MiKe: Algorithm for Mining Keyphrases

VINOTHINI KASINATHAN
Faculty of Computer Science and
Information Technology
Universiti Putra Malaysia
43400 UPM Serdang, Selangor
MALAYSIA
vinothini@apu.edu.my

AIDA MUSTAPHA
Faculty of Computer Science and
Information Technology
Universiti Tun Hussein Onn Malaysia
Parit Raja, 86400 Batu Pahat, Johor
MALAYSIA
aidam@uthm.edu.my

CHAN KIN MUN
School of Computing and Technology
Asia Pacific University of
Technology and Innovation
Bukit Jalil, 57000 Kuala Lumpur
MALAYSIA
kinmun@gmail.com

Abstract: The main challenge in mining from presentation slides lies in the fact that slides already contain keywords and keyphrases. A presentation mining system needs a keyphrase extraction algorithm that is able to mine the keywords/keyphrases in the slides and reorganize them from sequential to network-based while keeping the relationships within slides intact. This paper introduces a new keyphrase extraction algorithm called MiKe that extracts keyphrases from a collection of presentation slides and reconstruct the keyphrases into mind map-like visualization output. The descriptions include application of MiKe to a case study that illustrates the flow of MiKe in a presentation mining system.

Key–Words: Keyphrase extraction, Knowledge visualization, Text mining, Powerpoint slide, Mind map

1 Introduction

Microsoft PowerPoint is a well-known software to produce presentation materials with multimedia features that help to make the presentation more attractive. They are organized with pictures and keypoints in bullet forms. Although studies have shown that pictures (even those that have no relation whatsoever with the information presented) give impact on promoting joy in understanding the material [3], it is a major loophole that could mislead the student audience who wrongly interpret the content of presentations. This is because the presenter often exclude descriptions when presenting their key points. Without the narrative, the bullet point structure actually weakens the intention of the presentation.

The presentation slides are also arranged sequentially. [14] believed that in education, the sequential layout in slide-based presentation sometimes does not correspond to students' prior knowledge and understanding because it is common for the lecturer to skip or navigate back and forth the slides during the presentation. This learning process requires the students to reconstruct the logical flow in the presentation materials solely based on their own understanding especially at times where the presenter is not present. This is why the interaction between lecturers and students in conjunction with the slide presentation is at utmost importance. Presentation slides only support the lectures and work best when interaction between lecturers and students is present [13].

This means linearity in slide-based presentation affects learning. Without active involvement from the student during the lecture, this learning method inhibits development of critical thinking which is so essential in the eyes of employers. In bridging this gap, one of the solution to improve learning is by using knowledge visualization. According to [30], knowledge visualization is able to accelerate learning because human brain process images way better than verbal. Human is also known for their capability to recall and utilize pictures [7]. Grounded by this justification, this paper aims to provide a closing loop to the learning process that uses presentation slides in three steps; retrieve the texts from the slides, extract the keyphrases from the texts, and visualize the keyphrases in the form of mind maps.

The remaining of this paper is organized as follows. Section 2 introduces the concept of knowledge visualization, visual learners, and mind maps. Section 3 presents the proposed presentation mining architecture. Section 4 provides the proof of concept to demonstrate the feasibility of the proposed architecture. Section 5 details out the prototype development process based on the proof of concept, Section 6 describes the testing, and finally Section 7 concludes with some indication for future plans.

2 Knowledge Visualization

The main concept of visualization in education is to help students to learn. [26] proposed knowledge visualization technique to strengthen knowledge assimilation in supporting learning. There are several representations available to employ knowledge visualization techniques such as the mind map and the concept map. Picking appropriate representation affects the learning process because the representation has to improve learners' ability in capturing and processing the knowledge [29]. Visualization techniques should also able to transfer knowledge regardless of language [4], thus it is an important channel for sharing knowledge across different language background.

Another related concept is information visualization, which is primarily targeted to visualize the concept using pictures for understanding and learning. This is different from knowledge visualization because this concept strives to help students in analyzing data and detecting patterns [26]. While knowledge visualization enhances and promotes knowledge transfer and exchange between two or more individual more on the mean of communicating experience, information visualization summarizes the patterns that exist within the massive load of data and then produces a new meaning out of the data [27].

A more recent concept that is similar to both mentioned earlier is called visual analytics [4]. [4] suggested that visual analytics emphasizes more on analysing, reasoning and picturing over raw data and information. This is more advanced as compared to information visualization that still rely on human to understand the meaning of pattern derived. Yet, both are applied differently from knowledge visualization that is more focused on conveying knowledge.

2.1 Visual Learners

Each student has a distinct learning style because every individual perceives knowledge in different ways that best suit themselves. The learning style is also influenced by other factors such as prior knowledge and family background. An obvious example of this situation in real life can be seen during university lectures. Many universities including the Asia Pacific University train their students to adapt to different learning styles by diversifying the teaching and learning methods including tutorials and laboratory sessions. Different curriculum activities cater different learning styles to help students to succeed. For example, presentation slides supports the visual learners, verbal communication and explanations supports the auditory learners, and finally tutorial and lab activities support kinaesthetic learners.

Among the three types of learners; visual, auditory, and kinaesthetic, visual learners are said to learn best by getting information input through eyes. A typical visual learner loves colours and likes to study pictures and diagrams such as charts, graphs and maps [8, 19, 2, 21]. [16] claimed that visualization and graphical materials boost the learning experience among visual learners. Graphic organizers supports learning by categorizing and organizing the concept while the graphical visualization tools like the mind map lays out the concepts in a stimulating manner. A study by [18] also reported that visual learners tend to look at diagrams and chart more frequent on the same testing material compared to verbal learners who read more on text but pay less attention on charts. Visual learners are also capable to capture instructions easier when keywords from verbal command are written down and seen visually [25].

Nonetheless, images are always used sparingly in presentation slides because text is still the widely preferred choice [24]. A novice presenter would even cramp all text in one slide, which then be read out word by word. Such presentation incidentally become reading and listening session. Let alone the auditory learners, visual learners could probably unable to follow and absorb huge amount of knowledge in limited time with such stress in their cognitive load. Ineffective teachings may lead to frustration and disappointed among the students; this event would be worsen in an international student community such as the Asia Pacific University [2]. Visual learners also struggle in a large class because they need eye contact and quiet environment to study [23].

Different activities in curriculum that are able to associate with these learning styles can lead student to success [9]. Lecturers are supported by presentation slides for visual learners, verbal communication and explanations for auditory learners. On the other hand, tutorial and laboratory session gives a chance to kinaesthetic learners who learn by doing. However, in this research we are focus on assisting visual learners in their study life due to some weaknesses of PowerPoint slides place visual learners at disadvantage. Visualization materials also include multimedia items such as video and film. This is another potential issue where media can be costly and time consuming to produce plus copyright need to be taken into account for video produced and published by a third party [20].

2.2 Concept Map and Mind Map

One representation used in knowledge visualization is the concept map. Concept map is a graphical interface that is made up of nodes, links, and labels [27, 28]. In the concept map, the key points are represented by several nodes connected to each other via arrows and lines, and labeled with the relationship between the two nodes. Each node and relationship label are keywords that best describe the association concisely, hence presenting the ide as a whole.Research has proven that concept map is an essential tool to assess students' understanding and cultivate their thinking skills [26]. Concept map also assist students to recall and exhibit the memory on paper [22].

[5] introduced the mind mapping approach. Again, mind map is a type of diagram that display ideas by linking keywords that grow out from the central main idea. The creation of mind map is related to the way of how human mind works, for example the fact that the left brain is stronger in perceiving words, numbers and logic while the right brain is better at dealing with colours, music, and imagination. A mind map grows from a central image to capture reader's attention and ideas branching out from the image where each child branch is also represented by keywords or image. The blend of colored images and keywords harvests the potential of human brain to the full extent, bringing both left and right brain together to accept knowledge delivered by the mind map [28].

