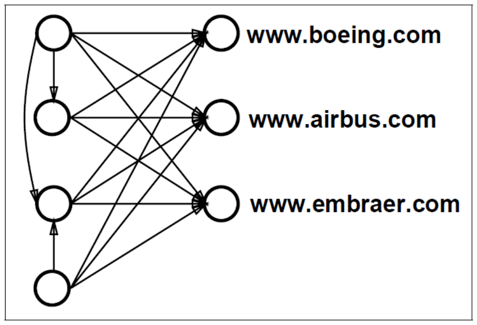


Figure 2.4: Bipartite Core [40]

Finding Bipartite Cores

Kumar et al [41] propose an algorithm for finding emerging communities. They suggest that co-citation of the nodes is an important indicator of them belonging to a community. Thus, they propose the idea that communities are dense Bipartite subgraphs, with a central core. Their algorithm is a two-step process.

1. **Inclusion/Exclusion:** This is the pruning part where a node is either kept in consideration for being part of a community or removed, based on certain criteria which are necessary for a node to be part of the community.
2. **Building cores:** Then they use the Apriori algorithm [11] to build cores iteratively from smaller cores, as any subset of a smaller degree core will have a subset which is a higher degree core and so on

Kumar et al [40] defines Bipartite Cores as: "A bipartite core in a graph consists of two (not necessarily disjoint) sets of nodes L and R, such that every node in L links to every node in R. Links from R to L, or within R or L may or may not be present." An example of bipartite core is shown in Figure 2.4. They also use a similar two-step method for finding communities.

1. The first step is the Elimination/Generation Algorithm to find Bipartite Cores in the graph. The *Elimination* part consists of pruning nodes, which we are sure will not be able to form cores we are interested in finding, as a result of non-fulfillment of some

required criteria, which is similar to the one in [41]. As an example, if we are looking for cores in which each node has a degree 4, we can remove all those nodes from consideration whose indegree or outdegree is lesser than 4. The *Generation* step consists of identifying those nodes which just make it to the considered list, by a narrow margin, and make sure whether all their neighbors have the required degree or not. Those which have neighbors with lower degrees are discarded. These two steps are done iteratively.

2. This step consists of actually finding the communities centered on these cores.This is done by creating a root-set of the nodes that make up the core(in Step1), the nodes pointed to by the nodes in L and the nodes that point to at least 2 nodes in R. Then HITS algorithm [38] is applied to this root-set to get a set of authorities and hubs, which together form a community.

Other Methods

- **Using shingles:** Shingles are fixed size fingerprints for a set of nodes. Two sets can be compared fast using shingles. Gibson et al [30] use Jaccard coefficient for defining shingles so that the number of positions on which the fingerprint vectors of two sets agree indicates their similarity. The basic idea is to group the nodes that have sufficiently overlapping shingles. This is done iteratively.

- **Using circuit laws:** Wu and Huberman [74] propose an algorithm which treats the graph as a electrical circuit and all the edges as unit resistance. A potential difference is applied to a two nodes, and the voltages on all nodes is calculated and the graph is split into two using the median voltage as separator. However, there needs to be some mechanism to identify that the two nodes across which a potential is applied are not in the same community.

Clustering

Clustering is the most natural choice for grouping similar nodes of a graph to summarize it. Most web graph summarization techniques described in this report use clustering to achieve graph compression, visualization or find communities. The techniques described

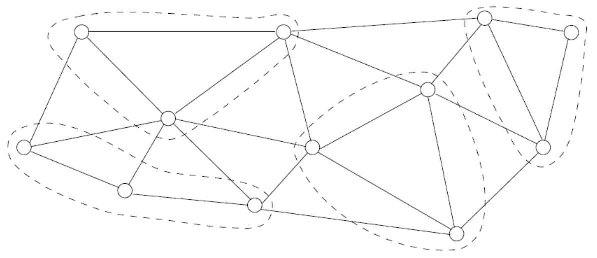


Figure 2.5: Graph Partitioning [52]

here are those which comprise the kind of work in clustering that has been applied to all these fields. The various kinds of clustering that can achieve the purpose of web graph summarization can be categorized into the following.

Graph Partitioning(Multilevel algorithms)

The problem of Graph partitioning can be explained by an example. Imagine a distributed environment, which consists of n processes, represented by nodes of a graph, that need to run on m processors. The inter-process communication can be represented as a edge between two nodes (processes). The objective is to keep the inter-process communication between processors to a minimum while making sure no processor is overloaded or under-loaded. In other words we seek to partition the process graph into as many partitions as we have processors, while keeping the edges between the partitions to a minimum and trying to make the partitions of approximately equal size. This is a graph partitioning problem. Karypis and Kumar [34] defines graph partitioning as -" The problem is to partition the vertices of a graph in p roughly equal parts, such that the number of edges connecting vertices in different parts is minimized." 2.5 illustrates this.

Karypis and Kumar[35] defines k-way partitioning as " Given a graph G = (V, E) with |V| = n, partition V into k subsets, V1, V2, . . . , Vk such that Vi ∩ Vj = ∅ for i != j , |Vi| = n/k, and Ui Vi = V , and the number of edges of E whose incident vertices belong to different subsets is minimized."

The kind of algorithms that use Graph partitioning to a graph recursively, to create a hierarchical structure are called Multilevel algorithms. "They (multilevel algorithms) usually combine a graph contraction algorithm which creates a series of progressively smaller and coarser graphs together with a local optimization method which, starting with the coarsest graph, refines the partition at each graph level" [73] . This is one of the most widely used techniques in visualization and finding communities.

A Multilevel Partitioning algorithm consists of three steps or phases. All these steps are comprehensively described in Karypis and Kumar [35].

- **Coarsening Phase:** In this phase, multiple nodes of a graph are grouped into a single node which is called multinode. The edges of a multinode are the union of the edges of all these constituent nodes, and its weight is the sum of their weights. Thus, we form a graph on a coarser level or a lower level of granularity by making such virtual nodes and edges. This step is done repeatedly till the coarse graph has few nodes. The techniques used for coarsening are Random Matching and Heavy Edge matching.
- **Partitioning Phase:** In this phase, the coarse graph obtained above is partitioned into two partitions of approximately equal sizes. However, Walshaw and Cross [73] report that creating an imbalance might actually lead to better partitions. The most common ways of partitioning are KL algorithm [36] and Spectral bisection.
- **Uncoarsening Phase:** This phase comprises of projecting the partitioned coarse graph back into the original graph and refining partitions at every level of unfolding. KL Refinement and Boundary KL Refinement can be used to do this. Dhillon at al [25] uses a weighted kernel-k means algorithm for the refinement, which does not consider only equal-sized clusters.

  There are many works which use these techniques. Walshaw and Cross [73] presents a mesh-partitioning algorithm which uses an enhanced multilevel algorithm with a Kerninghan-Lin type partition optimization algorithm. Rodrigues et al [63]uses this paradigm for Visualization.

Structural Clustering

Xu [75] proposes a method of clustering nodes, based on their common neighborhood. Vertices with lot of common neighbors are clustered together. They further classify nodes into hubs and outliers. It runs in time linear in the number of edges in the graph.

It defines the notion of Structural similarity for two nodes as the number of shared neighbors between them, normalized by the size of each node's neighborhood. Extending this idea, a vertex becomes a core of the cluster if it has high similarity with a large number of neighboring nodes. With this framework, they propose the SCAN algorithm which checks each node to see whether it is a core or not. If it is, then it starts a new cluster with it. If it is not, it is labeled as a non-member and further analysed. If it has edges to two or more clusters, it is deemed a hub, else it is classified as an outlier.

Simulated Annealing

Virtanen [72] uses Simulated Annealing as a local search heuristic, to cluster nodes of the graph, with respect to the fitness measure used for the simulated annealing. They define a few concepts as follows.

- The internal degree of a cluster is defined as the number of edges that start from within the cluster and end within it.
- The outward degree of the cluster is the number of edges that start from the cluster but end outside it.
- Local density of a cluster is the ratio of the internal degree of the cluster to the maximum internal degree possible within the cluster.i.e.

$$\text{Local density of cluster C} = \frac{\text{Internal degree of C}}{|C|(|C-1|)}$$

- It also defines relative density as the ratio of internal edges to the total number of edges incident on the cluster.

$$\text{Relative density of cluster C} = \frac{\text{Internal degree of C}}{\text{Internal degree of C + Outward degree of C}}$$

The product of Local density and Relative density is used as a fitness measure for clustering. The algorithm can iteratively find clusters by removing the best cluster found after each iteration, or reducing it to a single node.

Graph Compression

Graph Compression techniques, in general, seek to encode a graph in as few number of bits as possible for compact storage or fast transmission over the network. Since this too is a helpful summary, we can look into such methods which can compress large graphs, of the scale of the web. The main ideas behind these algorithms are the observations that

- Most pages point to common links
- Most links on a page are of the same domain, hence are lexicographically similar
- An efficient encoding mechanism will reduce the number of bits/link

The intuition behind the algorithm by Adler and Mitzenmacher [10] is that nodes in a web graph often copy links from other nodes. The authors call the nodes from which the links are copied as reference nodes and encode a node in terms of its reference node, setting the corresponding bits in the vector, which signify the link that was copied. They measure the costs for encoding all nodes as such and create, what they call an Affinity Graph. This graph is the same as the original graph, except that the weight on the edge between a node and its reference represent the cost of encoding, which was calculated before. Its root is the node which has a directed edge from every node in the graph but has no outgoing edges. Then they find the most optimal mapping of node from reference by finding the minimum weight directed spanning tree on the affinity graph, starting with root.

Randall et al [62] propose two algorithms called *Link2* and *Link3*. The paper is also inspired by the idea of multiple nodes pointing to common pages in the web graph(link copying can lead to this) and the fact that most pages point to nodes on the same host. This is intuitive as most pages have navigational links that point to pages on the same host. A university's website is a good example. Arising from this observation, if we order the URLs on a page lexicographically, the "gaps" between the "node pointing to" and the "node pointed to" will be quite small. Link2 algorithm stores only differences between such URLs, instead of the entire URL for the nodes. A further improvement is using smaller codes for the smaller values in the lexicographical ordering. The encoding is done by

Huffman and Nybble codes. Link 3 encodes adjacency by representing them with respect to another list called a representative list(or a union of many representative lists), alongwith the corrections from the representative. Suel and Yuan [66] make a distinction between storing URLs and storing the link structure. For storing URLs, it first strips off the host prefix and then encodes a URL in terms of the longest common prefix, with the URL just preceding it, in a lexicographically ordered arrangement.It then encodes the most frequent words with smaller Huffman codes and the rest of the words by Huffman codes for every two consecutive characters. Links are separated into local(on same host) and global. To store links, it encodes most popular links by short Huffman codes as a global encoding. For local encoding, it encodes most popular links of a host by Huffman codes and the rest by taking advantage of small gaps.

Boldi and Vigna [16] extend the ideas presented above using locality(small lexicographical gap), similarity(common pages pointed to) and consecutivity of the links. Consecutivity means that many links in a page are lexicographically consecutive. They then use Reference compression i.e. coding the adjacency list with respect to a reference list. They also propose using Differential compression which codes blocks , rather than each unit of the adjacency list. They also use Intervals, in which they identify long enough subsequences and represent them by their left extreme and length. The remaining are compressed using differential compression.

Blandford et al [15] do O(n)-bit encoding with O(1) access time for graphs satisfying either an nc, c < 1 edge or vertex separator theorem.

Raghavan and Garcia-Mollina [61] propose an S-node representation which is a hierarchical representation which groups a number of nodes(subgraph) into a supernode and encodes the lower-level graphs. To do this, it partitions the nodes, starting with the initial partition based on domain name i.e. all pages of asu.edu will be kept in the same partition. For the following partitions, it uses two methods- URL split or Clustered split. URL split partitions on the basis of common prefixes. Clustered split, on the other hand clusters pages with similar adjacency lists together using k-means.

Visualization Techniques

Visualization of graphs consists of finding techniques which make a graph easier to visualize and interpret intuitively, without overwhelming the user. We can apply visualization techniques to the web graph in order to summarize it.

We need to pick up the visualization techniques which can process large number of nodes. Most of the visualization techniques work well for small graphs but not for graphs comparable to the web. Visualization techniques are described below.

Hierarchical Graph Partitioning

This is a widely used method for Visualization. Graph clustering is also a widely used technique. For detailed treatment of Hierarchical Graph Partitioning, please refer to the "Graph partitioning" method under Clustering.

Rodrigues et al [63] use Hierarchical Partitioning for Visualization. Their algorithm uses k-way partitioning to create a hierarchy of communities within communities, which it calls the Graph-tree data structure. Papadopoulos and Voglis [54] recursively divide a graph into modules(a set of vertices having the same neighborhood) in a tree-like fashion, to be able to draw each module differently, to aid cognition. Eades et al [26] propose an algorithm to draw clustered graphs using orthogonal grid rectangular cluster drawings, which utilize recursive partitioning and gain optimality in some aesthetic criteria.

Graph layout

Graph Layout is the way of drawing graphs so that the user can visually understand it. The main goal of these algorithms is clarity and aesthetics. However, these algorithms are not well-suited for large graphs.

The major Graph layout algorithms are presented in Figure 2.6, which has been taken from [32].

Visualization of Features of a Graph

Visualization of some properties of a graph like Min-cut plots and A-plots instead of the whole graph [22] is useful in applications like anomaly detection. A Min-cut plot is obtained

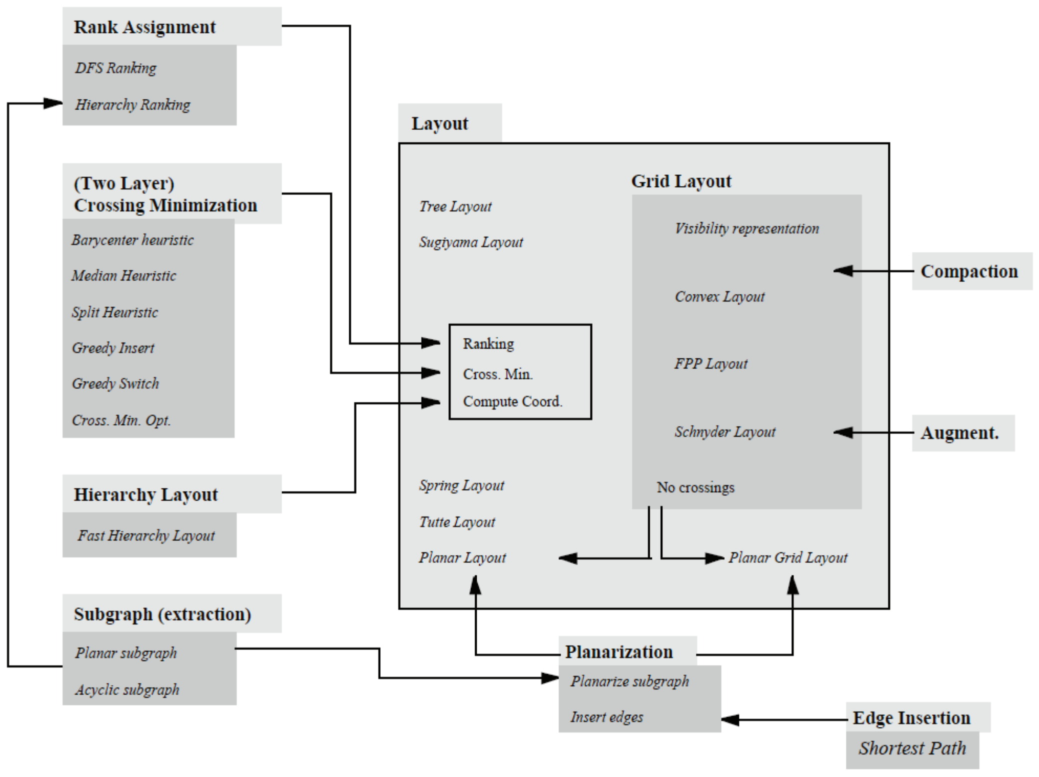


Figure 2.6: Overview of Graph Layout Algorithms [32]

by recursively dividing a graph on its Min-cut into two, and plotting a graph between the number of edges in the resulting graph(s) v/s the number of edges of the graph that were in the cut set.