However, [5] mentioned that a basic mind map does not contain any images and colors. The central image is replaced with a keyword that symbolizes the main theme and surrounded by simple patterns. They believed that the relevancy of the image drawn to the theme and its efficiency to restore the big picture are arguable. At present, most of the mind maps do not follow all the rules as proposed by [5]. Aside from a note-taking tool, mind map can contribute in many ways in education such as knowledge exchange between teachers and students [30]. Mind mapping is easy to construct and the time needed to draw a mind map is far great lesser [6]. With the title being at the centre and keywords grow outwards, mind maps match the way brain manages information [6, 5].

Mind mapping technique is also able to make ideas and relationship between ideas more visible, hence it is very important for students to master the technique. Beyond the educational circle, organizations use mind map in meeting to present issues aside from cultivate brainstorming culture [11]. Drawing mind map manually, however, requires the reader to have certain level of understanding which often result from in-depth reading [1], which takes up a lot of time [15] and becomes more difficult when large amount of text need to be understood. At present, there are a variety of software tools for user to produce their personal mind map such as Text 2 Mind Map (`http://www.text2mindmap.com`) and Mind Meister (`http://www.mindmeister.com`). The problem remains whereas the tools still require manual text input from the users. To resolve this issue, [1] proposed automatic generation of a mind map without the user has to worry about understanding the input text.

3 Presentation Mining

This research attempts to bridge the learning gap that arises from studying the presentation slides. The first concern is the misinterpretation of the original structure from the contents of the slides by the instructor in their teaching sequence and different student reconstruction during self-learning or revision. The second is the ineffectiveness of the presentation slide itself, being text-laden that demotivates learning among the visual learners. Following the concept of "Presentation Mapping" by [12], this research takes up a text mining approach to mine the presentation slides and automatically generate a graphical knowledge display of similar characteristics with a mind map. The proposed framework for "Presentation Mining" is shown in Figure 1. In this framework, a new keyphrase extraction algorithm called MiKe is introduced.

From Figure 1, a presentation mining system consists of three main layers, which are input, core, and output. The system receives a PowerPoint file path to process a collection of presentation slides. Next, the contents of the PowerPoint file are retrieved and are fed to the first stage of the process, which is text preprocessing. Text pre-processing is a mandatory step in text mining to standardise the text format for better mining process. Next, the processed text is sent to MiKe, the keyphrase extraction algorithm. Basically, MiKe extracts keywords and kyphrases from the slides based on the word co-occurrence concept.

Once the MiKe algorithm has extracted the keyphrases from the presentation slides, they are passed forward to the information visualisation stage called the MiKe VisualD. At this stage, before a particular keyphrase is written onto the MiKe VisualD, it is checked against Protégé keyphrases ontology to verify whether extracted keyphrase appears in the domain ontology. Note that domain ontology, although not compulsory, helps to verify the extracted keyphrases. In MiKe VisualD, the keyword or keyphrases are colored blue if the keyphrase exists in the ontology and red is otherwise. However, red-colored keyphrases are not necessarily wrong, but requires further verification such as via semantic approach. The inputs and outputs of each layer in the presentation mining system are illustrated in Table 1.

4 Proof of Concept

In order to demonstrate the feasibility of the proposed presentation mining architecture, a set of proof of concept has been developed. Some terminologies related to PowerPoint are described in Table 2. Next, this paper will show the mapping of a presentation from different number of slides into MiKe VisualD.

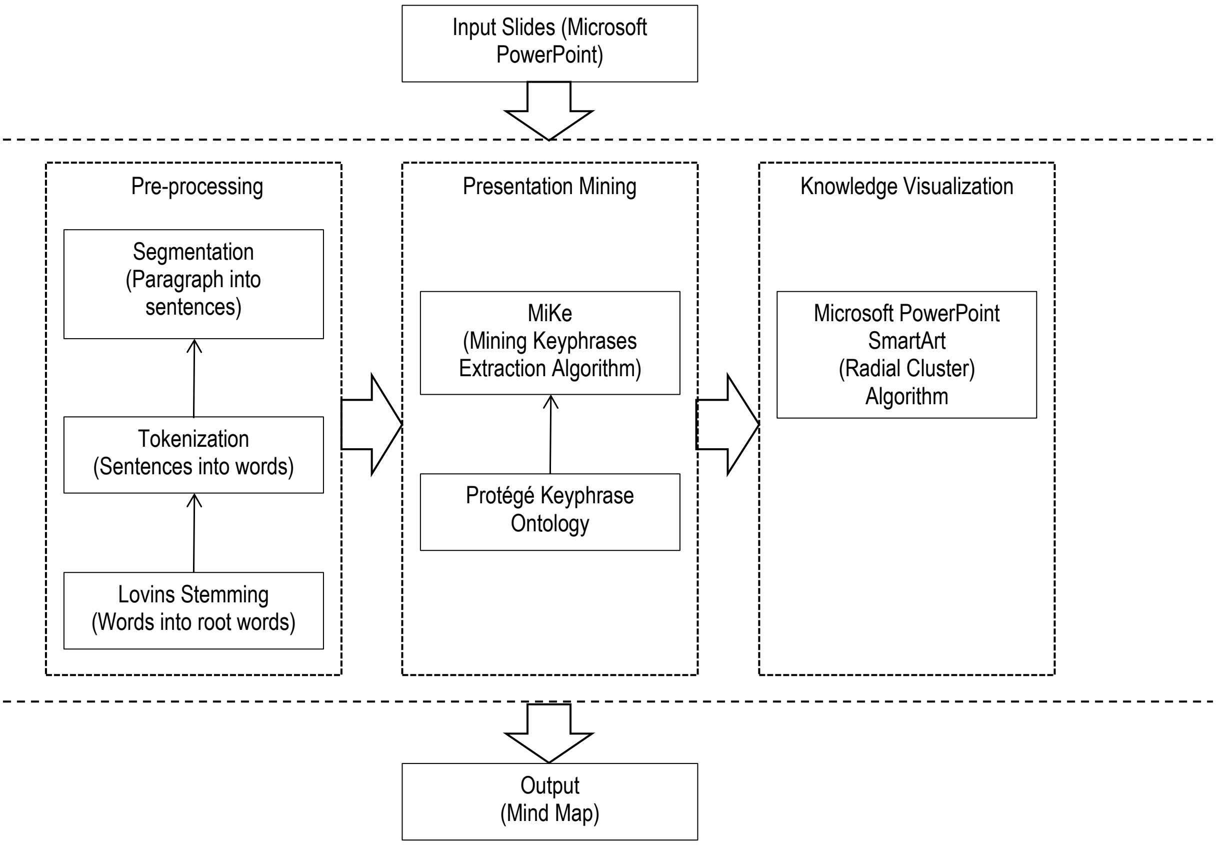
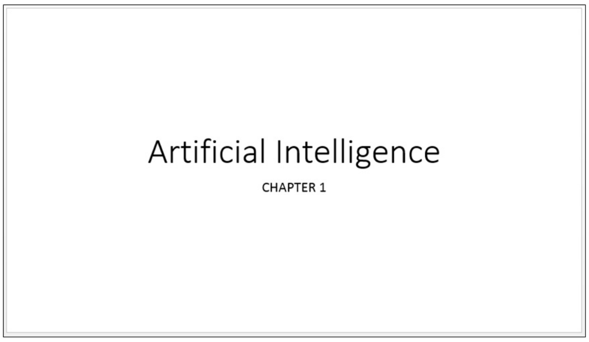

Input Slides (Microsoft PowerPoint)
Pre-processing
Segmentation (Paragraph into sentences)
Tokenization (Sentences into words)
Lovins Stemming (Words into root words)
Presentation Mining
MiKe (Mining Keyphrases Extraction Algorithm)
Protégé Keyphrase Ontology
Knowledge Visualization
Microsoft PowerPoint SmartArt (Radial Cluster) Algorithm
Output (Mind Map)

Figure 1: The framework for presentation mining

4.1 One-slide Input

The first example is mining from one slide as input to the presentation mining system. This will result in the extraction of presentation title from the main slide containing title. To facilitate understanding, the extraction of presentation title is illustrated after having performed the text pre-processing procedures. Figure 2 shows the input slide.



Artificial Intelligence

CHAPTER 1

Figure 2: Input from slide1.xml

The title placeholder has the following xml code contained in the <p:spTree>:

```
<p:sp>
<p:nvSpPr>
  <p:cNvPr id="2" name="Title 1"/>
  <p:cNvSpPr><a:spLocks noGrp="1"/></p:cNvSpPr>
  <p:nvPr><p:ph type="ctrTitle"/></p:nvPr>
</p:nvSpPr>
<p:spPr/>
<p:txBody>
  <a:bodyPr/>
  <a:lstStyle/>
  <a:p>
  <a:r>
  <a:rPr lang="en-US" dirty="0" smtClean="0"/>
  <a:t>Artificial Intelligence</a:t>
  </a:r>
  <a:endParaRPr lang="en-US" dirty="0"/>
  </a:p>
</p:txBody>
</p:sp>
```

Subtitle placeholder has the following xml code contained in a <p:spTree>:

```
<p:nvSpPr>
  <p:cNvPr id="3" name="Subtitle 2"/>
  <p:cNvSpPr><a:spLocks noGrp="1"/>
  </p:cNvSpPr>
  <p:nvPr><p:ph type="subTitle" idx="1"/></p:nvPr>
</p:nvSpPr>
<p:spPr/>
<p:txBody>
  <a:bodyPr/>
```

Table 1: Inputs and outputs for each layer in presentation mining

Stage	Input required		Output generated
Input	– File path of PowerPoint presentation file – Destination path of MiKe VisualD		Titles and contents of each slide are retrieved from the PowerPoint presentation
Modules	Segmentation	Titles and contents of each slide are retrieved from the PowerPoint presentation	Contents being segmented into sentences
	Tokenization	Content in sentences form	Sentences further chunked into words with stop words removed
	Lovins Stemming	Every words chunked from sentences	Stem of each word
	MiKe	Stemmed words	List of words with high co-occurrence weightage
	Knowledge Visualization	Words list	MiKe VisualD
Output	MiKe VisualD		

```
  <a:lstStyle/>
  <a:p>
  <a:r>
  <a:rPr lang="en-US" dirty="0" smtClean="0"/>
  <a:t>CHAPTER 1</a:t>
  </a:r>
  <a:endParaRPr lang="en-US" dirty="0"/>
  </a:p>
</p:txBody>
</p:sp>
```

Note that there are two lines in the XML code being considered. The first line is the presentation title, while the second line is the content. The system will not extract the content from title slide because the branches of the title node must be title of each slides that follows. Hence, it should not sit at the same level as successor slides title. Figure 3 shows the resulting output produced by MiKe VisualD.

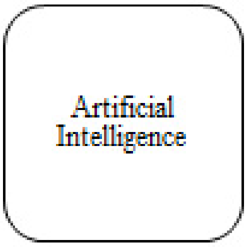
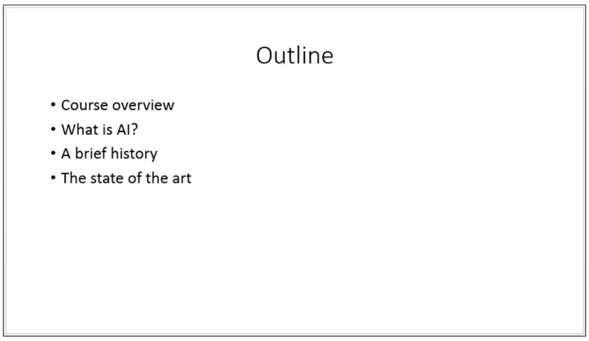


Figure 3: Output from slide1.xml

Similar to a mind map or a concept map, the output from the figure serves as the main node for the graphical representation output. The next section will show the subsequent output when two slides are fed into the system.

4.2 Two-slide Input

The second example is mining from two slides by the presentation mining system. Figure 4 shows the input slide while the code remains the same as slide 1.



Outline

- Course overview
- What is AI?
- A brief history
- The state of the art

Figure 4: Input from slide2.xml

Next, the following code taken from slide2.xml. Title placeholder has the following xml code contained in a <p:spTree>:

Table 2: PowerPoint terminologies

Terminology	Descriptions
Presentation	A file with .pptx (or .ppt) extension. Often, a presentation consists of minimum one slide master.
Slide master	Slide that hold details such as theme and layout of each slide that follows.
Presentation slide	Every single page in a presentation. It contains contents to display in a slideshow.
Slide layout	The structure in which the contents of slide are organised. Each slide layout has at least one text placeholder except blank layout.
Placeholder	A dotted frame box which able to hold text, images, chart and other graphics. By default, it shows messages to guide user things to do with it.

```
<p:sp>
<p:nvSpPr>
<p:cNvPr id="2" name="Title 1"/>
  <p:cNvSpPr>
  <a:spLocks noGrp="1"/>
  </p:cNvSpPr>
  <p:nvPr>
  <p:ph type="title"/>
  </p:nvPr>
</p:nvSpPr>
<p:spPr/>
<p:txBody>
  <a:bodyPr/><a:lstStyle/>
  <a:p><a:pPr algn="ctr"/><a:r>
  <a:rPr lang="en-US" dirty="0" smtClean="0"/>
  <a:t>Outline</a:t>
  </a:r><a:endParaRPr lang="en-US" dirty="0"/>
  </a:p>
</p:txBody>
```

Content placeholder has the following xml code contained in a <p:spTree>:

```
<p:sp>
<p:nvSpPr>
  <p:cNvPr id="3" name="Content Placeholder 2"/>
  <p:cNvSpPr><a:spLocks noGrp="1"/></p:cNvSpPr>
  <p:nvPr><p:ph idx="1"/></p:nvPr>
</p:nvSpPr>
<p:spPr/>
<p:txBody>
  <a:bodyPr/><a:lstStyle/>
  <a:p><a:r><a:rPr lang="en-US" dirty="0"
            smtClean="0"/>
  <a:t>Course overview</a:t>
  </a:r></a:p>
  <a:p><a:r><a:rPr lang="en-US" dirty="0"
            smtClean="0"/>
  <a:t>What is AI?</a:t>
  </a:r></a:p>
  <a:p><a:r><a:rPr lang="en-US" dirty="0"
            smtClean="0"/>
  <a:t>A brief history</a:t>
  </a:r></a:p>
  <a:p><a:r><a:rPr lang="en-US" dirty="0"
            smtClean="0"/>
  <a:t>The state of the art</a:t>
  </a:r><a:endParaRPr lang="en-US" dirty="0"/>
            </a:p>
</p:txBody>
</p:sp>
```

Figure 5 shows the output for two-slide input; one title slide and one title with four bulleted points contents. Since the title has only one word, it is therefore extracted and form the first level node without modification. The content of the slide will be retrieved from the slide and processed with text pre-processing procedures before the keyphrases are taken out to form the second level nodes.