A-Plot contains the following:

- Adjacency matrix of the graph in which nodes are ordered in decreasing order of their network value
- Degree of a node with rank of its network value
- Adjacency matrix of the graph in which nodes are ordered in decreasing order of their degree

They also identify patterns in the plots which are representative of real large graphs.

Web Navigation Systems

These systems are designed to help the user in navigating websites or the web in general. One way to do this is to summarize the web graph so that the user only sees the most relevant part of it at a time, and is not overwhelmed by the multiplicity of options to click next.

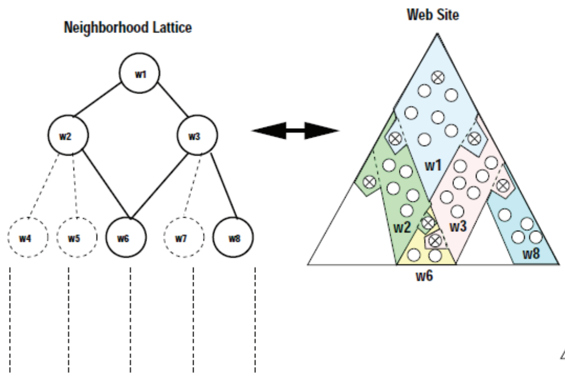
Candan and Li [19] present an approach to solve the problem, which they use for website summarization. They define two kinds of neighborhoods - the Physical neighborhood, which is just determined by the URL prefix and the Logical neighborhood which is a hierarchy of neighborhoods, with entry points to child neighborhoods, which reflects the natural course of navigation, as described in Li et al [43], and is shown in Figure 2.7.

The algorithm starts by summarizing the root neighborhood into fewer nodes(called focus nodes) by graph summarization technique, and then summarizes each successive neighborhood the user enters.

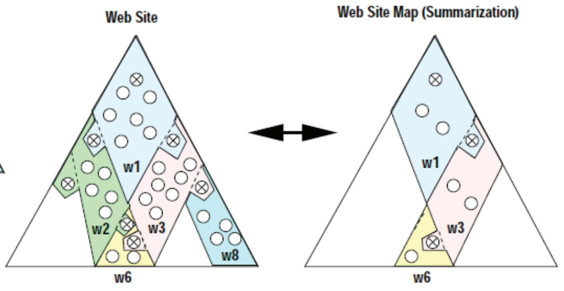
To find the summary of a neighborhood, the algorithm starts with a set of seed nodes, comprising of the focused entry nodes from its parents and the focused entry nodes to lower-level of neighborhoods, and k-dominant nodes in the neighborhood [18]. Edges on the shortest path, between pair of dominant nodes that does not pass through any other dominant node are kept in the summary. The nodes that are selected thus become the focused nodes for the next iteration of the algorithm on sub-neighborhoods.

Information-Theoretic Approaches

Minimum Description Length(MDL) is widely used for summarization, as is also observed in [42]. MDL is inspired from the observation that there are redundancies in the data. So, it seeks to represent data in the form of a model, which is representative for most of the data, and a set of corrections, for the data points that the model didn’t accommodate or which are extraneous in the model. The best MDL description is the one which minimizes the size of the model + corrections.



(a) Entry points



(b) Summarization

Figure 2.7: Logical Neighborhoods and the Summarization Process. Crossed circles denote entry points of neighborhoods. Each sub-neighborhood contains one entry point per its parent neighborhoods. Each parent neighborhood includes the entry points of its sub-neighborhoods.[19]

For a given graph, Navlakha et al [49] represent it in the form of a graph summary, alongwith a set of edge corrections. The graph summary is a coarser representation of the original graph, consisting of supernodes and superedges, similar to the concepts described before. A supernode is a virtual node that is the grouping of multiple nodes, while a superedge from a supenode A to B is the aggregation of edges that connect any node of A to any node of B. The superedge corrections, for the edges that are not a part of the model, are represented as -(x,y) for edges not present in the original graph but are present in the model, and +(x,y) for edges that are present in the original graph and not in the model. Two algorithms are suggested by them for computing MDL representations.

- The Greedy algorithm iteratively merges those pairs of nodes together, which minimize the MDL cost.
- The Randomized algorithm randomly picks nodes and merges it with the best node in its 2-hop neighborhood.

They also suggest versions of these algorithms for lossy summarization (it is not possible to exactly reconstruct the original graph using the summary and the corrections). Those techniques too are acceptable for us too, since we seek to summarize the graph

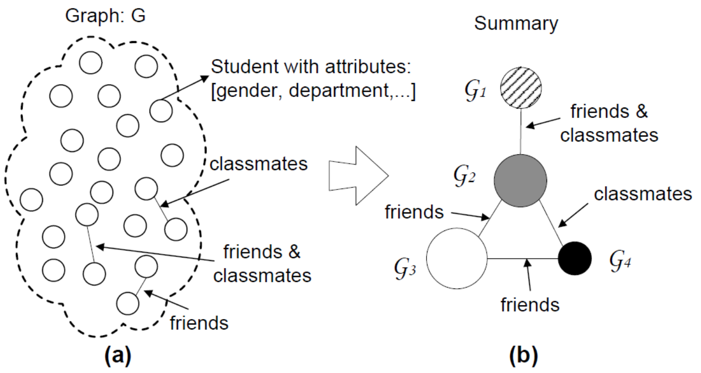


Figure 2.8: Graph Summarization by Aggregation [70]

approximately.

Database Type approaches

These kind of approaches are characterized by operations analogous to SELECT and GROUP BY as in databases. Also the attributes of nodes are of concern, which is different from almost all the other approaches presented.

Tian et al [70] present such a database-type framework for summarization.

- A summary graph is created from the nodes of the graph by grouping nodes based on attributes selected by the user and the relationships between them. An algorithm called SNAP is used for this purpose.
- The user can control the granularity of summaries by operations analogous to "drill-down" and "roll-up" in OLAP. An algorithm called k-SNAP is used for generating k-size graph.

Figure 2.8 describes the mechanism of the graph summarization in a simple manner.

k-SNAP is achieved by both Top-down and Bottom-up approaches. The first approach starts from the group of all nodes with desired attributes and splits repeatedly, based on some criteria, till k-groups are reached. The second approach, on the other hand, builds up from the nodes and merges till k-groups are reached. K-SNAP is proved to be NP complete thus heuristics are used.

Graph Anonymization

These techniques seek to anonymize the nodes in a graph to maintain privacy, so that reconstruction and discovering of node identities is avoided.

As observed by Backstrom et al [13], it is not enough just to remove the identities of the nodes. The structure of the graph eg. degrees of its nodes, their similarity etc. is also a very good indicator of the identity of nodes in it [31].So, the graph itself needs to be anonymized.

Liu and Terzi [44] is a recent work in this direction. A degree sequence of a graph is a vector whose ith position stores the degree of the ith node in the graph. The algorithm is a two-step approach.

- The first step creates a k-anonymized vector of the degree sequence of the given graph such that the Degree Anonymization cost is minimized.
- The second step consists of creating a graph whose degree sequence is the k-anonymized vector obtained above, and whose every node shares the same degree with k-1 other nodes.

Chapter 3

BACKGROUND

3.1 MEAD Summarizer

MEAD [60] is a public-domain, portable, *Centroid-based* multi-document text summarization system. It also contains packages for evaluation of summaries [7].

MEAD is an *Extraction-based* summarizer which consists of assigning a score to each sentence in the text based on some heuristics and then extracting the highest scoring sentences to create a summary.

MEAD accepts a cluster of documents as input and and produces a summary of this cluster as output. The summarization process in MEAD consists of Feature Extraction, Sentence Scoring and Re-ranking. In the beginning, each document is considered as a bag of words and after stemming and stop-word removal, is represented as weighted TF*IDF vector.

Feature Extraction

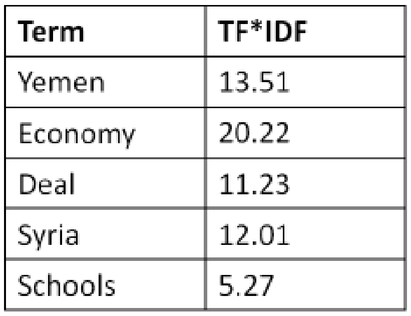
MEAD uses a combination of the following features in assigning a score to each sentence

- Cosine similarity with *Centroid vector* of cluster
- Position of the sentence in the document
- Content Overlap with the first sentence in the document
- Length of the sentence

These features are further explained in detail below.

Centroid

A Centroid is a set of statistically important words in a cluster of documents. The Term Frequency(TF) of a word in the cluster is the average number of times the word appears across the cluster. The Inverse Document Frequency(IDF) for each word in the cluster is computed from the corpus. The centroid is a TF*IDF vector of all words in the cluster.



Term	TF*IDF
Yemen	13.51
Economy	20.22
Deal	11.23
Syria	12.01
Schools	5.27

Figure 3.1: Centroid Example

The Centroid is computed iteratively by only including those documents whose similarity with the current Centroid is above a certain threshold. The similarity is computed according to the formula below.

$$sim(D,C) = \frac{\sum_k (d_k * c_k * idf(k))}{\sqrt{\sum_k (d_k)^2} \sqrt{\sum_k (c_k)^2}} \tag{3.1}$$

Initially just the first document is considered and thus the centroid is the document itself. Then the next document is processed if its similarity with the centroid is greater than the threshold. This way documents are added iteratively and the centroid is recomputed after every iteration.

MEAD is a Centroid-based summarizer and hypothesizes that sentences containing words from the Centroid are more representative of the text. The next Section describes how this is used as a feature in summarization.

Centroid Value

For each sentence, its total centroid value is computed as the sum of centroid values of each word in it. For example let the vector shown in Figure 3.1 be the centroid computed from the cluster of documents input to MEAD. For the sentence "Opposition in Yemen Divided Over Deal", the centroid value will be 24.74 which is the sum of centroid values of its individual words (opposition=0; yemen=13.51; divided=0; deal=11.23).

Centroid value of a sentence is used as a feature in computing its score for creating the overall summary.

Positional Value

The position of a sentence within the document is also considered as a measure of its importance. The first Sentence in the document gets same score as the highest ranking sentence according to Centroid Value. The Positional Value for the i^{th} sentence is computed according to the formula below , where n is the total number of sentence in the document and C_{max} is the Centroid Value of the highest ranking sentence in the document in terms of Centroid Value.

$$P_i = \frac{n - i + 1}{n} * C_{max}$$

First Sentence Overlap

MEAD rewards similarity of a sentence with the first sentence in the document. The first sentence overlap is computed by the inner product of the vectors for the current sentence and the first sentence of the document.

$$F_i = \vec{S_1}\vec{S_i}$$

Sentence Length

A longer sentence is supposed to contain more information and thus is considered more important. Length is the number of words in the document. This feature was introduced in Radev and Winkel [57] in 2002.

Sentence Scoring

The score of each sentence in the document is computed as a weighted linear combination of its Centroid Value(C), Position Value(P) and the First Sentence Overlap(F) in [60] as shown below.

$$Score(S_i) = w_c C_i + w_p P_i + w_f F_i$$

where w_c ,w_p and w_f are weights assigned to each feature.

Re-Ranking

After the sentences are ranked according to their total scores, for every sentence that overlaps with a higher ranking sentence, a Redundancy Penalty is subtracted from its score. This is done to reduce repetition in the summary. The cross-sentence word overlap is computed according to the following formula:

$$R_s = 2 * \frac{\text{(number of overlapping words)}}{\text{(number of words in sentence1} + \text{number of words in sentence2)}}$$

$$w_R = Max_s(SCORE(s))$$

Redundancy Penalty is computed as $w_R R_s$ where w_R is the weight of the penalty. Thus the score of the sentence along with the Redundancy Penalty becomes

$$Score(S_i) = w_c C_i + w_p P_i + w_f F_i - w_R R_s$$

After this correction, all sentences are reranked according to their new scores. This process is repeated until the summary extract stops changing on reranking.

3.2 Multidimensional Scaling

Multidimensional Scaling [71] is a set of techniques to map data into multidimensional space, given their pairwise dissimilarities so that their distances in the multidimensional space closely reflect their actual dissimilarities. These techniques seek to minimize stress which is the sum of difference between distances in the mapped values and actual dissimilarities for every pair of objects.

$$stress = \sqrt{\frac{\sum_{i,j}(d'_{i,j} - d_{i,j})^2}{\sum_{i,j} d^2_{i,j}}}$$

The following general algorithm can be used to obtain the MDS representation of given data:

- Start with a (random) configuration of points in smaller dimension
- Apply some form of steepest descent iteratively to minimize the stress.
- If moving objects does not decrease the stress, add new dimensions

3.3 PageRank

The importance of a web page can be measured by the number and the importance of pages that link to it. A high number of links and links from important pages increase the importance of a page.

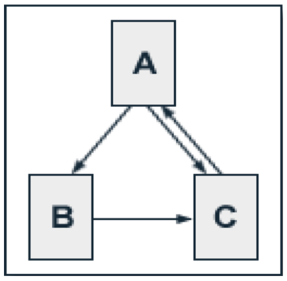


Figure 3.2: A Small Network

PageRank [17, 53] is a method to estimate the relative importance of web pages by just using the link structure of the web. The PageRank of a page is the sum of the PageRanks of all pages that link to it and this rank is further evenly divided among the pages it links to. Thus PageRank is defined recursively and is influenced by ranks of other pages.