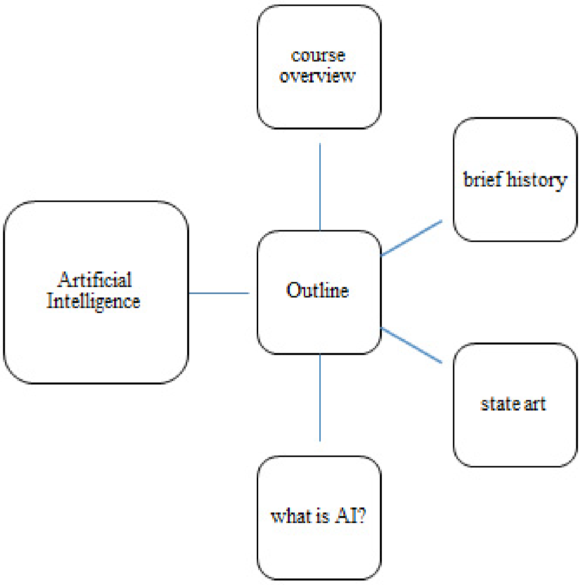
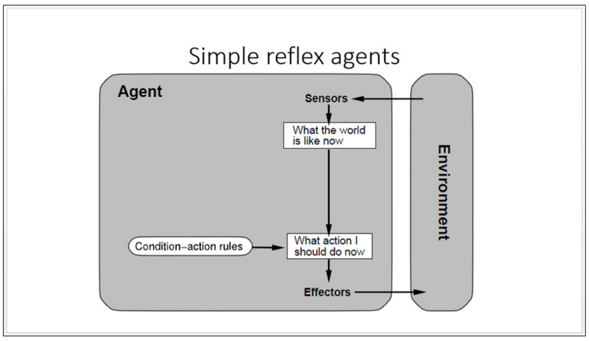


Figure 5: Output from slide2.xml

From the figure, note that the point "What is AI?" becomes the node because the pre-processing module avoid empty node; hence the title as it is.

4.3 Three-slide Input

The third example is mapping from three slides as input to the presentation mining system. Figure 6 shows the input slide.



Simple reflex agents

Figure 6: Input from slide3.xml

Code remains the same for slide 2. The following code taken from slide3.xml. Title placeholder has the following xml code contained in a <p:spTree>:

```
<p:sp>
<p:nvSpPr>
  <p:cNvPr id="2" name="Title 1"/>
  <p:cNvSpPr><a:spLocks noGrp="1"/></p:cNvSpPr>
  <p:nvPr><p:ph type="title"/></p:nvPr>
</p:nvSpPr>
<p:spPr/>
<p:txBody>
  <a:bodyPr/><a:lstStyle/>
  <a:p><a:pPr algn="ctr"/>
  <a:r><a:rPr lang="en-US" dirty="0" smtClean="0"/>
  <a:t>Simple reflex agents</a:t></a:r>
  <a:endParaRPr lang="en-US" dirty="0"/></a:p>
</p:txBody>
</p:sp>
```

Content placeholder has the following xml code contained in a <p:spTree>:

```
<p:pic>
  <p:nvPicPr>
  <p:cNvPr id="4" name="Content Placeholder 3"/>
  <p:cNvPicPr>
  <a:picLocks noGrp="1" noChangeAspect="1"/>
  </p:cNvPicPr>
  <p:nvPr><p:ph idx="1"/></p:nvPr></p:nvPicPr>
  <p:blipFill><a:blip r:embed="rId2"/>
  <a:stretch><a:fillRect/></a:stretch></p:blipFill>
  <p:spPr><a:xfrm>
  <a:off x="1700463" y="1309108"/>
  <a:ext cx="8566484" cy="5246816"/></a:xfrm>
  <a:prstGeom prst="rect"><a:avLst/></a:prstGeom>
  </p:spPr>
</p:pic>
```

Figure 7 shows the output for three-slide input. Only the title is processed (i.e. Simple reflex agent) but it has no child node. The system is able to read only text and also no text is detected under <a:t> tag.

4.4 Four-slide Input

The fourth example is mapping from four slides as input to the presentation mining system. Figure 8 shows the input slide.

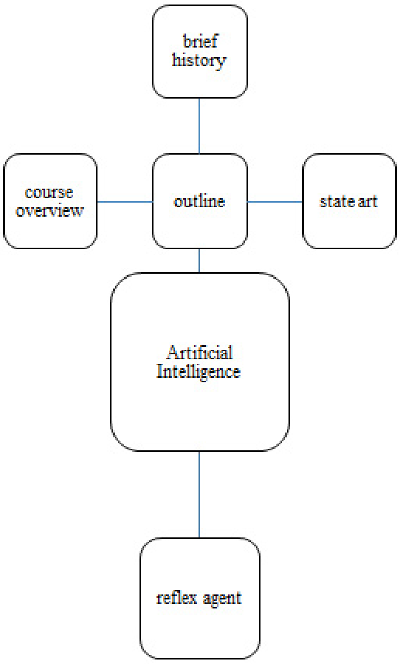
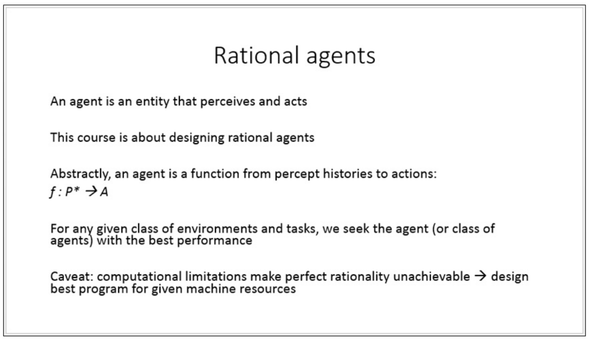


Figure 7: Output from slide3.xml



Rational agents

An agent is an entity that perceives and acts

This course is about designing rational agents

Abstractly, an agent is a function from percept histories to actions:
$f : P^* \rightarrow A$

For any given class of environments and tasks, we seek the agent (or class of agents) with the best performance

Caveat: computational limitations make perfect rationality unachievable → design best program for given machine resources

Figure 8: Input from slide4.xml

The code remains the same for slide 3. The following code taken from slide4.xml. Title placeholder has the xml code contained in a <p:spTree>:

```
<p:sp>
  <p:nvSpPr>
  <p:cNvPr id="2" name="Title 1"/><p:cNvSpPr>
  <a:spLocks noGrp="1"/></p:cNvSpPr>
  <p:nvPr><p:ph type="title"/></p:nvPr>
      </p:nvSpPr><p:spPr/>
  <p:txBody>
  <a:bodyPr/><a:lstStyle/>
  <a:p><a:pPr algn="ctr"/>
  <a:r><a:rPr lang="en-US" dirty="0"
      smtClean="0"/>
  <a:t>Rational agents</a:t></a:r>
  <a:endParaRPr lang="en-US" dirty="0"/>
  </a:p>
  </p:txBody></p:sp>
<p:sp>
<p:nvSpPr>
  <p:cNvPr id="3" name="Content Placeholder 2"/>
  <p:cNvSpPr>
  <a:spLocks noGrp="1"/></p:cNvSpPr>
  <p:nvPr><p:ph idx="1"/></p:nvPr>
</p:nvSpPr>
<p:spPr/>
<p:txBody><a:bodyPr>
  <a:normAutofit fontScale="85000"
     lnSpcReduction="20000"/>
  </a:bodyPr><a:lstStyle/>
  <a:p><a:pPr marL="0" indent="0"><a:buNone/>
  </a:pPr>
  <a:r><a:rPr lang="en-US" dirty="0"
     smtClean="0"/>
  <a:t>An agent is an entity that perceives
     and acts</a:t></a:r></a:p>
  <a:p><a:pPr marL="0" indent="0"><a:buNone/>
  </a:pPr><a:endParaRPr lang="en-US"
     dirty="0"/></a:p>
  <a:p><a:pPr marL="0" indent="0"><a:buNone/>
   </a:pPr>
  <a:r><a:rPr lang="en-US" dirty="0"
      smtClean="0"/><a:t>This course is about
     designing rational agents
</a:t></a:r></a:p>
  <a:p><a:pPr marL="0" indent="0"><a:buNone/>
  </a:pPr><a:endParaRPr lang="en-US" dirty="0"/>
     </a:p>
  <a:p><a:pPr marL="0" indent="0"><a:buNone/>
     </a:pPr>
  <a:r><a:rPr lang="en-US" dirty="0"
     smtClean="0"/>
  <a:t>Abstractly, an agent is a function
     from percept histories to actions:</a:t>
</a:r></a:p>
  <a:p><a:pPr marL="0" indent="0"><a:buNone/>
     </a:pPr>
  <a:r><a:rPr lang="en-US" i="1" dirty="0"/>
  <a:t>f</a:t></a:r>
  <a:r><a:rPr lang="en-US" i="1" dirty="0"
     smtClean="0"/><a:t> : P* </a:t></a:r>
  <a:r><a:rPr lang="en-US" i="1" dirty="0"
     smtClean="0">
  <a:sym typeface="Wingdings"
     panose="05000000000000000000"
     pitchFamily="2" charset="2"/>
</a:rPr>
  <a:t> A</a:t></a:r></a:p>
  <a:p><a:pPr marL="0" indent="0"><a:buNone/>
     </a:pPr>
  <a:endParaRPr lang="en-US" i="1" dirty="0">
  <a:sym typeface="Wingdings"
     panose="05000000000000000000"
     pitchFamily="2" charset="2"/>
  </a:endParaRPr></a:p><a:p><a:pPr marL="0"
     indent="0"><a:buNone/></a:pPr>
  <a:r><a:rPr lang="en-US" dirty="0"
     smtClean="0">
  <a:sym typeface="Wingdings"
     panose="05000000000000000000"
     pitchFamily="2" charset="2"/>
</a:rPr>
  <a:t>For any given class of environments and
     tasks, we seek the agent (or class of
     agents) with the best performance</a:t>
</a:r></a:p>
  <a:p><a:pPr marL="0" indent="0"><a:buNone/>
 </a:pPr>
  <a:endParaRPr lang="en-US" dirty="0">
     <a:sym typeface= "Wingdings"
     panose="05000000000000000000"
       pitchFamily="2" charset="2"/>
  </a:endParaRPr></a:p><a:p>
<a:pPr marL="0" indent="0"><a:buNone/></a:pPr>
  <a:r><a:rPr lang="en-US" dirty="0" smtClean="0">
  <a:sym typeface="Wingdings"
     panose="05000000000000000000"
     pitchFamily="2" charset="2"/>
</a:rPr>
 <a:t>Caveat: computational limitations make
    perfect rationality unachievable design best
    program for given machine resources</a:t>
   </a:r>
   <a:endParaRPr lang="en-US" dirty="0"/></a:p>
</p:txBody>
</p:sp>
```