As an example, consider the scenario in Figure 3.2. Here,

$$PR(A) = PR(C)/1$$

Since C has just one outgoing link to A, PageRank of A is equal to the entire PageRank of C. However,

$$PR(B) = PR(A)/2$$

A's PageRank is divided equally among both its outgoing links. Therefore,

$$PR(C) = PR(A)/2 + PR(B)/1$$

Let u be a web page. Let B_u be the set of pages pointing to u and F_u be the set of pages that u points to. N_i represents the number of links from a web page i. Then according to the simplified idea of PageRank,

$$PR(u) = \sum_{v \in B_u} \frac{PR(v)}{N_v}$$

Let A be a square matrix whose rows and columns represent web pages. If there is an edge from u to v, $A_{u,v} = 1/N_u$. Otherwise, $A_{u,v} = 0$. If there is a rank vector R of all web pages such that $R = AR$, then R is the eigenvector of A with eigenvalue 1, which is the principal eigenvalue for a stochastic matrix like A. The principal eigenvector gives the stationary distribution. We can compute it by power iteration.

This model has the problem of rank sinks which are pages that accumulate PageRank but do not distribute it. To overcome this problem, a rank source $E(u)$ is introduced over web pages which is a user-defined parameter.

$$PR(u) = \sum_{v \in B_u} \frac{PR(v)}{N_v} + E(u)$$

In matrix notation, we then have $R = AR + E$. Since $||R||_1 = 1$ we can rewrite $R = (A + E \times 1)R$. Therefore R is an eigenvector of $(A + E \times 1)$

PageRank can also be explained in terms of the "random surfer" model ie. a surfer performing a random walk on the web graph. Assume there is a "random surfer" who keeps clicking on a link at random, visiting the page and then clicking at a link at random on that page and so on. The importance of a page is the probability that the random surfer will find himself on that page. Further the surfer might get bored and jump to a random page. This is reflected by the user defined distribution E.

3.4 CUTS

CUTS: Curvature-Based Development Pattern Analysis and Segmentation for Blogs and other Text Streams [56] is an algorithm to segment a text stream or blog using underlying topic development patterns.

The algorithm generates a representation for each blog entry and maps them onto a curve. Then this curve is analyzed to identify topic segments and topic development patterns.

The algorithm can be broken down into the following steps:

Representation of Stream

Each entry in the blog is considered as a bag of words and its TF*IDF vector is constructed after stemming and stop-word removal. For the i_{th} entry in the blog, let the vector be $P_i = w_{i,1}, w_{i,2}, ..., w_{i,n}$ where n is the size of the set of keywords contained in all entries concerned.

The cosine similarity between any two entries i and j can then be computed as:

$$s_{i,j} = \sum_{k=1}^{n} w_{i,k} \cdot w_{j,k}$$

Then the dissimilarity between the entries i and j is $d_{i,j} = 1 - s_{i,j}$.A dissimilarity matrix is generated by computing the pairwise dissimilarities for all text entries.

$$D = \begin{bmatrix} d_{1,1} & d_{1,2} & ... & d_{1,N} \\ d_{2,1} & d_{2,2} & ... & d_{2,N} \\ : & : & d_{i,j} & : \\ d_{N,1} & d_{N,2} & ... & d_{N,N} \end{bmatrix}$$

Curve Construction

Using the dissimilarity values obtained in the last step, the entries are plotted onto a 1-dimensional space using Multidimensional Scaling [71]where the distances between entries closely reflect their dissimilarities.

Then a 2-dimensional curve is constructed by plotting the time or sequence number of entries onto the x-axis and the MDS-computed distance coordinate on the y-axis. Thus the x-axis denotes the ordering between the entries and the y-axis denotes the dissimilarities between them. This curve obtained by plotting consecutive entries is called the *CUTS Curve*. An example is shown in Figure 3.3. The CUTS curve represents the development of textual content over time.

The paper identifies three different kinds of topic development patterns - Dominated, Drifting and Interrupted as shown in Figure 3.4.

- **Dominated:** A dominated pattern is shown in (a). The entries are similar to each other and hence lie close to each other in the 1-dimensional MDS coordinate space. When these entries are plotted in sequence, a horizontal pattern is obtained. Since the topic remains relatively stable, this is a Dominant pattern.
- **Drifting:** When successive entries talk about slightly different topics, a Drifting pattern is obtained as shown in (b). Even though consecutive entries are similar, there is a huge difference between topics at the beginning and topics at the end of the pattern. The slope of such a curve gives the speed at which the topic evolves.

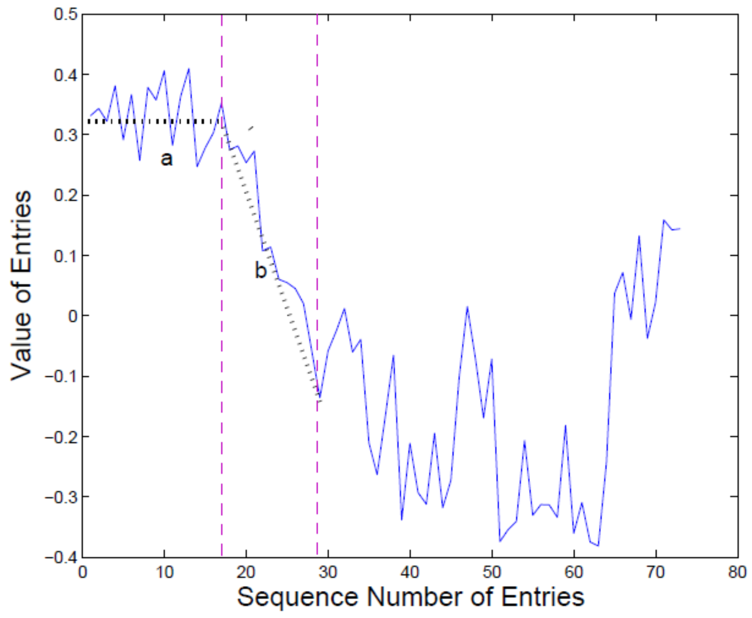


Figure 3.3: CUTS Curve [56]

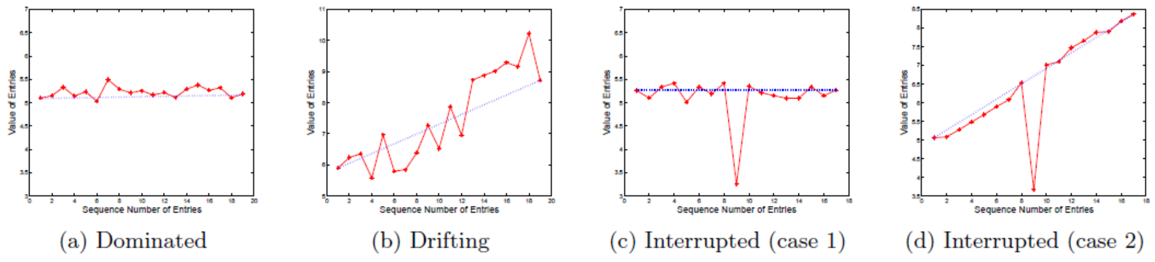
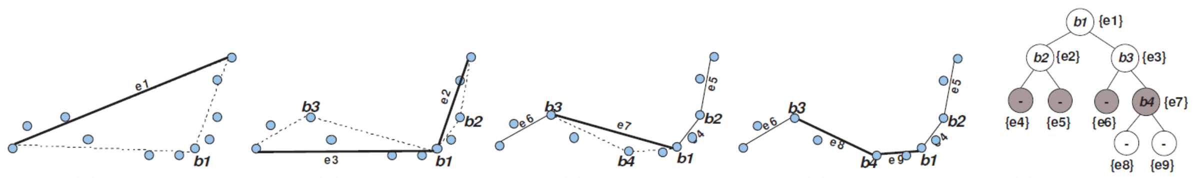


Figure 3.4: Topic Development Patterns [56]

- **Interrupted:** In an Interrupted pattern, a sudden and temporary topic emerges in the pattern as shown in (d) and (e). The interrupt consists of two opposite drifting patterns with start and end (respectively) very similar to each other which is why it is regarded as an interruption within an otherwise dominant or drifting topic.

Curve Segmentation

The curve obtained in Figure 3.4 is segmented using polygonal approximation. Let the curve be denoted by $C = \{x_i, y_i\}_{i=1}^{N}$. The goal of curve segmentation is to find the subset of dominant points $D = \{x_i, y_i\}_{i=1}^{M}$ (where $M \leq N$ and $D \subseteq C$) such that each resulting curve segment has uniform features.



(a) Step 1 (b) Step 2 (c) Step 3 (d) Step 4 (e) Resulting tree

Figure 3.5: Adaptive Curve Segmentation [45]

In a polygonal approximation approach, the curve is split into a series of straight lines. This paper uses Adaptive Curve Segmentation [45] to segment the curve. The working of this algorithm is explained with the help of an example in Figure 3.5.

In the first step the entire curve is represented by the line segment e_1 which is obtained by joining the farthest points on the curve. Next, the point on the curve which is farthest from this line b_1 is located. This is called a break-point because it divides the curve into two parts. Thus e_1/b_1 becomes the root of the binary tree representing the cuts curve as shown in (e). The same process is repeated for the parts on either side of b_1. The left part yields a break-point b_3 and the right part yields a break-point b_2 as shown in (b). This process is repeated until the number of points in any given segment is at most *MinSpan*, which is a user-defined parameter. As we go from root to the leaf nodes in a tree constructed this way, each level approximates the curve with a higher level of granularity.

The optimal series of lines for approximating the curve is determined by traversing the binary tree in (e) and comparing the significance of a node to the significance of its child nodes. If the significance of any of the children exceeds that of the parent, the parent is removed from consideration and the children get upgraded.

A curve segment can be defined as a 4-tuple
$g_i = (k_i, \sigma_i, (x_{start}, y_{start})_i, (x_{end}, y_{end})_i)$.

- k_i is the *Slope* of the line segment which indicates the topic drift in the curve
- σ_i is the average of the distance between points on the curve to the line segment approximating them. This measures the *Concentration*. A higher σ means that the topics being covered are far from the line approximating them. This signifies that a

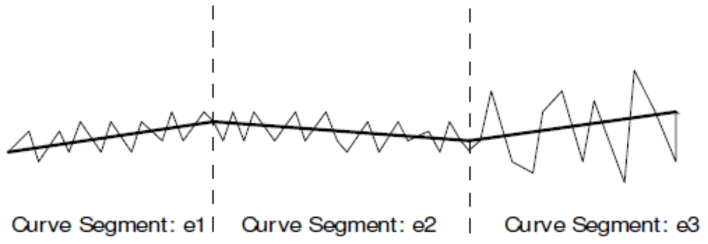


Figure 3.6: Combining Curve Segments [56]

wide variety of loosely concentrated topics are being covered. On the other hand, a low average distance indicates highly concentrated topics in the segment. In Figure 3.6 e1 and e2 are more concentrated than e3.

- $(x_{start}, y_{start})_i$ and $(x_{end}, y_{end})_i$ denote the start and end points of the segment.

Eliminating Over-Segmentation

The Curve Segmentation algorithm is a greedy approach and may lead to oversegmentation. This would lead to a decrease in precision. Thus, topic segments with relatively homogeneous development need to be combined into a single *base topic segment*.

Given two consecutive curves g_i and g_{i+1}, they can be combined into a single topic segment if:

$$(|k_i - k_{i+1}| < \lambda_{drifting}) \wedge (|\sigma_i - \sigma_{i+1}| < (\sigma_i + \sigma_{i+1})/2)$$

Here $\lambda_{drifting}$ is a tunable parameter which reflects the algorithm's sensitivity to differences in topic evolution speeds. It is calculated as a percentage of the overall change in the data.

Topic Development Patterns

After the base topic segments are identified, they are classified into topic development patterns as follows:

- Dominated:$|k_i| < \lambda_{drifting}$

- Drifting:$|k_i| \geq \lambda_{drifting}$

- Interrupted: Two base topic segments can be combined into an interrupted segment if they are drifting, have opposite slopes and their start and end (respectively) are similar to each other:

$$(|k_h| \geq \lambda_{drifting}) \wedge (|k_{h+1} \geq \lambda_{drifting}) \wedge$$

$$(k_h \times k_{h+1} < 0) \wedge$$

$$(|k'_{h-1} - \tilde{k_h}| + |k'_{h+2} - \tilde{k_h}| < \lambda_{drifting})$$

  where k'_{h-1} and k'_{h+2} are the slopes of the base topic segments just before b_h and after b_{h+1} respectively and

$$\tilde{k_h} = \frac{|(y_{start})_h - (y_{end})_{h+1}|}{|(x_{start})_h - (x_{end})_{h+1}|}$$

3.5 Text Summarization Using CUTS

TDSum approach [39] uses the topic development patterns identified by the CUTS algorithm (Section 3.4) to create a summary. Initially the CUTS algorithm is applied to the text to be summarized. After the CUTS curve is obtained, its features are used to budget the required compression amount of the summary into compression amounts for the individual segments. The CUTS curve tree shown in Figure 3.5, (e) is traversed in a top-down manner. The root of the tree is initially allocated the total given budget. Then each child is allocated a part of the parent's budget recursively by comparing values of the following features:

- Length (the number of words in a segment).
- Slope
- Concentration

Compression amounts of left and right child of a node in the binary tree representing the CUTS curve are calculated using the following equations:

$$C_l \cdot L_l + C_r \cdot L_r = C \cdot (L_l + L_r) \tag{3.2}$$

$$\frac{C_l}{C_r} = W_L \cdot \left(\frac{L_l}{L_r}\right)^{i_L} + W_S \cdot \left(\frac{S_l}{S_r}\right)^{i_S} + W_{AD} \cdot \left(\frac{AD_l}{AD_r}\right)^{i_{AD}} \tag{3.3}$$

Each feature is given a weight such that $W_L + W_S + W_{AD} = 1$. i_L, i_S and i_{AD} are inversion parameters and can have values 1 or -1 depending on whether the feature contributes directly or inversely to the compression budget.

After the compression budget is calculated for a node in the tree, it is determined whether the node can be marked for selection. If a node is marked for selection, the subtree rooted at that node is not explored further for assigning compression budgets. A node is marked for selection in the following cases:

- If the node is a leaf node, it is marked as selected and sent as input to the summarizer.
- If the compression budget allocated to the node is too high or 100% then it is marked as selected and included in the summary.
- If the compression budget allocated to the node is less than the shortest sentence in the text, then it is marked selected and not included in the summary.

Chapter 4

APPROACHES TO SUMMARIZING NON-LINEAR NARRATIVES

4.1 Representative Algorithms

We discussed the broad categories of approaches that can be used to summarize non-linear narratives like Strandmaps in Section 1.4. In this Section, we will look at representative algorithms from each of these approaches. Our goal is to compare these approaches to understand their effectiveness in creating summaries of Strandmaps.

4.2 Purely Text-Based Approach

To create a purely text-based summary of a Strandmap, the textual content in each node of the Strandmap is extracted into separate documents and then this set of documents is sent as input to the multi-document summarizer MEAD [60]. The MEAD summarizer takes a cluster of documents as input and returns a centroid-based text summary as output. It has been covered in detail in Section 3.1.

The summary obtained thus is just a text-based summary ie. no information about links between the nodes is used in creating this summary.

The algorithm for obtaining a purely text-based summary of a Strandmap can be summed up as:

- For every node n_i in Strandmap S, extract its text into document d_i
- Send documents $d_0, d_1 ... d_N$ to MEAD as *input*, where N is the number of nodes in the Strandmap
- The *output* of MEAD is the *purely text-based* summary of S

4.3 Purely Structure-Based Approach - PageRank (without Content)

To create a purely structure-based summary of a Strandmap, we use PageRank [17, 53] which has been explained in Section 3.3. PageRank is a measure of structural importance of a node in a graph and we use this notion of importance in creating a summary.

The original Strandmap can be directly used as a graph and PageRank of its nodes can be computed using the PageRank algorithm but this will not be sufficient for our purpose. In a Strandmap, a node may contain multiple sentences. Just ranking the nodes will not help us rank the sentences within it. Since a sentence is the basic unit of our summary, we need to transform this graph in terms of sentences to obtain the relative structural importance of every sentence within a Strandmap. For this we create a *Sentence Graph* from the Strandmap in which every sentence is represented by a node called a *Sentence Node*. Links between original nodes are transformed into links between *Sentence Nodes* and are called *Sentence Links*.

Starting with the given Strandmap, we create Sentence Nodes by creating a new node in the Sentence Graph for every sentence in the Strandmap and creating internal and external links from it. The internal links are between these sentence nodes and correspond to the order in which the text is read within a node. We put a link from first sentence of node to the second sentence, from the second to the third and so on.

- For every node n_i in Strandmap S, where n_i consists of sentences $s_1, s_2 ... s_N$, create N new nodes $p_1, p_2 ... p_N$ in Sentence Graph SG such that p_j contains sentence s_j.
- In Sentence Graph SG create $N - 1$ links $l_{1,2}, l_{2,3} ... l_{N-1,N}$ such that $l_{j,j+1}$ points from p_j to p_{j+1}.

After the nodes have been created as described above, we translate the links between nodes originally present in the Strandmap to external links in the Sentence Graph. Two methods are described below which differ in the way the *Sentence Links* corresponding to these links are created between Sentence Nodes.

Uniform Connectivity

In this method, all links of a node in the Strandmap are preserved in each of the Sentence Nodes derived from that node. Thus in terms of external linkage, each Sentence Node is structurally equivalent to the original node in the Strandmap from which it is derived. So for each link that existed, we create multiple links,one for each of the Sentence Nodes derived from the original nodes.

If there is a link from node n_i to n_k in S, And $P = p_1, p_2...p_N$ are sentence nodes derived from n_i, And $Q = q_1, q_2...q_{N'}$ are sentence nodes derived from n_k, For each $p_a \in P$ and $q_b \in Q$, there exists a link from p_a to q_b(Create a link from each sentence node in P to each sentence node in Q). Figure 4.1(b) shows this kind of Sentence Graph.

Biased Connectivity

In this method, the first sentence node link inherits all the incoming links of a node and the last sentence node inherits all the outgoing links of the node. The sentence node which corresponds to the first sentence in the node is given more importance for an incoming link and the sentence node derived from the last sentence is given more importance for outgoing link. The reason behind this is because often in text summarization, the sentences or words in the beginning of a document or a paragraph are considered more important than the other sentences in the text [68, 69, 60]. Taking just the first sentence of each paragraph to create a summary is a heuristic which is often used to create a quick summary. Moreover such an arrangement of links maintains the flow in which the sentences would be read if the user just hopped from node to node reading every node's content.

If there is a link from node n_i to n_k in S, And $P = p_1, p_2...p_N$ are sentence nodes derived from n_i in the order of sentences in the node, And $Q = q_1, q_2...q_{N'}$ are sentence nodes derived from n_k in the order of sentences in the node, Create a link from p_N to q_1. Figure 4.1(c) shows this kind of Sentence Graph.

4.4 Combining Text and Structure

The previous sections discussed about purely text-based and purely link-based approaches but Strandmaps contain both links and text. So, each of these solutions are two ends of the spectrum and do not completely capture the nature of Strandmaps. We need an approach which can combine both links and text in a meaningful way. We propose two approaches which are described in this Section.

Approach I : Topic-development Curve Analysis

TDSum [39] uses the underlying topic-development patterns found by CUTS [56] to create summary of a text as explained in Section 3.5. We use the same kind of approach to

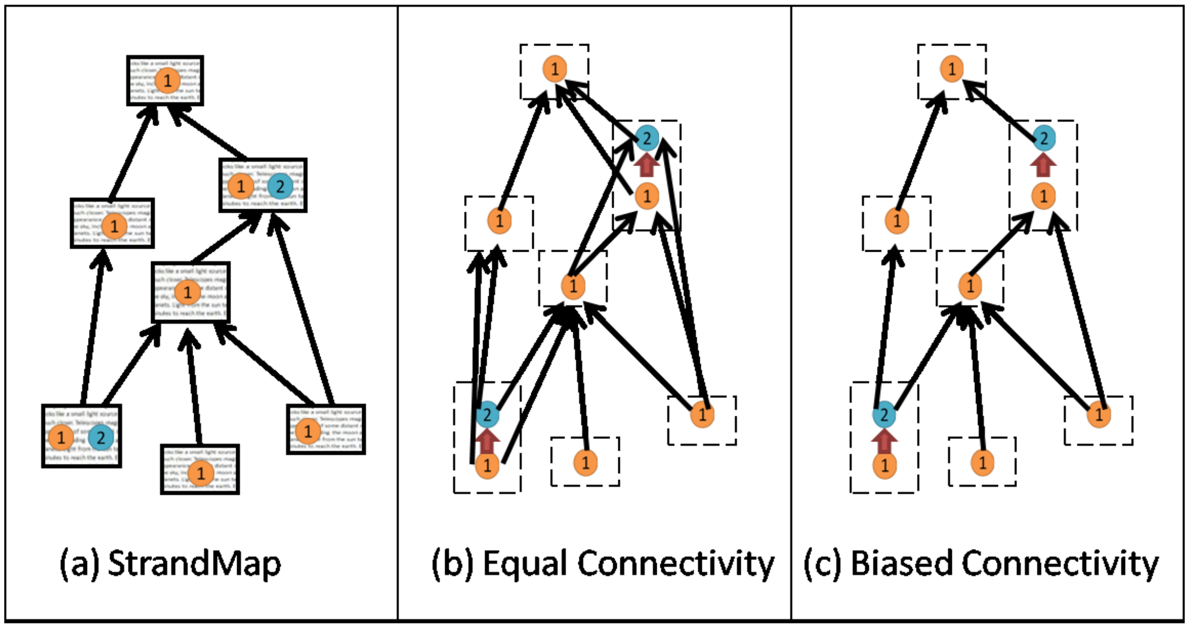


Figure 4.1: Sentence Graphs

summarize a Strandmap.

In a CUTS curve, the y-axis represents the 1-dimensional MDS coordinate which gives a measure of the similarity between text entries. This can be done for Strandmaps too. Each node in the Strandmap contains text that can be represented as a term-weight vector and pairwise dissimilarities between nodes can be computed. MDS coordinate of each node can be obtained which will give a measure of how close the nodes are in terms of textual content.

The y-axis of the CUTS curve represents the order or sequence of text entries. A major difference between a Strandmap and a text stream is the absence of ordering among nodes in a Strandmap. We need to know the order between text entries in order to create the CUTS curve. The optimal reading order for a Strandmap should meet the following criteria:

- No advanced concepts should come before the prerequisites are covered
- Similar content should be together

To figure out this optimal reading order, we use *Constrained* Multidimensional Scaling explained in the next Section.

Constrained Multidimensional Scaling (CMDS)

Constrained MDS is a technique we developed to find the optimal reading order of a Strandmap. MDS can be used to bring similar content together but we need to make sure that while doing so, the ordering of prerequisites is not violated. The idea of Constrained MDS is to start with a dissimilarity matrix and perform Multidimensional Scaling(Section 3.2) but while reducing overall stress, allow only those moves that do not violate the pre-requisite ordering of the Strandmap. This algorithm is described below:

- Given:
  - Pairwise similarities between all nodes
  - Prerequisite constraints in the Strandmap
- Initialize:
  - Assign random coordinates such that nodes are in Topological order
- Iterate: Steepest descent to minimize error without violating constraints
  - Error is the total sum of the differences between the computed CMDS coordinate and the actual dissimilarity between every pair of points.

$$Error = \sum_{i:0\to(N-1),j:0\to(N-1)} |distance_{i,j} - dissimilarity_{i,j}| \tag{4.1}$$

  - Gradient for a point A is the normalized total error between A and the rest of the points. It is used to determine the direction in which to move the point to minimize stress for a point.

$$Gradient_A = \frac{\sum_{i:0\to(N-1),i\neq A}(-1)^p Error(A,i)}{(N-1)^2} \tag{4.2}$$

  where p is *even* if A is to the left of i and p is *odd* if A is to the right of i

  - Computing new coordinate for point A

$$x'_A = x_A + (\delta \times Gradient_A) \tag{4.3}$$

    where δ is the rate of movement

  - If the new positions computed do not yield a configuration with lower stress, or if the new positions violate constraints, the rate is reduced

$$\delta' = \delta/2 \tag{4.4}$$

    till $\delta \geq \delta_t$ where δ_t is a user-defined constant for the lowest acceptable value of δ

- Stop:
  - If no better configuration can be found in spite of decreasing rate and $\delta < \delta_t$
  - Number of iterations crosses *numiter* which is a user-defined constant for the maximum number of iterations allowed for convergence.

Every node in the Strandmap contains text. The text of each node is processed and each node is represented as a term-weight vector, after stemming and stop-word removal. Once we have term-weight vectors of each node, we can find the similarity between any two such vectors using cosine similarity. In addition, WordNet [27] is used to find synonymous words and add a percentage of their weight to total similarity.

$$s_{i,j} = \sum_{k=1}^{n} w_{i,k} \cdot w_{j,k}$$

The dissimilarity can be calculated as $d_{i,j} = 1 - s_{i,j}$ and a dissimilarity matrix is generated by computing the pairwise dissimilarities for all text entries. We also obtain all pairs of pre-requisite constraints from the Strandmap. If node A points to node B in the Strandmap, it implies that A is a pre-requisite of B and therefore should be read/taught before B. Thus $A \rightarrow B$ is a constraint.

After the preprocessing described above, we initialize the CMDS-coordinates of each node.We perform a Topological sort on the nodes of the Strandmap as described in

Cormen et al [24] and obtain an ordered sequence of nodes. Then we assign random coordinates to all these nodes starting with the lowest random number for the first node in the topological ordering and increasing random numbers for each successive node. Note that there might be more than one ordering possible but we just need one for initialization to ensure that pre-requisite constraints are not being violated.

All the nodes have now been assigned some random coordinates. Now we use a Steepest descent approach [65] to minimize stress. The nodes are moved in the direction in which total stress is minimized. If there is a move which violates any pre-requisite constraint, the rate of movement is adjusted till no constraint is violated. If some constraint is still violated, the move is not allowed. Doing this ensures that similar content is pulled together but none of the pre-requisite ordering is destroyed. A *configuration* consists of a set of values of CMDS coordinates for each node in the Strandmap. *Error*(Equation (4.1)) of each configuration is defined as the sum of differences of the distance (computed from the current values of coordinates) and the actual dissimilarity (cosine) for each pair of nodes in the Strandmap. A good configuration is one which has a low value of *Error* because it means that estimated coordinates in CMDS space closely match the actual textual similarities between nodes. We start with the initial value of *Error* and try to obtain configurations with successively lower *Error*. For this we keep updating the coordinates of each point in the hope that this move will lower the overall *Error*. To determine in which direction the point should be moved, we use the *Gradient* as defined in Equation (4.2). The Gradient of a point is the average error between that point and the rest of the points. The value p in the equation decides whether the gradient is positive or negative. Since we are working in 1 dimension, if point A is to the left of point i, we need to increase the value of A's coordinate so that it can move closer to i and decrease the error. So we need an even p to add this error. If however, point A is to the right of point i, we need to decrease the value of A's coordinate and so we need an even p. The Gradient is further divided by the number of points because each point would need to move and dividing thus would give us the amount for a single point. Then we calculate the new coordinate of every point and a new configuration is obtained. If the new configuration obtained does not lead to a lesser Error, or if moving the points thus is violating some constraint in the Strandmap, we discard this configuration and create a new one by decreasing δ. This is done iteratively till we keep getting an configuration with lower error which does not violate any constraints.

If we reach a situation where decreasing δ till its threshold does not help find a better configuration or the maximum number of iterations set by the user has been reached, we stop iterating.

The configuration we end up with gives the locally optimal CMDS coordinates of nodes which keep similar content together without violating any constraints. The nodes are arranged in order of increasing CMDS Coordinates and the order of positions of nodes that we obtain is used to plot the CUTS curve.

Summarization of Strandmaps with CUTS

Constrained MDS gives us the reading order of the Strandmap. The positions of each node obtained by arranging nodes in order of increasing CMDS values is used as the sequence number on the x-axis of the CUTS curve. The MDS coordinates of each node are plotted on the y-axis as before.

Segments from the CUTS curve are obtained as explained in Section 3.4 through 3.4. After segments are obtained, the total summary budget given is distributed among the segments as described in Section 3.5.

Every node in the Strandmap contains a number of sentences. We send a single document to MEAD containing all these sentences and obtain the overall Centroid-score of each sentence as computed by MEAD. This gives an indication of the sentence's importance to the central meaning of the entire Strandmap. A curve segment encompasses one or more nodes. We rank all sentences within a segment according to their Centroid scores.

For obtaining summary of the Strandmap, we consider each segment one by one and keep extracting the top Centroid-score sentences from the segment until the summary budget of that segment is exhausted.

- For every segment Z in the CUTS curve, which has a summary budget B, and

```
  sentences $U = u_1, u_2 ... u_N$ in order of decreasing Centroid scores
• while $numwords(SegmentSummary) < B$
    – $SegmentSummary = SegmentSummary + u_1$
    – $U = U - u_1$ and re-rank
```

A correction was added to this algorithm. If the addition of the last sentence overshoots the summary budget of the segment, a coin-toss is done to decide whether to keep that sentence or not. This technique was added because on the average, this method yielded much larger summaries than the other methods due to presence of multiple segments (If the last sentence overshoots the limit in a large number of segments, it leads to a lot of extra sentences in the summary).

All the segment summaries are combined in order of segments, to create the overall summary of the Strandmap.

Approach II: PageRank With Content

PageRank is an algorithm to estimate the importance of a node in the graph in terms of its structural importance (Section 3.3). In the original algorithm, each outgoing link from a node has an equal probability of being visited ie. the "random surfer" can decide to visit any of the links on a web page with equal probability. In order to combine text with structure, these transition probabilities are biased to reflect similarity between nodes. In this case, the "random surfer" would choose to go to the link which is most similar to the web page he is currently on, in terms of textual content. This makes intuitive sense in the case of Strandmaps as users would like to progress from one concept to another which is similar to it or create lesson plans that transition from one concept to another related or advanced concept. Therefore, in Strandmaps users are more likely to progress from one concept to a similar concept. To reflect this way of navigation, we need to bias the transition probabilities in Pagerank to favor transition between similar concepts.