Figure 9 shows the final output. From the figure, the resulting output from four-slide input appears more like a common presentation design.

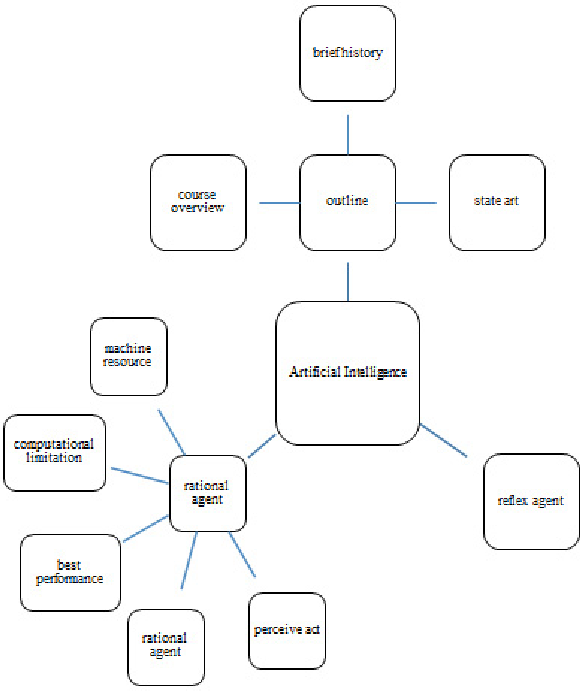


Figure 9: Output from slide4.xml

The newest slide added has five paragraphs and the keyphrases are extracted from these paragraph. The maximum number of keyphrases form the second level nodes, which is five. This is to maintain the readability in MiKe VisualD whereby the higher the number of second level nodes, the smaller the text font size would be. At the end, MiKe VisualD has grown up to two levels with main title situated in the middle and keyphrases radiant out from it.

5 Prototype Development

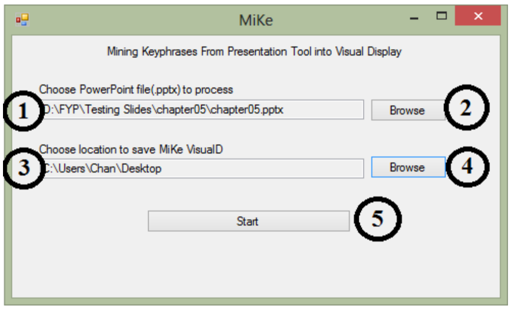
Based on the proof of concept, a prototype of MiKe was developed. Figure 10 shows the main graphical

Implementation of MiKe

The system is built with C# using Microsoft Visual Studio 2013 and Microsoft PowerPoint 2013, both running on 32 bits Windows 8. The system is made up of 12 classes compiled in 9 C# codes. These codes will be discussed in detail in the following sections.

user interface to communicate with the system. The system consists of 12 classes compiled in nine C# codes as discussed in the following paragraphs.

1 **Form1.cs**


MiKe
Mining Keyphrases From Presentation Tool into Visual Display
Browse
Browse
Start
1
2
3
4
5

Choose PowerPoint file(.pptx) to process: D:\FYP\Testing Slides\chapter05\chapter05.pptx

Choose location to save MiKe VisualD: C:\Users\Chan\Desktop

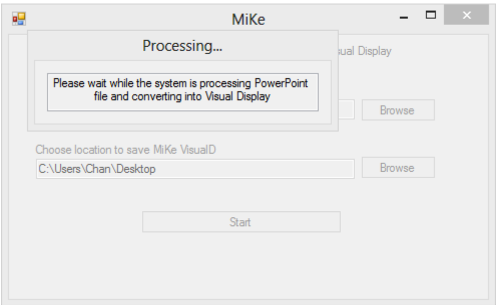
Figure1 Graphical user interface of MiKe

Figure 10: Graphical user interface for MiKe

Figure 1 shows the main graphical user interface to communicate with the system.

Form1.cs Once the start button on MiKeGUI form is clicked and all the criteria are met, the system will start to process. This time, a message is prompted and the MiKeGUI is disabled temporary while the system is running. Once the process completes, the message will be disposed and MiKeGUI is back to active. Figure 11 shows the interface during processing the keyphrases.

Once the start button on MiKeGUI form is clicked and all the criteria are met, the system will start to process. This time, a message is prompted and the MiKeGUI is disabled temporary while the system is running. Once the process completes, the message will be disposed and MiKeGUI is back to active.



Processing...

Please wait while the system is processing PowerPoint file and converting into Visual Display

Figure 2 Wait Message

Figure 11: The wait message

PowerPointProcessor.cs

This algorithm is used to get all titles and slide contents from the presentation file provided by user. It processes slides by slides, retrieve title and follow by slide contents and stores as string. The whole retrieval process requires Open XML SDK to get all of the text from a presentation.

PowerPointProcessor.cs The MiKe algorithm extracts all titles and slide contents from the presentation file provided by user. The process is slide by slide, retrieve title and followed by slide contents and stores as string. The entire process requires OpenXML SDK to get all of the text from a presentation.

```
Algorithm for extracting text from PowerPoint presentation
Begin
Get slide title and raw title from slides
Extract keywords from raw title
Get slide contents
End
```

The code first create an instance of *PresentationDocument* class to represent presentation document from a file path. Instance of *PresentationDocument* class contains *PresentationPart*, which carries the definition for a slide presentation. In addition to that, each slide has a unique slide ID to distinguish from each other and also used as an indicator to access each slides in a presentation. Putting altogether, title and slides contents are accessed slides by slides.

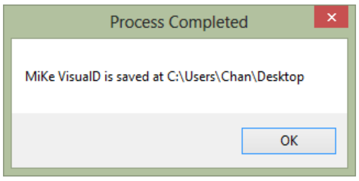
The code first create instance of PresentationDocument class to represent presentation document from a file path. Instance of PresentationDocument class contains PresentationPart, which carries the definition for a slide presentation. In addition to that, each slide has a unique slide ID to distinguish from each other and also used as an indicator to access each slides in a presentation. Putting altogether, title and slides contents are accessed slides by slides.