Instead of distributing Pageranks equally among all outgoing links in the graph, we give a higher rank to similar pages. If there is a forward link from page u to page v, and W

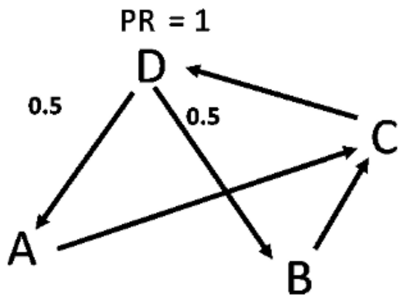


Figure 4.2: Pagerank Example

is the set of all nodes to which u has forward edges,

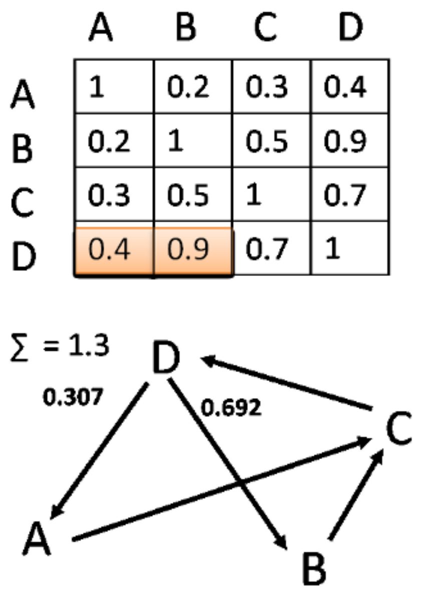
$$TransitionProbability_{u \to v} = \frac{Sim(u,v)}{\sum_{i:i\epsilon W} Sim(u,w)} \tag{4.5}$$

where $Sim(x,y)$ is the cosine similarity between the term-weight representations of web pages x and y given by the formula in Equation (3.1).

Let us consider the graph in Figure 4.2. In the original Pagerank algorithm, if the rank of D was 1, it would be split equally between A and B as 0.5 each. If we used Pagerank with content and used the cosine similarity values given in the matrix in Figure 4.3, the rank of D would not be divided equally but according to their similarities with D. D has outgoing links to A and B and the sum of similarity between D and A and D and B is $0.4 + 0.9 = 1.3$. Thus the rank of A is $\frac{0.4}{1.3}$ and that of B is $\frac{0.9}{1.3}$.

To use this idea for summarization of Strandmaps, we rank all sentences according to their similarity-weighted PageRank and keep extracting the top-ranking sentence until the summary budget is exhausted.

- create $SentenceGraphs$ from the Strandmap as described in Section 4.3 having both Uniform and Biased connectivity.
- Define transition probabilities from Equation (4.5) and find the similarity-weighted Pagerank of each sentence in the Strandmap.
- While(number of words in summary $<$ summary budget), add sentence with highest rank to the summary



	A	B	C	D
A	1	0.2	0.3	0.4
B	0.2	1	0.5	0.9
C	0.3	0.5	1	0.7
D	0.4	0.9	0.7	1

Figure 4.3: Pagerank with Content Example

Chapter 5

EVALUATION

The goal of our evaluation was to find if any of the approaches were creating good summaries of the Strandmap and which approach was creating the most effective summary.

5.1 Evaluation Challenges

The evaluation of text summarization algorithms is usually done by comparing the summary generated by the algorithm with a reference summary. These reference summaries are created by humans and available in Document Understanding Conference(DUC) [9]. However no such reference summaries exist for non-linear narratives like Strandmaps. Moreover there is no established algorithm to create such summaries which can be used as a baseline. Thus we needed to create such benchmark summaries ourselves by conducting user studies. However the inherent problem in these user studies is the factor of human subjectivity and variability. There needs to be a very large sample to establish which sentences constitute a good summary by majority. Another problem is that users might not agree with each other on what constitutes a good summary.

5.2 User Study Design

A number of user studies were conducted which have been detailed in the Appendix. After getting user feedback, there were some biases discovered in each and hence a new study had to be conducted. Thus every study was an iterative improvement over the last one.

For the final user study, we applied the following design lessons which were learnt from the previous studies:

- **Bias introduced by paper-based studies:** The initial studies we conducted were paper-based. The users were shown Strandmaps on the screen while sentences of the Strandmap were listed on sheets of paper. Users were asked to rank each sentence in order of its perceived importance in creating a summary. Sometimes these sentences spanned several pages and we noticed that most often users marked the sentence on the last page with lower ranks. This could be due to the extra effort required to turn pages every time a sentence's rank needed to be decided. It would be a cognitive overload or might not even be possible to remember all sentences thus turning pages was the only way for good comparison. By making it difficult to compare all sentences together, we biased the summaries in favor of sentences on earlier pages. After identifying this bias, the studies were conducted online where it was comparatively easier to scroll back and forth.

- **Interference from overlapping content:** Some of the studies conducted had "Earthquakes' and "Plate Tectonics" as two different Strandmaps. These topics are related and their Strandmaps have a lot of overlapping content, so users already had some residual knowledge of the subject while browsing the map that was presented later. This made them exclude some information already covered in the last map while selecting sentences for creating a summary or discount their importance while ranking sentences. In the final study, the set of Strandmaps presented were non-overlapping and varied in topics.

- **Bias from extra information** The name of the Strandmap eg. "Solar System", "Astronomy" etc. and the listing of the topic strands were visible on the screen. This made users select sentences which had these exact keywords. This bias was identified from user feedback and in later studies no meta information about the Strandmaps was presented.

- **Limited size of human memory** Tasks during the initial studies consisted of selecting a number of sentences from the Strandmap to create a summary or relatively ranking each sentence in the Strandmap in order of importance for creating a summary. Both these tasks required the user to keep all sentences in mind while judging a sentence because the selection or ranking was relative. However, this poses a cognitive overload and users may only be able to rank sentences in relation to those they remember. Thus we redesigned the task to avoid this.

- **Selection Tasks are not reusable** If users are asked to select a group of sentences from the given set, which correspond to a summary size of a particular percentage

Features	**Map 1**	**Map 2**	**Map 3**	**Map 4**	**Map 5**
Name	Cells and Organs	Detecting Flaws in Arguments	Disease	Using Tools and Devices	Waves
SMAP ID	SMS-SMAP-1405	SMS-SMAP-2554	SMS-SMAP-1446	SMS-SMAP-2507	SMS-SMAP-1364
#nodes	25	19	22	24	17
#sentences	32	19	30	24	27
#users	13	12	11	12	12

Table 5.1: Strandmaps for Evaluation

compression of the original text, it cannot be further reused for any other compression percentage.

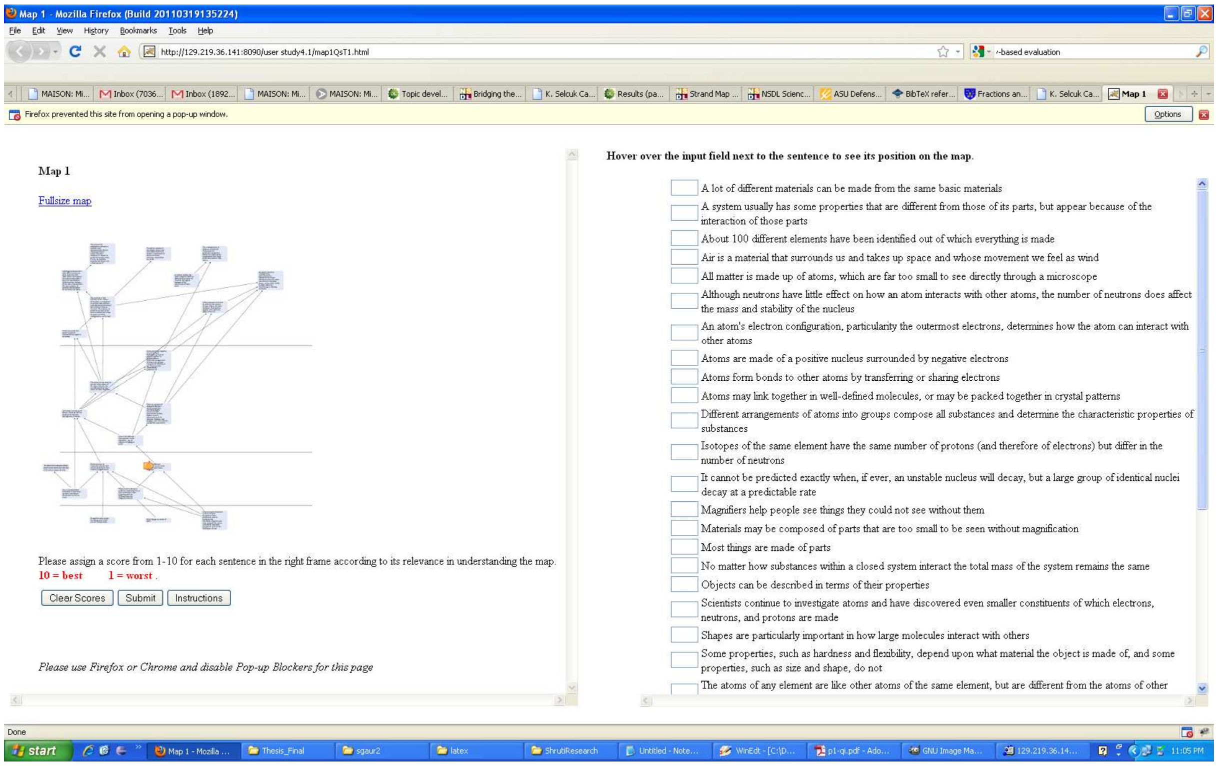
5.3 User Study

The points mentioned in Section 5.2 were incorporated in the design of the final study. The final user study was conducted on 5 Strandmaps from NSDL and responses were received from $11-13$ users per map. Information about the maps selected is in Table 5.1.

The study was available online and Figure 5.1 shows the interface. The left part of the screen showed an NSDL Strandmap and the right part listed all sentences contained in the map. Users were first asked to go through the Strandmap on the left and then asked to score each sentence on the right individually according to its importance in creating a summary of the Strandmap on a scale of $1-10$.

Task:*Please go through the fullsize map(click on the link and zoom it) understand its meaning and assign scores(1 through 10) to each sentence on the right according to its importance in creating the summary of the map. A score of 1 means the sentence is irrelevant and should be omitted in creating the summary and a score of 10 means that the sentence is indispensable for the meaning of the map to be understood properly and should always be included in the summary. You should assign a score to every sentence.*

- Clicking the *Fullsize Map* link on the left showed the Strandmap zoomed to its original size.
- The Strandmap shown did not display a heading, topics or grade levels so that both the algorithm and the human would have the same information. This was done to remove the bias additional information would have on user selection.
- On the right hand, all the sentences contained in the Strandmap were listed in


Map 1 - Mozilla Firefox (Build 20110319135224)
File Edit View History Bookmarks Tools Help
http://129.219.36.141:8090/user study4.1/map1QsT1.html
Firefox prevented this site from opening a pop-up window.
Options
Done
start
11:05 PM

Map 1

Fullsize map

Please assign a score from 1-10 for each sentence in the right frame according to its relevance in understanding the map.
10 = best 1 = worst .

Clear Scores | Submit | Instructions

Please use Firefox or Chrome and disable Pop-up Blockers for this page

Hover over the input field next to the sentence to see its position on the map.

A lot of different materials can be made from the same basic materials
A system usually has some properties that are different from those of its parts, but appear because of the interaction of those parts
About 100 different elements have been identified out of which everything is made
Air is a material that surrounds us and takes up space and whose movement we feel as wind
All matter is made up of atoms, which are far too small to see directly through a microscope
Although neutrons have little effect on how an atom interacts with other atoms, the number of neutrons does affect the mass and stability of the nucleus
An atom's electron configuration, particularity the outermost electrons, determines how the atom can interact with other atoms
Atoms are made of a positive nucleus surrounded by negative electrons
Atoms form bonds to other atoms by transferring or sharing electrons
Atoms may link together in well-defined molecules, or may be packed together in crystal patterns
Different arrangements of atoms into groups compose all substances and determine the characteristic properties of substances
Isotopes of the same element have the same number of protons (and therefore of electrons) but differ in the number of neutrons
It cannot be predicted exactly when, if ever, an unstable nucleus will decay, but a large group of identical nuclei decay at a predictable rate
Magnifiers help people see things they could not see without them
Materials may be composed of parts that are too small to be seen without magnification
Most things are made of parts
No matter how substances within a closed system interact the total mass of the system remains the same
Objects can be described in terms of their properties
Scientists continue to investigate atoms and have discovered even smaller constituents of which electrons, neutrons, and protons are made
Shapes are particularly important in how large molecules interact with others
Some properties, such as hardness and flexibility, depend upon what material the object is made of, and some properties, such as size and shape, do not
The atoms of any element are like other atoms of the same element, but are different from the atoms of other

Figure 5.1: User Study Interface

alphabetical order. This was to remove any perception of importance ordering could imply.

- Users could hover over the sentences in the interface to see their corresponding position on the Strandmap marked by an orange arrow.
- Users were asked to rank each statement individually instead of relatively ranking it or selecting a number of sentences from the entire set. This was done to eliminate relative comparison as it would be difficult to do that on a global level

5.4 Evaluation Measure

After conducting the user study, for every Strandmap, we found the score assigned to each sentence in it by each user. For each sentence in a Strandmap we found the average rating assigned to each sentence in it, across all users. Thus, for a Strandmap S containing sentences $X = x_1, x_2 ... x_N$, and the set of users $U = u_1, u_2 ... u_M$ such that the score assigned to sentence x_i by user u_j is s_{ij}, the average score for a sentence, $AS(x_i)$

is $\frac{\sum_{j:1\rightarrow M} s_{ij}}{M}$. We evaluate a given algorithm by calculating its *User Overlap*. For every sentence in the summary created by the algorithm, we find its average score and add all such average scores to find the User Overlap. If an algorithm A gives a set of sentences $Y = y_1, y_2...y_P$ then User Overlap $= \frac{\sum_{i:i\rightarrow P} AS(x_i)}{M}$.

Since user overlap values for one map cannot be compared with those of another, we calculate the *Normalized User Overlap* which is the user overlap for an approach, divided by the maximum user overlap among all approaches for that Strandmap. Thus the maximum Normalized User Overlap for each map will be 1.

Chapter 6

RESULTS

6.1 User Ratings

Table 6.1 gives the maximum, minimum and average scores users assigned to sentences in each of the Strandmaps.

We observe that the Standard Deviation values in Table 6.1 are very small. This indicates that scores given by users tended to cluster around the average. Thus, scores awarded to sentence on the average were very close to each other. Moreover, users did not use the whole scale from 1 to 10 as implied by the maximum and minimum values.

6.2 User Agreement

ANOVA tests [20] showed that the ratings given by users were not from the same distribution and the Null hypothesis was rejected with more than 99% confidence as shown in Table 6.2.

User disagreement is not surprising given that Strandmaps are complex structures with multiple interconnected topics. So summaries can widely vary within the set of users.

6.3 Experimental Setup

Experiments were carried out to find out the best approach for creating summaries of non-linear narratives like Strandmaps.

	Map 1	**Map 2**	**Map 3**	**Map 4**	**Map 5**
Average	5.89	6.61	6.25	5.92	6.35
Max	7.62	8.42	7.82	8.00	8.17
Min	2.92	4.50	5.18	4.42	4.08
Std.Dev.	1.27	1.01	0.78	0.91	1.25

Table 6.1: Distribution of User Ratings

	Map 1	**Map 2**	**Map 3**	**Map 4**	**Map 5**
P Values	1.93E-17	4.58E-11	8.69E-21	5.04E-28	2.76E-19

Table 6.2: P-Values from ANOVA test for User Agreement

Configurations	Map 1	Map 2	Map 3	Map 4	Map 5	Average
FFF	6.38	6.73	6.25	6.31	6.56	6.45
FFT	6.89	7.27	6.16	5.95	6.40	6.54
FTF	6.07	6.92	6.39	6.08	6.53	6.40
FTT	5.63	7.07	6.69	5.71	6.31	6.28
TFF	6.09	7.52	6.49	5.77	6.01	6.38
TFT	5.98	6.58	6.17	5.44	6.38	6.11
TTF	5.75	7.31	6.30	6.05	6.32	6.35
TTT	5.73	7.19	6.42	6.21	6.13	6.34
Only L	6.42	6.73	6.92	6.18	6.31	6.51
Atul's Config	5.69	6.83	6.11	6.40	6.18	6.24

Table 6.3: Different Configurations for CUTS-based Approach

The 5 Strandmaps described in Table 5.1 were used with each of the algorithms described in Section 4 and a 25% summary was created for each map. This means that the summary retains 25% of the words in the original text. The User Overlap measures for each algorithm were calculated and compared for statistical significance.

The Topic-development based approach uses the formula in Equation (3.3) to assign summary budgets to the left and right children of a node. This formula has many parameters which can be changed to change behavior. We used many different configurations and chose the one with the highest user overlap score in the comparison. These are shown in Table 6.3.

A configuration consists of either inverting a feature or not. There are 3 features -Length, Slope and Average Distance and each feature could either be considered in direct proportion or inverted. Note that Concentration is inverse of Average Distance. A value of T means the feature is inverted and F means it is not inverted. For the first 8 configurations equal weights are assigned to all features ie. $W_L = W_S = W_{AD} = 0.333$. *Only L* means that only Length was used as a parameter. *Atul's Configuration* refers to the configuration that proved most efficient in summarization of text using CUTS [39]. This configuration used features - Length, Inverse-Slope and Average Distance ie. Inverse-Concentration. The weights were $W_L = 0.059$, $W_S = 0.471$ and $W_{AD} = 0.471$

	Map 1	Map 2	Map 3	Map 4	Map 5
Purely Text-based	6.81	**8.04**	**7.08**	6.48	6.63
PageRank without Content (Biased Connectivity)	6.71	7.47	6.98	**6.58**	6.94
PageRank without Content (Uniform Connectivity)	6.72	7.67	6.77	**6.58**	7.29
CUTS-based	**6.89**	7.27	6.16	5.95	6.40
PageRank with Content (Biased Connectivity)	6.72	7.71	6.81	**6.58**	7.49
PageRank with Content (Uniform Connectivity)	6.72	7.71	6.70	**6.58**	**7.50**

Table 6.4: Comparison between Approaches

6.4 Comparison among Approaches

Table 6.4 shows the User Overlap of all approaches for all Strandmaps. The Sentence Graphs constructed for PageRank (without Content) based approach and PageRank with Content are constructed using both Biased and Uniform connectivity as explained in sections 4.3 and 4.3. In the CUTS-based approach, the budgeting as shown in the formula in Equation (3.3) was directly proportional to Length, Slope and Concentration(inverse of average-distance) and equal weights of 0.33 were assigned to each feature i.e. $W_L = 0.33, W_S = 0.33, W_A D = 0.33$, the reason for which has been discussed in Section 6.3.

There are two observations worth noting from these results:

- **All the presented approaches perform better than the random average case** If we compare these techniques in Table 6.4to the average user score in Table 6.1, we see that they have a greater user overlap than the average score assigned by users. This means that all approaches perform better than if we randomly selected sentences to create a summary. This is especially interesting in, as the experiments in Table 6.1 showed, that users themselves disagree what a good summary is. Nevertheless, these results clearly show that Strandmaps have textual and structural features that can be leveraged to obtain good summaries.
- **PageRank with Content performs better than PageRank** The boldfaced numbers in Table 6.4 show the maximum user overlap for that map. We see that PageRank with Content performs equal to or better than PageRank without Content in all maps except Map3. In Figure 6.1 we see that PageRank with Content performs better on

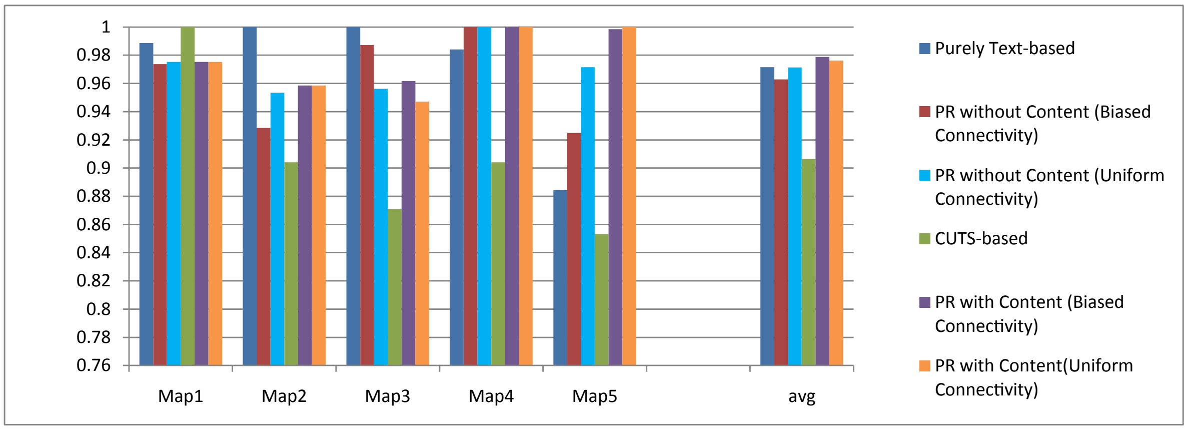


Figure 6.1: Normalized User Overlap Graph

the average, in terms of Normalized User Overlap, than PageRank.

Performing Single factor ANOVA on the results obtained from PageRank without Content(both biased and uniform connectivity), Pure Text and PageRank with Content(both biased and uniform connectivity) with $\alpha = 0.5$ gives a P-value 0.99. Since the P value is greater than 0.5, we cannot reject the *Null hypothesis*. ANOVA on all the approaches (including CUTS-based approach) with $\alpha = 0.5$ gives a P-value 0.57.

The graph showing the Normalized User Overlap is shown in Figure 6.1.

Performing Single factor ANOVA on the normalized user overlap results obtained from PageRank without Content(both biased and uniform connectivity), Pure Text and PageRank with Content(both biased and uniform connectivity) with $\alpha = 0.5$ gives a P-value 0.94 so we cannot reject the *Null hypothesis*. ANOVA on all the approaches (including CUTS-based approach) with $\alpha = 0.5$ gives a P-value 0.04 so we can reject the null hypothesis in this case. Thus we can conclude that CUTS-based approach does not perform as well as the others.

6.5 Graph Rewiring

Strandmaps are educational-concept maps and the links are from lower level to higher level concepts which are related to each other. Graph Rewiring consists of randomly shuffling the edges in a Strandmap. This is done to remove the inherent dependence between text and links in it. The textual descriptions of those nodes could also be very

	Map 1	Map 2	Map 3	Map 4	Map 5
PageRank with Content (Biased Connectivity)	6.72	7.71	6.81	**6.58**	7.49
PageRank with Content after Rewiring(Biased)	5.94	5.93	6.26	5.50	7.06
PageRank with Content (Uniform Connectivity)	6.72	7.71	6.70	**6.58**	**7.50**
PageRank with Content after Rewiring(Uniform)	6.07	5.93	6.26	5.50	7.06

Table 6.5: Effect of Graph Rewiring

similar to each other. We performed a set of experiments by rewiring the graph to break this inherent relationship between text and links in a Strandmap. While the purely-text based approach will not change on rewiring, PageRank with content is expected to decline in performance as the relation between text and structure would break.

We created a list of all the edges in a map and then picked two random numbers. Then we swapped the destinations of the edges at these two indices. We did this a number of times and then added these rewired edges to the graph instead of the original ones. Then we used the PageRank with Content algorithm on this graph. If we compare the user overlap scores of the original PageRank with Content approach and this one (Table 6.5), we do observe that performance deteriorates. This is in accordance with our assumption that breaking the link between text and structure in a Strandmap will lead to worse performance of PageRank with Content approach. If we compare with the average case in Table 6.1 we see that after rewiring, the algorithm sometimes even performs worse than the random case. This shows that in addition to text, structure is also an important part of the Strandmap.

6.6 Conjectures

Interdependence between Text and Structure in Strandmaps

Our conjecture is that text in Strandmaps already reflect the structure. The Strandmaps are so designed that text and structure are intertwined. Keywords are often repeated along a path which make nodes linking to each other somewhat similar in content as well. We believe, this is the reason why text-based approaches and structure-based approaches both provide similar degrees of effectiveness.

PageRank with Content

From the graph shown in Figure 6.1 we see that *PageRank with Content* approach in which the *Sentence Graph* is created using *Biased Connectivity* has maximum *Normalized User Overlap* score. Thus we speculate that this method combines text and structure well even though it does not make any assumption about interdependence between text and structure.

Difference in Performance across Strandmaps

From the graph shown in Figure 6.1 we see that some algorithms perform well in some maps but not others. For example, the CUTS-based algorithm performs well in Map1 but then performs the worst in the rest of the graphs. This might be due to inherent differences in the Strandmaps themselves. Understanding the relationships between Strandmap structures and effectiveness of the approaches considered here requires future studies.

Chapter 7

CONCLUSION

Navigating within non-linear structures is a challenge for all users when the space is large but the problem is most pronounced when the users are blind or visually impaired. Such users access digital content through screen readers like JAWS which read out the text on the screen. However presentation of information in a linear manner without visual cues and information about spatial dependencies is very inefficient for users who are blind.

The NSDL Science Literacy StrandMaps are visual layouts which serve as primary interfaces to help students and teachers browse educational resources. A Strandmap about a specific science or math topic shows relationships between concepts belonging to the topic and how they build upon one another across grade levels.

NSDL Strandmaps are non-linear narratives which need to be presented to users who are blind in an effective way. A good summary of the Strandmap can give the users an idea about the concepts that are explained in that map and help them decide whether they want to view the map or not. In addition, a preview-based navigation mechanism can help users decide which path they want to take from a certain node, based on a preview of nodes further reachable from that path.

Given a non-linear narrative like a Strandmap which has both text and structure, and a word limit w, our goal is to find the best way to create its summary. We propose the following approaches:

- Purely Text-based Approach using a Multi-document Text Summarizer MEAD
- PageRank without Content
- Approaches Combining both Text and Structure
  - CUTS-Based Approach
  - PageRank with Content

Since no reference summaries for such structures were available, we conducted user studies to find human-generated summaries of Strandmaps. These were then evaluated and we found that PageRank with Content Approach, using Biased Connectivity to create Sentence Graphs, combined content and structure well. This method had the highest normalized user overlap score but it was closely followed by Pure Text and PageRank(without Content) based approaches and it could not be statistically proved that the three approaches were different from each other. CUTS-based approach performed the worst on an average, even though it performed the best in one of the maps.

We also came to the conclusion that text and structure are intertwined in a Strandmap by design.

7.1 Future Work

From the results displayed in Figure 6.1 we see that some algorithms perform well in some maps but not so well in others. No algorithm always performs consistently across the maps. We discussed this in conjecture 6.6. This might be due to differences in properties of the maps like structure or similarity of content. This needs to be investigated further in future work.

REFERENCES

[1] http://strandmaps.nsdl.org.

[2] http://www.freedomscientific.com.

[3] http://www.gwmicro.com.

[4] http://www.yourdolphin.com.

[5] http://maison.asu.edu/maison.html.

[6] http://strandmaps.nsdl.org/cms1-2/docs/cms/cms1-2/index.jsp.

[7] http://www.summarization.com/mead.

[8] http://maison.asu.edu/index.jsp.

[9] Document understanding conference. http://duc.nist.gov/.

[10] Micah Adler and Michael Mitzenmacher. Towards compressing web graphs. In *In Proc. of the IEEE Data Compression Conference (DCC*, pages 203–212, 2000.

[11] Rakesh Agrawal and Ramakrishnan Srikant. Fast algorithms for mining association rules. pages 487–499, 1994.

[12] S.T. Ahmed, K.S. Candan, S. Cidambaram, S. Gaur, Jong Wook Kim, Mijung Kim, H. Sundaram, Xinxin Wang, and Renwei Yu. Enabling accessible interfaces to digital library content. In *Multimedia and Expo, 2009. ICME 2009. IEEE International Conference on*, pages 1841 –1842, 28 2009-july 3 2009.

[13] Lars Backstrom, Cynthia Dwork, and Jon Kleinberg. Wherefore art thou r3579x?: anonymized social networks, hidden patterns, and structural steganography. In *Proceedings of the 16th international conference on World Wide Web*, WWW '07, pages 181–190, New York, NY, USA, 2007. ACM.

[14] Regina Barzilay and Michael Elhadad. Using lexical chains for text summarization. In *In Proceedings of the ACL Workshop on Intelligent Scalable Text Summarization*, pages 10–17, 1997.

[15] Daniel K. Blandford, Guy E. Blelloch, and Ian A. Kash. Compact representations of separable graphs. In *Proceedings of the fourteenth annual ACM-SIAM symposium on Discrete algorithms*, SODA '03, pages 679–688, Philadelphia, PA, USA, 2003. Society for Industrial and Applied Mathematics.

[16] P. Boldi and S. Vigna. The webgraph framework i: compression techniques. In *Proceedings of the 13th international conference on World Wide Web*, WWW '04, pages 595–602, New York, NY, USA, 2004. ACM.

[17] Sergey Brin and Lawrence Page. The anatomy of a large-scale hypertextual web search engine. *Comput. Netw. ISDN Syst.*, 30:107–117, April 1998.