Each slide titles is then sent to extract keyphrases. This is because certain slides may has long title therefore a keyphrase is needed to represent the slides. Then, the extracted keyphrases from slides titles will go through $if - else$ conditions to examine whether the slide has a single word title, no title or repeated title to sort out necessary actions.

TextAnalyzer.cs This code is the core of the system. It calculates term co-occurrence and its important weightage. The formula was developed by [10] based on the journal article by [17], which utilizes the Jensen–Shannon divergence and the mutual information between keyphrases to group them. Then, it calculates x'^2 value using the formula in Equation 1.

$$x'(w) = \sum_{c \in G} \left\{ \frac{(freq(w,c) - n_w p_c)^2}{n_w p_c} \right\} - \max_{c \in G} \left\{ \frac{(freq(w,c) - n_w p_c)^2}{n_w p_c} \right\} \quad (1)$$

SmartArtDesigner.cs This code is implemented to draw MiKe VisualD. It utilizes a SmartArt template created by the reseacher as a basis and send in keyphrases into each nodes. Before keyphrases are generated into graph, it is checked against Protégé whether it is already existing in the ontology. This leads to node's colour change: red indicates the keyphrase is not existed in the ontology and blue indicates the keyphrase existed in the ontology. Once the process is complete, the message in Figure 12 will be prompted to notify user.



MiKe VisualD is saved at C:\Users\Chan\Desktop

Figure 3 Completion message

Figure 12: The completion message

Code Comparison with Similar Algorithms

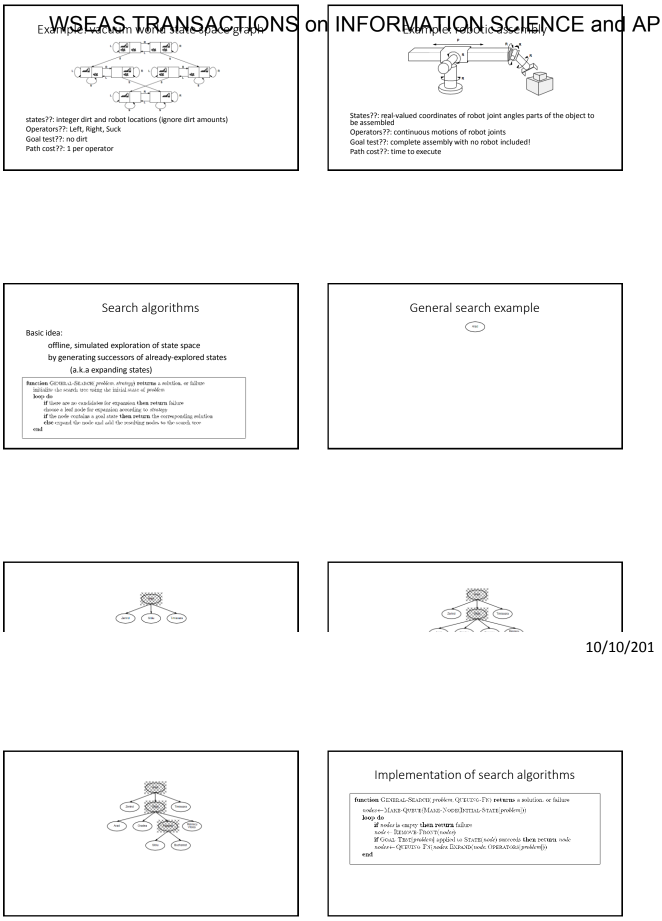
6 Testing

The developer has compared two algorithms throughout the system development lifecycle. First algorithm is cited from Kasinathan, Aida and Mohamad Firdaus (2013) who proposed web-based system called Graphical Knowledge Display to extract keywords from structured-based presentation file. Second algorithm was developed by Jensen (2010) which based on Matsuo and Ishizuka (2003) word co-occurrence algorithm.

Testing was carried out using presentation slides from the Microsoft PowerPoint 2010 and above within the domain of Aritificial Intelligence. The system was developed using the C# programming language with input from the Microsoft PowerPoint 2013, both running on 32 bits Windows 8. Figure 13 and Figure 14 show the input slides from page 13 to 29.

Based on these algorithms, the developer tailors and improves them to suit the system requirements. As a result, the developer finishes the system named MiKe which contains the essences of both renowned algorithms.

For better understanding, these three algorithms will be compared in the following tables. The following part presents an overview comparison of the three algorithms. Afterward, comparisons among each algorithms by pair are showed in detailed.



Example: vacuum world state space graph

states??: integer dirt and robot locations (ignore dirt amounts)
Operators??: Left, Right, Suck
Goal test??: no dirt
Path cost??: 1 per operator

Example: robotic assembly

States??: real-valued coordinates of robot joint angles parts of the object to be assembled
Operators??: continuous motions of robot joints
Goal test??: complete assembly with no robot included!
Path cost??: time to execute

Search algorithms

Basic idea:
offline, simulated exploration of state space
by generating successors of already-explored states
(a.k.a expanding states)

```
function GENERAL-SEARCH( problem, strategy) returns a solution, or failure
  initialize the search tree using the initial state of problem
  loop do
     if there are no candidates for expansion then return failure
     choose a leaf node for expansion according to strategy
     if the node contains a goal state then return the corresponding solution
     else expand the node and add the resulting nodes to the search tree
  end
```

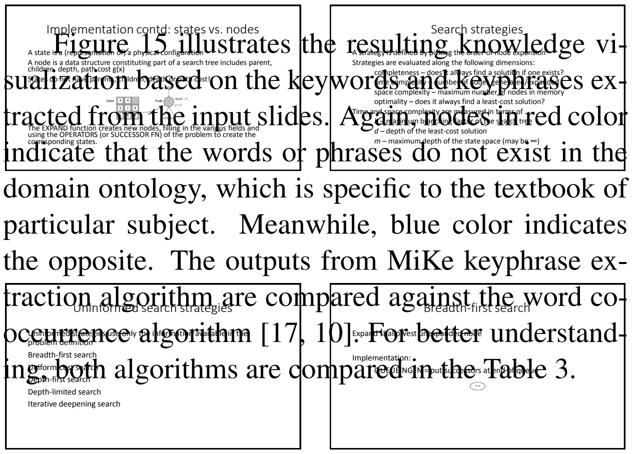
General search example

Implementation of search algorithms

```
function GENERAL-SEARCH( problem, QUEUING-FN) returns a solution, or failure
  nodes ← MAKE-QUEUE(MAKE-NODE(INITIAL-STATE[problem]))
  loop do
     if nodes is empty then return failure
     node ← REMOVE-FRONT(nodes)
     if GOAL-TEST[problem] applied to STATE(node) succeeds then return node
     nodes ← QUEUING-FN(nodes, EXPAND(node, OPERATORS[problem]))
  end
```

Figure 13: Input slides page 13 to 20

Figure 15 illustrates the resulting knowledge visualization based on the keywords and keyphrases extracted from the input slides. Again, nodes in red color indicate that the words or phrases do not exist in the domain ontology, which is specific to the textbook of particular subject. Meanwhile, blue color indicates the opposite. The outputs from MiKe keyphrase extraction algorithm are compared against the word co-occurrence algorithm [17, 10]. For better understanding, both algorithms are compared in the Table 3.



Implementation contd: states vs. nodes

A state is a (representation of) a physical configuration
A node is a data structure constituting part of a search tree includes parent, children, depth, path cost g(x)
States do not have parents, children, depth or path cost!
The EXPAND function creates new nodes, filling in the various fields and using the OPERATORS (or SUCCESSOR FN) of the problem to create the corresponding states.