[18] K. Selçuk Candan and Wen-Syan Li. Using random walks for mining web document associations. In *Proceedings of the 4th Pacific-Asia Conference on Knowledge Discovery and Data Mining, Current Issues and New Applications*, PADKK '00, pages 294–305, London, UK, 2000. Springer-Verlag.

[19] K. Selçuk Candan and Wen-Syan Li. Discovering web document associations for web site summarization. In *Proceedings of the Third International Conference on Data Warehousing and Knowledge Discovery*, DaWaK '01, pages 152–161, London, UK, 2001. Springer-Verlag.

[20] K. Seluk Candan and Maria Luisa Sapino. *Data Management for Multimedia Retrieval*. Cambridge University Press, New York, NY, USA, 2010.

[21] Deepayan Chakrabarti and Christos Faloutsos. Graph mining: Laws, generators, and algorithms. *ACM Comput. Surv.*, 38, June 2006.

[22] Deepayan Chakrabarti, Christos Faloutsos, and Yiping Zhan. Visualization of large networks with min-cut plots, a-plots and r-mat. *Int. J. Hum.-Comput. Stud.*, 65:434–445, May 2007.

[23] Aaron Clauset, M. E. J. Newman, and Cristopher Moore. Finding community structure in very large networks. *Phys. Rev. E*, 70(6):066111, Dec 2004.

[24] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein. *Introduction to Algorithms*. The MIT Press, New York, 2001.

[25] Inderjit Dhillon, Yuqiang Guan, and Brian Kulis. A fast kernel-based multilevel algorithm for graph clustering. In *Proceedings of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining*, KDD '05, pages 629–634, New York, NY, USA, 2005. ACM.

[26] Peter Eades, Qingwen Feng, Tom Sawyer Software, and Hiroshi Nagamochi. Drawing clustered graphs on an orthogonal grid. In *J. Graph Algorithms Appl*, pages 146–157. Springer-Verlag, 1999.

[27] Christiane Fellbaum, editor. *WordNet An Electronic Lexical Database*. The MIT Press, Cambridge, MA ; London, May 1998.

[28] Gary William Flake, Steve Lawrence, and C. Lee Giles. Efficient identification of web communities. In *Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '00, pages 150–160, New York, NY, USA, 2000. ACM.

[29] Robert M. GagnÃĺ and Richard T. White. Memory structures and learning outcomes. *Review of Educational Research*, 48(2):187–222, Spring 1978.

[30] David Gibson, Ravi Kumar, and Andrew Tomkins. Discovering large dense subgraphs in massive graphs. In *In VLDB Š05: Proceedings of the 31st international conference on Very large data bases*, pages 721–732, 2005.

[31] Michael Hay, Gerome Miklau, David Jensen, Philipp Weis, and Siddharth Srivastava. Anonymizing social networks. Technical report, SCIENCE, 2007.

[32] Ivan Herman, Guy Melançon, and M. Scott Marshall. Graph visualization and navigation in information visualization: A survey. *IEEE Transactions on Visualization and Computer Graphics*, 6:24–43, January 2000.

[33] Eduard Hovy and Chin yew Lin. Automated text summarization and the summarist system. In *In Proc. of the TIPSTER Text Program*, 1998.

[34] George Karypis and Vipin Kumar. Analysis of multilevel graph partitioning. In *Proceedings of the 1995 ACM/IEEE conference on Supercomputing (CDROM)*, Supercomputing '95, New York, NY, USA, 1995. ACM.

[35] George Karypis and Vipin Kumar. A fast and high quality multilevel scheme for partitioning irregular graphs. *SIAM J. Sci. Comput.*, 20:359–392, December 1998.

[36] B. W. Kemighan and S. Lin. An efficient heuristic procedure for partitioning graphs. 1970.

[37] Andrea Kennel, Louis Perrochon, and Alireza Darvishi. Wab: World wide web access for blind and visually impaired computer users. *SIGCAPH Comput. Phys. Handicap.*, pages 10–15, June 1996.

[38] Jon M. Kleinberg. Authoritative sources in a hyperlinked environment. *J. ACM*, 46:604–632, September 1999.

[39] Atul Kolhatkar. Text summarization using topic development patterns.

[40] Ravi Kumar, Prabhakar Raghavan, Sridhar Rajagopalan, and Andrew Tomkins. Extracting large-scale knowledge bases from the web. In *Proceedings of the 25th International Conference on Very Large Data Bases*, VLDB '99, pages 639–650, San Francisco, CA, USA, 1999. Morgan Kaufmann Publishers Inc.

[41] Ravi Kumar, Prabhakar Raghavan, Sridhar Rajagopalan, and Andrew Tomkins. Trawling the web for emerging cyber-communities. In *Computer Networks*, pages 1481–1493, 1999.

[42] Laks V. S. Lakshmanan, Raymond T. Ng, Christine Xing Wang, Xiaodong Zhou, and Theodore J. Johnson. The generalized mdl approach for summarization. In *Proceedings of the 28th international conference on Very Large Data Bases*, VLDB '02, pages 766–777. VLDB Endowment, 2002.

[43] Wen-Syan Li, Okan Kolak, Quoc Vu, and Hajime Takano. Defining logical domains in a web site. In *Proceedings of the eleventh ACM on Hypertext and hypermedia*, HYPERTEXT '00, pages 123–132, New York, NY, USA, 2000. ACM.

[44] Kun Liu and Evimaria Terzi. Towards identity anonymization on graphs. In *Proceedings of the 2008 ACM SIGMOD international conference on Management of data*, SIGMOD '08, pages 93–106, New York, NY, USA, 2008. ACM.

[45] D G Lowe. Three-dimensional object recognition from single two-dimensional images. *Artif. Intell.*, 31:355–395, March 1987.

[46] H. P. Luhn. The automatic creation of literature abstracts. *IBM J. Res. Dev.*, 2:159–165, April 1958.

[47] Mandar Mitra, Amit Singhal, and Chris Buckley. Automatic text summarization by paragraph extraction, 1997.

[48] Jane Morris and Graeme Hirst. Lexical cohesion computed by thesaural relations as an indicator of the structure of text. *Comput. Linguist.*, 17:21–48, March 1991.

[49] Saket Navlakha, Rajeev Rastogi, and Nisheeth Shrivastava. Graph summarization with bounded error. In *Proceedings of the 2008 ACM SIGMOD international conference on Management of data*, SIGMOD '08, pages 419–432, New York, NY, USA, 2008. ACM.

[50] M. E. J. Newman. The structure and function of complex networks. *SIAM REVIEW*, 45:167–256, 2003.

[51] M. E. J. Newman and M. Girvan. Finding and evaluating community structure in networks. *Phys. Rev. E*, 69(2):026113, Feb 2004.

[52] Vol Nr and Per-Olof Fjällström. Algorithms for graph partitioning: A survey, 1998.

[53] L. Page, S. Brin, R. Motwani, and T. Winograd. The pagerank citation ranking: Bringing order to the web. In *Proceedings of the 7th International World Wide Web Conference*, pages 161–172, Brisbane, Australia, 1998.

[54] Charis Papadopoulos and Costas Voglis. C.: Drawing graphs using modular decomposition. In *Graph Drawing. Volume LNCS 3843*, pages 343–354. Springer, 2005.

[55] American Association for the Advancement of Science Project 2061. *AAAS Benchmarks for Science Literacy*. Oxford University Press, New York, 1993.

[56] Yan Qi and K. Selçuk Candan. Cuts: Curvature-based development pattern analysis and segmentation for blogs and other text streams. In *Proceedings of the seventeenth conference on Hypertext and hypermedia*, HYPERTEXT '06, pages 1–10, New York, NY, USA, 2006. ACM.

[57] Dragomir Radev and Adam Winkel. Multi document centroid-based text summarization. In *In ACL 2002*, 2002.

[58] Dragomir R. Radev. Lexrank: graph-based centrality as salience in text summarization. *Journal of Artificial Intelligence Research (JAIR*, 22, 2004.

[59] Dragomir R. Radev, Hongyan Jing, and Malgorzata Budzikowska. Centroid-based summarization of multiple documents: Sentence extraction, utility-based evaluation, and user studies, 2000.

[60] Dragomir R. Radev, Hongyan Jing, Malgorzata Styś, and Daniel Tam. Centroid-based summarization of multiple documents. *Inf. Process. Manage.*, 40:919–938, November 2004.

[61] Sriram Raghavan and Hector Garcia-molina. Representing web graphs. pages 1–10. ACM Press, 2003.

[62] Keith H. Randall, Raymie Stata, Janet L. Wiener, and Rajiv G. Wickremesinghe. The link database: Fast access to graphs of the web. In *Proceedings of the Data Compression Conference*, DCC '02, pages 122–, Washington, DC, USA, 2002. IEEE Computer Society.

[63] Jose F. Rodrigues Jr., Agma J. M. Traina, Christos Faloutsos, and Caetano Traina Jr. Supergraph visualization. In *Proceedings of the Eighth IEEE International Symposium on Multimedia*, ISM '06, pages 227–234, Washington, DC, USA, 2006. IEEE Computer Society.

[64] Nadav Rotem. http://libots.sourceforge.net.

[65] Stuart Russell and Peter Norvig. *Artificial Intelligence: A Modern Approach*. Prentice-Hall, Englewood Cliffs, NJ, 2nd edition edition, 2003.

[66] Torsten Suel and Jun Yuan. Compressing the graph structure of the web. In *Proceedings of the Data Compression Conference*, DCC '01, pages 213–, Washington, DC, USA, 2001. IEEE Computer Society.

[67] Pang-Ning Tan, Michael Steinbach, and Vipin Kumar. *Introduction to Data Mining, (First Edition)*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 2005.

[68] Wen tau Yih, Joshua Goodman, Lucy Vanderwende, and Hisami Suzuki. Multi-document summarization by maximizing informative content-words. In *In Proceedings of IJCAI-07 (The 20th International Joint Conference on Artificial Intelligence*, 2007.

[69] Simone Teufel and Marc Moens. Sentence extraction as a classification task. pages 58–65, 1997.

[70] Yuanyuan Tian, Richard A. Hankins, and Jignesh M. Patel. Efficient aggregation for graph summarization.

[71] Warren Torgerson. Multidimensional scaling: I. theory and method. *Psychometrika*, 17:401–419, 1952. 10.1007/BF02288916.

[72] Satu Virtanen. Clustering the chilean web, 2003.

[73] C. Walshaw and M. Cross. Mesh partitioning: a multilevel balancing and refinement algorithm, 1998.

[74] F. Wu and B. A. Huberman. Finding communities in linear time: a physics approach. *The European Physical Journal B - Condensed Matter and Complex Systems*, 38(2):331–338, March 2004.

[75] Xiaowei Xu. Scan: an structural clustering algorithm for netowrksŤ, to be published. In *in Proc. of 13 th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 1215.

Appendix A

USER STUDIES

A.1 User Study Version 1

In this study users were first shown Strandmaps on the original NSDL interface and then given sheets of paper on which all the sentences present in the Strandmap were listed. Users had to choose a given number of sentences from this list which they thought would be best to describe its summary. The paper form is shown in Figure A.1.

Three Strandmaps were used in this study and 5 responses were collected for each map - 3 at 20% compression and 2 at 40% compression. The Strandmaps used were:

- Classical Mechanics
- Plate Tectonics
- Changes in the Earth's Surface

Lessons Learnt

- For every sentence the user faced the binary decision of selecting it or not. So, there was no way to gauge relative importance of other sentences not selected by the user.
- Due to fixed summary length assigned to each user, these results could not be reused for summaries at different compression amounts.
- Due to a low number of responses and responses being further split by summary sizes, there was very little data for each Strandmap, for each summary length.

A.2 User Study Version 2

Like in the previous study, in this study users were first shown Strandmaps on the original NSDL interface and then given sheets of paper on which all the sentences present in the Strandmap were listed. However this time users had to distinctly rank the given sentences in order of their relevance in describing the summary of the Strandmap starting with 1 as most relevant. The paper form is shown in Figure A.2.

The same 3 Strandmaps were used in this study and 12 users gave their responses for each map.

Lessons Learnt

This study gave us a way to create summaries at multiple compression rates from the ranking given by the users. We could get the relative importance of every sentence in the Strandmap. There was more data for every map. However, we discovered the following problems:

- There were a considerable number of sentences in each Strandmap which spanned several sheets of paper. We found that sentences on the latter pages received worse rankings from most users. This might have been due to the additional effort and break in attention from turning pages back and forth.
- We realized that it was a cognitive overload to keep all sentences in memory and create a ranking. Thus users would have ranked a sentence comparing it just with those they recalled from memory. Thus such methods which required remembering all sentences for fair comparison was an impossible task.

Task type _____________

Please choose ___ sentences from the graph, which best describe its summary. You can mark a X in front of the selected sentences:

The sun's gravitational pull holds the earth and other planets in their orbits, just as the planets' gravitational pull keeps their moons in orbit around them.

The change in motion (direction or speed) of an object is proportional to the applied force and inversely proportional to the mass.

Whenever one thing exerts a force on another, an equal amount of force is exerted back on it.

Gravitational force is an attraction between masses.

The strength of the force is proportional to the masses and weakens rapidly with increasing distance between them.

Any object maintains a constant speed and direction of motion unless an unbalanced outside force acts on it.

Isaac Newton, building on earlier descriptions of motion by Galileo, Kepler, and others, created a unified view of force and motion in which motion everywhere in the universe can be explained by the same few rules.

Newton's system was based on the concepts of mass, force, and acceleration; his three laws of motion relating them; and a physical law stating that the force of gravity between any two objects in the universe depends only upon their masses and the distance between them.

Ptolemy, an Egyptian astronomer living in the second century A.D., devised a powerful mathematical model of the universe based on continuous motion in perfect circles, and in circles on circles.

With the model, he was able to predict the motions of the sun, moon, and stars, and even of the irregular "wandering stars" now called planets.

The Newtonian system made it possible to account for such diverse phenomena as tides, the orbits of planets and moons, the motion of falling objects, and the earth's equatorial bulge.

For several centuries, Newton's science was accepted without major changes because it explained so many different phenomena, could be used to predict many physical events (such as the appearance of Halley's comet), was mathematically sound, and had many practical applications.

Although overtaken in the 1900s by Einstein's relativity theory, Newton's ideas persist and are widely used.

Moreover, his influence has extended far beyond physics and astronomy, serving as a model for other sciences and even raising philosophical questions about free will and the organization of social systems.

No matter how well one theory fits observations, a new theory might fit them just as well or better, or might fit a wider range of observations.

Mathematics provides a precise language to describe objects and events and the relationships among them.

In addition, mathematics provides tools for solving problems, analyzing data, and making logical arguments.

In the long run, theories are judged by the range of observations they explain, how well they explain observations, and how useful they are in making accurate predictions.

Science is based on the assumption that the universe is a vast single system in which the basic rules are everywhere the same and that the things and events in the universe occur in consistent patterns that are comprehensible through careful, systematic study.

Figure A.1: User Study Version 1

Name ____________ **Graph: Astronomy**

If you had to create a summary of this graph, which sentences from it do you think would be most relevant to describe it?

All sentences contained in this graph are listed below. Please rank all of them in order of their relevance to describe the summary of this graph (starting with 1 as most relevant).

You can write numbers in the space provided before each sentence.

The sun's gravitational pull holds the earth and other planets in their orbits, just as the planets' gravitational pull keeps their moons in orbit around them.

The change in motion (direction or speed) of an object is proportional to the applied force and inversely proportional to the mass.

Whenever one thing exerts a force on another, an equal amount of force is exerted back on it.

Gravitational force is an attraction between masses.

The strength of the force is proportional to the masses and weakens rapidly with increasing distance between them.

Any object maintains a constant speed and direction of motion unless an unbalanced outside force acts on it.

Isaac Newton, building on earlier descriptions of motion by Galileo, Kepler, and others, created a unified view of force and motion in which motion everywhere in the universe can be explained by the same few rules.

Newton's system was based on the concepts of mass, force, and acceleration; his three laws of motion relating them; and a physical law stating that the force of gravity between any two objects in the universe depends only upon their masses and the distance between them.

Ptolemy, an Egyptian astronomer living in the second century A.D., devised a powerful mathematical model of the universe based on continuous motion in perfect circles, and in circles on circles.

With the model, he was able to predict the motions of the sun, moon, and stars, and even of the irregular "wandering stars" now called planets.

The Newtonian system made it possible to account for such diverse phenomena as tides, the orbits of planets and moons, the motion of falling objects, and the earth's equatorial bulge.

For several centuries, Newton's science was accepted without major changes because it explained so many different phenomena, could be used to predict many physical events (such as the appearance of Halley's comet), was mathematically sound, and had many practical applications.

Although overtaken in the 1900s by Einstein's relativity theory, Newton's ideas persist and are widely used.

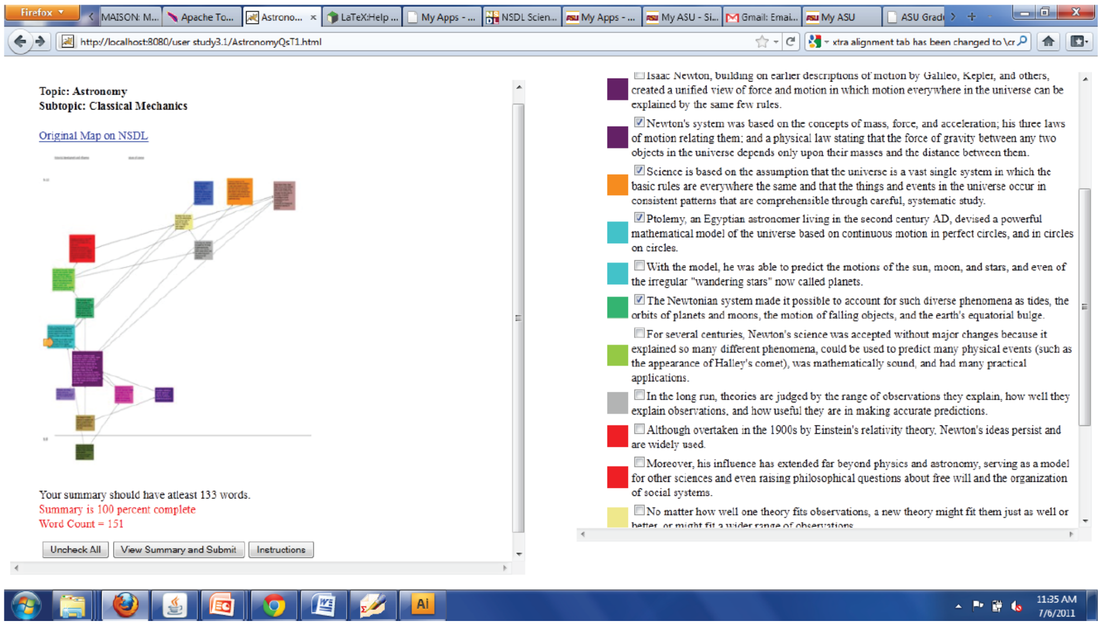
Moreover, his influence has extended far beyond physics and astronomy, serving as a model for other sciences and even raising philosophical questions about free will and the organization of social systems.

No matter how well one theory fits observations, a new theory might fit them just as well or better, or might fit a wider range of observations.

Mathematics provides a precise language to describe objects and events and the relationships among them.

In addition, mathematics provides tools for solving problems, analyzing data, and making logical arguments.

Figure A.2: User Study Version 2



Topic: Astronomy
Subtopic: Classical Mechanics

Your summary should have atleast 133 words.
Summary is 100 percent complete
Word Count = 151

🟪 ☐ Isaac Newton, building on earlier descriptions of motion by Galileo, Kepler, and others, created a unified view of force and motion in which motion everywhere in the universe can be explained by the same few rules.

🟪 ☑ Newton's system was based on the concepts of mass, force, and acceleration; his three laws of motion relating them; and a physical law stating that the force of gravity between any two objects in the universe depends only upon their masses and the distance between them.

🟧 ☑ Science is based on the assumption that the universe is a vast single system in which the basic rules are everywhere the same and that the things and events in the universe occur in consistent patterns that are comprehensible through careful, systematic study.

🟦 ☑ Ptolemy, an Egyptian astronomer living in the second century AD, devised a powerful mathematical model of the universe based on continuous motion in perfect circles, and in circles on circles.

🟦 ☐ With the model, he was able to predict the motions of the sun, moon, and stars, and even of the irregular "wandering stars" now called planets.

🟩 ☑ The Newtonian system made it possible to account for such diverse phenomena as tides, the orbits of planets and moons, the motion of falling objects, and the earth's equatorial bulge.

🟩 ☐ For several centuries, Newton's science was accepted without major changes because it explained so many different phenomena, could be used to predict many physical events (such as the appearance of Halley's comet), was mathematically sound, and had many practical applications.

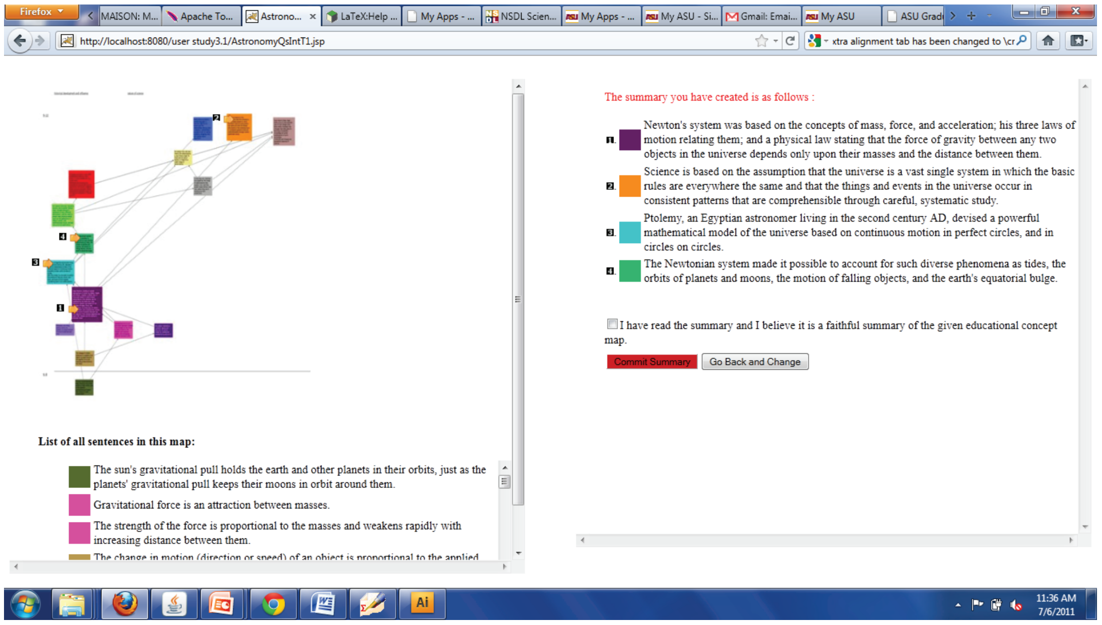
⬜ ☐ In the long run, theories are judged by the range of observations they explain, how well they explain observations, and how useful they are in making accurate predictions.

🟥 ☐ Although overtaken in the 1900s by Einstein's relativity theory, Newton's ideas persist and are widely used.

🟥 ☐ Moreover, his influence has extended far beyond physics and astronomy, serving as a model for other sciences and even raising philosophical questions about free will and the organization of social systems.

🟨 ☐ No matter how well one theory fits observations, a new theory might fit them just as well or better, or might fit a wider range of observations.

Figure A.3: User Study Version 3 Screen 1



List of all sentences in this map:

🟩 The sun's gravitational pull holds the earth and other planets in their orbits, just as the planets' gravitational pull keeps their moons in orbit around them.

🟪 Gravitational force is an attraction between masses.

🟪 The strength of the force is proportional to the masses and weakens rapidly with increasing distance between them.

🟫 The change in motion (direction or speed) of an object is proportional to the applied

The summary you have created is as follows :

1. 🟪 Newton's system was based on the concepts of mass, force, and acceleration; his three laws of motion relating them; and a physical law stating that the force of gravity between any two objects in the universe depends only upon their masses and the distance between them.
2. 🟧 Science is based on the assumption that the universe is a vast single system in which the basic rules are everywhere the same and that the things and events in the universe occur in consistent patterns that are comprehensible through careful, systematic study.
3. 🟦 Ptolemy, an Egyptian astronomer living in the second century AD, devised a powerful mathematical model of the universe based on continuous motion in perfect circles, and in circles on circles.
4. 🟩 The Newtonian system made it possible to account for such diverse phenomena as tides, the orbits of planets and moons, the motion of falling objects, and the earth's equatorial bulge.

☐ I have read the summary and I believe it is a faithful summary of the given educational concept map.

Figure A.4: User Study Version 3 Screen 2

A.3 User Study Version 3

Learning from the ineffectiveness of paper-based studies, we created a study which was available online. The screens for this study are shown in Figures A.3 and A.4.

The users were shown the Strandmap on the left part of the screen and all the sentences in it were listed in the frame on the right. A box before the sentence was color coded to match the color of the node to which the sentence belonged to. The sentences were listed in order such as not to violate the prerequisite ordering in the Strandmap. Users had to pick sentences on the right so as to create a meaningful summary of the map at 25% compression. A word counter was provided for keeping a count of the number of words selected. A Percentage Calculator was also provided which calculated how much of the summary had been created. Only when the percentage was 100% and the word counter exceeded the required minimum number of words, would the users be able to submit the summary. Users could change their selection of sentences to select another set of sentences according to the limit.

Users could visualize the summary at any time using the "View Summary" button. which took them to the next screen where they could "Commit Summary" for final submission after creating the full summary or go back using the "Go Back and Change" button.

We used the same 3 Strandmaps as before and 16 users gave their responses for all 3 maps.

Lessons Learnt

Even though this study consisted of selecting sentences at a particular compression rate and could not be used for other amounts, we thought that the cognitive overload in relatively ranking each sentence was considerably reduced. This was a design tradeoff. We decided we could repeat the study for multiple compression rates. However this study too, suffered from some drawbacks:

- The same Strandmaps had been used in previous studies and some users had attempted all the previous studies. So, there was a problem of residual information that they recalled from past studies.
- The Strandmaps *Plate Tectonics* and *Changes in the Earth's Surface* had overlapping content and users who saw the second map were biased against what they had already learnt from the last map.
- In this interface, the heading of the Strandmap and search keywords were displayed above the map. This lead several users to believe that sentences containing those keywords were more important than the others.
- As was discussed in earlier studies, selection tasks were not reusable but we made a design tradeoff. However in the final study, the task was to rate each sentence instead of relative ranking or selection which was a good solution.

Appendix B

SUMMARIZATION IN MAISON

B.1 Summary Preview for Navigation in NSDL StrandMaps

The Text Summary module in MAISON annotates each edge of the StrandMap with a summary of the content that is reachable (within n hops) of the current node. This is used to give users a text preview of what to expect further along that edge so that they can decide which edges to follow to reach a particular educational resource or learn about certain concepts. For every outgoing edge from the user's current node, the summarization module takes the description text of its immediate neighbor as well its neighbors and generates summaries using MEAD [60] for multi-document centroid-based summarization. The summary obtained from MEAD is used as annotation on that edge. In Figure 1.7, if the user is at node A, there are three outgoing edges. For the rightmost edge, the summary annotation will consist of the summaries of immediate neighbor (B) and its neighbors(C, D,E). The text of all these nodes, which form the lookahead neighborhood for that edge are summarized.As it is expensive to generate summaries in real-time and to have smooth user navigation the summaries are cached internally using a cache.

This system is available at [8] by using "Summary of the Upcoming Nodes" as Link Preview Option.

B.2 MAISON Web Annotation Plugin

The MAISON Web Annotation Plugin annotates links on a Webpage on request to help people who are blind have better navigation experience.

On the Client side, the user sends an annotation request (Keyword/Text Summary) through a Firefox Extension (Addon). Then the Server side extracts text from the requested link, generates the annotation for the link and sends this back to the client as response. The annotation can either be a summary or a set of keywords depending on the request from the client. The text summary is created using Open Text Summarizer [64] which is a very fast text summarizer. Summaries are also cached for faster response time.

Figure B.1 shows the interaction between the Client and Server.

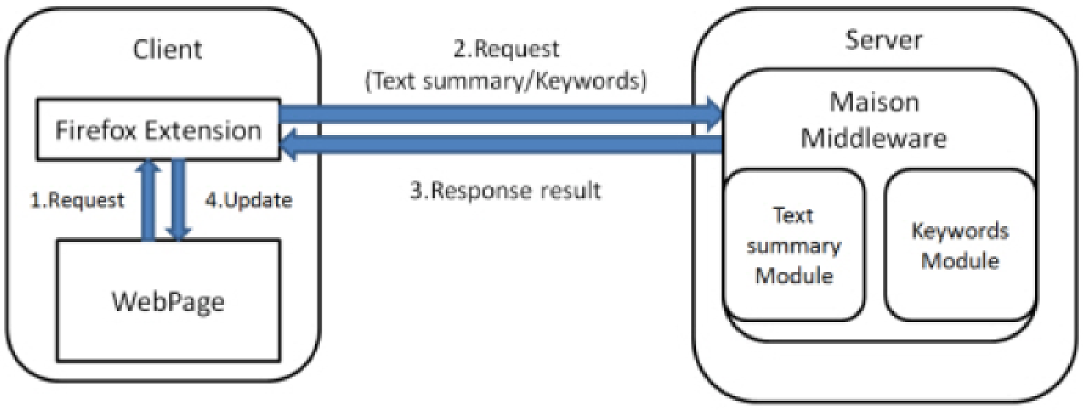


Figure B.1: MAISON Web Annotation System