Search strategies

A strategy is defined by picking the order of node expansion
Strategies are evaluated along the following dimensions:
completeness – does it always find a solution if one exists?
time complexity – number of nodes generated/expanded
space complexity – maximum number of nodes in memory
optimality – does it always find a least-cost solution?
Time and space complexity are measured in terms of
b – maximum branching factor of the search tree
d – depth of the least-cost solution
m – maximum depth of the state space (may be ∞)

Uninformed search strategies

Uninformed strategies use only the information available in the problem definition
Breadth-first search
Uniform-cost search
Depth-first search
Depth-limited search
Iterative deepening search

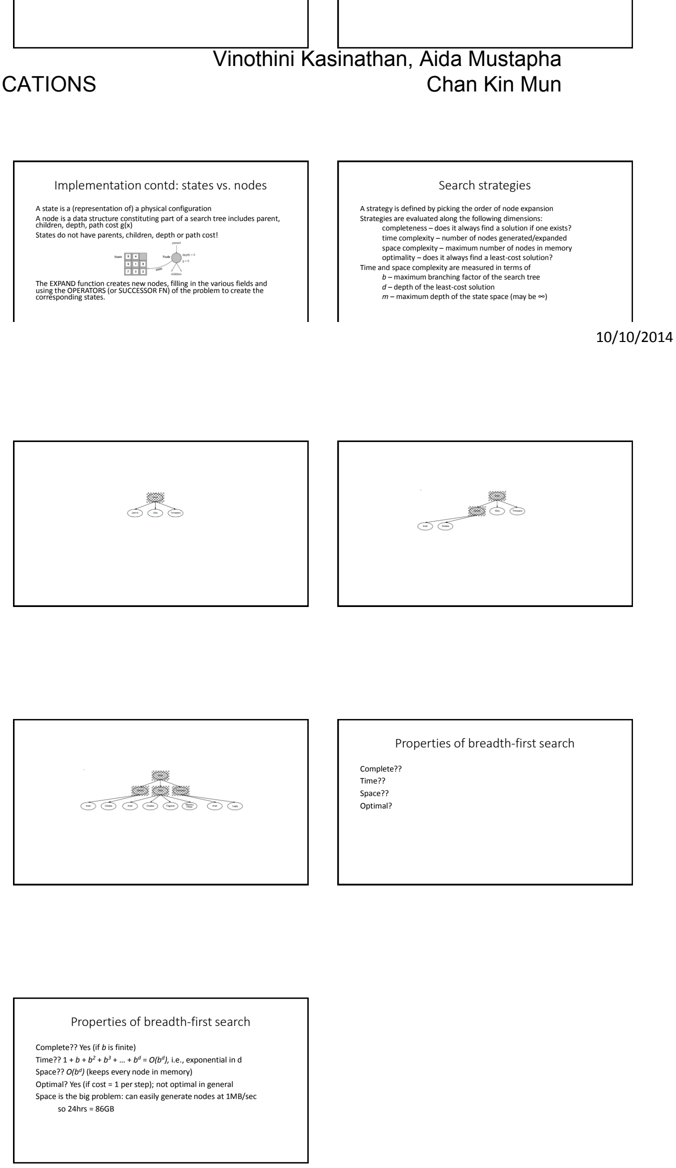
Breadth-first search

Expand shallowest unexpanded node

Implementation:
QUEUEINGFN = put successors at end of queue

7 Conclusion

This paper introduced the concept of “Presentation Mining” that reconstruct the content from a PowerPoint slide into mind maps, which is then used as a pedagogical tool for learning. This is in response to the deteriorating effectiveness in slide-based presentation among the visual learners. A detailed account for a presentation mining algorithm called MiKe (Mining Keyphrases) is also provided. When MiKe returns a collection of candidate keyphrases, the algorithm scans through and selects keyphrases found matches in the Protégé ontology. There is no arguments on taking keyphrases that exist in the ontology, but selecting keyphrases with two terms that appear first not only



Implementation contd: states vs. nodes

A state is a (representation of) a physical configuration
A node is a data structure constituting part of a search tree includes parent, children, depth, path cost g(x)
States do not have parents, children, depth or path cost!

The EXPAND function creates new nodes, filling in the various fields and using the OPERATORS (or SUCCESSOR FN) of the problem to create the corresponding states.

Search strategies

A strategy is defined by picking the order of node expansion
Strategies are evaluated along the following dimensions:
completeness – does it always find a solution if one exists?
time complexity – number of nodes generated/expanded
space complexity – maximum number of nodes in memory
optimality – does it always find a least-cost solution?
Time and space complexity are measured in terms of
b – maximum branching factor of the search tree
d – depth of the least-cost solution
m – maximum depth of the state space (may be ∞)

Properties of breadth-first search

Complete??
Time??
Space??
Optimal?

Properties of breadth-first search

Complete?? Yes (if *b* is finite)
Time?? $1 + b + b^2 + b^3 + \ldots + b^d = O(b^d)$, i.e., exponential in d
Space?? $O(b^d)$ (keeps every node in memory)
Optimal? Yes (if cost = 1 per step); not optimal in general
Space is the big problem: can easily generate nodes at 1MB/sec
so 24hrs = 86GB

Figure 14: Input slides page 21 to 29

wasting a keyphrase node for irrelevant keyphrases but also ignoring more important keyphrases. In the future, this research plans to improve the performance of the keyphrase extraction algorithm in producing the final mind map. Currently, MiKe is only able to process two-word keyphrases (i.e., Artificial Intelligence) and ignore three-word keyphrases (i.e. Artificial Neural Network). This must be improved because keyphrases within the domain of Artificial Intelligence covers up to four-word keyphrases (i.e., randomized weighted majority algorithm, full joint probability distribution).

At present, MiKe is able to deal with standard slides titles and special cases including repeating titles and empty titles. For slides with repeating title, keyphrases extracted from both slides were combined and grouped under the same branch in MiKe VisualD. For empty title, the slides is regarded as continued from the previous slides, therefore the keyphrases were grouped under the same branch as well. The resulting graphical knowledge visualization output

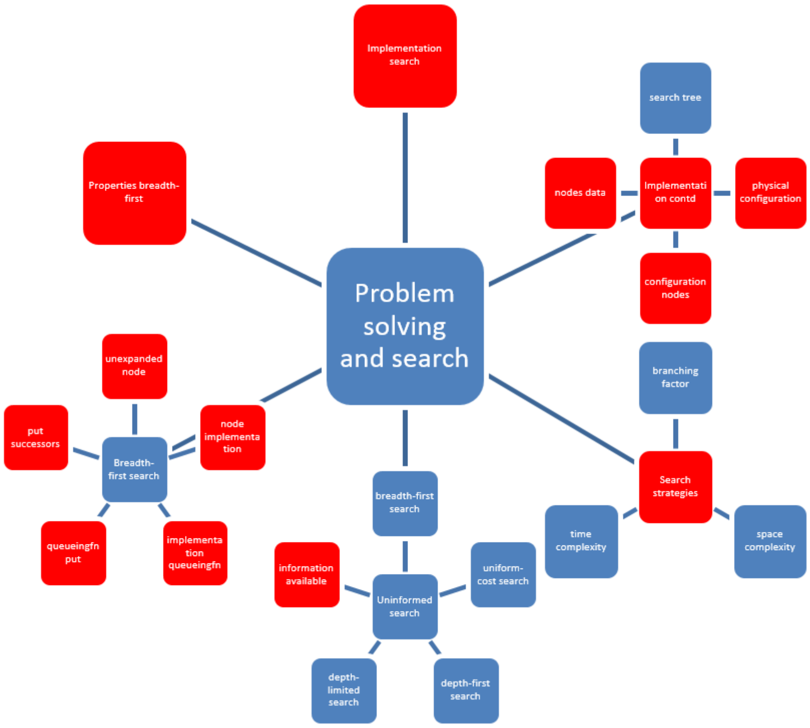


Figure 15: Output for slides 13 to 29

adopts the basic concept from mind maps [5] but it is different from other mind map drawing tools in place where MiKe serves. The proposed presentation mining system has an automated mind map drawing functionality whereby the concepts are extracted via a keyphrase extraction algorithm rather than manually being fed by users like Text 2 Mind Map.

MiKe VisualD generates the mind map back into PowerPoint presentation (.pptx) format, therefore, promoting a single format from input to output. The choice of software platform is also justified by the current use among the target students, which are at university level. Students have the freedom and chance to modify the output from MiKe VisualD. Finally, although the current SmartArt designer algorithm produced commendable output, readability is inversely proportional to number of keyphrases. Therefore, as the number of keyphrases increases, the size of each nodes decreases as well as the size of font. In future, a clustering task to group the mined contents is desired before the keyphrases extraction phase takes place so the word co-occurrence weightage can be calculated across many slides and is not just limited to one slide.

Acknowledgements: This project is sponsored by Universiti Putra Malaysia under Geran Putra–Inisiatif Putra Siswazah (GP/IPS/2013/9399844).

References:

[1] M Abdeen, R El-Sahan, A Ismaeil, S El-Harouny, M Shalaby, and MCE Yagoub. Direct automatic generation of mind maps from text with m 2 gen. In *Science and Technology for Humanity (TIC-STH), 2009 IEEE Toronto International Conference*, pages 95–99. IEEE, 2009.

[2] Mohamad Jafre Zainol Abidin, Abbas Ali Rezaee, Helan Nor Abdullah, and Kiranjit Kaur Balbir Singh. Learning styles and overall academic achievement in a specific educational system. *International Journal of Humanities and Social Science*, 1(10):143–152, 2011.

[3] Robert A. Bartsch and Kristi M. Cobern. Effectiveness of powerpoint presentations in lectures, 2003.

[4] Stefan Bertschi, Sabrina Bresciani, Tom Crawford, Randy Goebel, Wolfgang Kienreich, Martin Lindner, Vedran Sabol, and Andrew Vande Moere. What is knowledge visualization? perspectives on an emerging discipline. In *Information Visualisation (IV), 2011 15th International Conference on*, pages 329–336. IEEE, 2011.

Table 3: Output comparison between two algorithms

Item	Jensen [10]	MiKe
Features	Extract keywords from a body of text	Extract keyphrases from Power Point, treating each slide as a body of text individually
Input	Text file of the body of text	Microsoft PowerPoint slides (.pptx)
Output	Keywords display on command prompt	Microsoft PowerPoint slides (.pptx)
Result	Top 10 keywords from the body of text	A series of Microsoft PowerPoint SmartArt mind maps
Programming language	C#	C#
Application type	Command prompt application	Desktop application with GUI

[5] Tony Buzan, Barry Buzan, and James Harrison. *The mind map book: Unlock your creativity, boost your memory, change your life*. Pearson BBC Active, 2010.

[6] Vincent Chik, Beryl Plimmer, and John Hosking. Intelligent mind-mapping. In *Proceedings of the 19th Australasian conference on Computer-Human Interaction: Entertaining User Interfaces*, pages 195–198. ACM, 2007.

[7] Zhong Da and Zhang Jianping. Knowledge visualization-an approach of knowledge transfer and restructuring in education. In *Information Technology and Applications, 2009. IFITA'09. International Forum on*, volume 3, pages 716–719. IEEE, 2009.

[8] Grace Fleming. Learning styles: Know and use your personal learning style. *About. com Guide. Last accessed on*, 30, 2010.

[9] Abbas Pourhosein Gilakjani. Visual, auditory, kinaesthetic learning styles and their impacts on english language teaching. *Journal of Studies in Education*, 2(1):104–113, 2011.

[10] T Jensen. Keyword extraction in c# with word co-occurrence algorithm, 2010.

[11] Tang Jeung-tai and Chiang Chihui. Organizational knowledge sharing through mind mapping. In *Fuzzy Systems and Knowledge Discovery, 2009. FSKD'09. Sixth International Conference on*, volume 2, pages 305–309. IEEE, 2009.

[12] V. Kasinathan, A. Mustapha, and M.F.C.A. Rani. Structure-based algorithm for presentation mapping in graphical knowledge display. *International Journal of Information and Education Technology*, 3(2):196–200, 2013.

[13] Ian M Kinchin, Deesha Chadha, and Patricia Kokotailo. Using powerpoint as a lens to focus on linearity in teaching. *Journal of further and Higher Education*, 32(4):333–346, 2008.

[14] I.M. Kinchin and L.B. Cabot. Using concept mapping principles in powerpoint. *European Journal of Dental Education*, 11(4):194–199, 2007.

[15] R Kudelic, Mladen Konecki, and M Malekovic. Mind map generator software model with text mining algorithm. In *Information Technology Interfaces (ITI), Proceedings of the ITI 2011 33rd International Conference on*, pages 487–494. IEEE, 2011.

[16] Claudia Leopold and Detlev Leutner. Science text comprehension: Drawing, main idea selection, and summarizing as learning strategies. *Learning and Instruction*, 22(1):16–26, 2012.

[17] Yutaka Matsuo and Mitsuru Ishizuka. Keyword extraction from a single document using word co-occurrence statistical information. *International Journal on Artificial Intelligence Tools*, 13(01):157–169, 2004.

[18] Tracey J Mehigan, Mary Barry, Aidan Kehoe, and Ian Pitt. Using eye tracking technology to identify visual and verbal learners. In *Multimedia and Expo (ICME), 2011 IEEE International Conference on*, pages 1–6. IEEE, 2011.

[19] Lori S Mestre. Matching up learning styles with learning objects: what's effective? *Journal of Library Administration*, 50(7-8):808–829, 2010.

[20] Dennis R Myers, Catherine Sykes, and Scott Myers. Effective learner-centered strategies for teaching adults: using visual media to engage the adult learner. *Gerontology & Geriatrics Education*, 29(3):234–238, 2008.

[21] Usha G Shenoy, Karthiyane Kutty, VMS Shankar, and Nachal Annamalai. Changes in the learning style in medical students during their mbbs course. *International Journal of Scientific and Research Publications*, 2(9):1–4, 2012.

[22] Jane Alexen Shuyska and Chris Davies. Thinkspace: the collaborative process of designing new technologies for the classroom. In *Proceedings of the 4th International Symposium on Wikis*, page 11. ACM, 2008.

[23] Stephen M Stahl and Richard L Davis. Applying the principles of adult learning to the teaching of psychopharmacology: overview and finding the focus. *CNS Spectr*, 14(4), 2009.

[24] Judy Strauss, Hope Corrigan, and Charles F Hofacker. Optimizing student learning: Examining the use of presentation slides. *Marketing Education Review*, 21(2):151–162, 2011.

[25] Maren Urner, Dietrich Samuel Schwarzkopf, Karl Friston, and Geraint Rees. Early visual learning induces long-lasting connectivity changes during rest in the human brain. *Neuroimage*, 77:148–156, 2013.

[26] Minhong Wang and Michael J Jacobson. Guest editorial-knowledge visualization for learning and knowledge management. *Educational Technology & Society*, 14(3):1–3, 2011.

[27] Wang Xiao-yue and Mu Yan. The using of knowledge visualization tools in e-science environment take concept maps for example. In *Control, Automation and Systems Engineering, 2009. CASE 2009. IITA International Conference on*, pages 39–42. IEEE, 2009.

[28] Zhou Yan, Wang Lei, and Yin Liqun. A research for the classification of knowledge visualization. In *Electrical and Control Engineering (ICECE), 2011 International Conference on*, pages 6235–6238. IEEE, 2011.

[29] Fangfang Zhang. The application of visualization technology on knowledge management. In *Intelligent Computation Technology and Automation (ICICTA), 2008 International Conference on*, volume 2, pages 767–771. IEEE, 2008.

[30] Jianping Zhang, Da Zhong, and Jiahua Zhang. Knowledge visualization: An effective way of improving learning. In *Education Technology and Computer Science (ETCS), 2010 Second International Workshop on*, volume 1, pages 598–601. IEEE, 2010.