Mathematical and Software Engineering, Vol. 6, No. 2 (2020), 12-18.
Varεpsilon Ltd, http://varepsilon.com/index.php/mse/

BFS algorithm and SADT in Knowledge Management in E-learning on Data Structures

Valentina Dyankova[1] **and Milko Yankov**[2]

[1]Department of Mathematics and Computer Science, Konstantin Preslavsky University of Shumen, Bulgaria, v.spasova@shu.bg.
[2]Reward Gateway (UK) Ltd - Branch, Plovdiv, Bulgaria, milkoyankov@gmail.com

Abstract

The successful acquisition of knowledge and skills about the conceptual apparatus in a given subject area requires the concepts to be studied in mutual connection and conditionality. This requires a very precise refinement of the logical relationship between them. The present study presents an approach to building a formal model of the semantic relationship between the concepts in the course on Data Structures. Emphasis is placed on the adaptation of the breadth-first search (BFS) algorithm to the established formal model in order to plan and optimize training time. The created training plan contains the minimum volume of necessary basic learning units, which allow the learner to move to the acquisition of a concept by Data Structures.

Keywords: BFS algorithm; SADT methodology; e-Learning; Data Structures.

1 Introduction

The recently increasing use of information and communication technologies in education is characterized by widely available web-based educational resources, increased demand for lifelong learning opportunities, increased attention to the quality of education and the time to acquire knowledge and skills in the relevant subject area. These characteristics determine the need for both the provision of systematized learning materials for learners and the optimal administration of the learning process. These two activities must be planned according to the goals set by the learner. These are activities that require significant time resources, which can be reduced with the introduction of technological solutions based on effective knowledge management.

The purpose of this article is to provide an approach to knowledge management in online training on Data Structures. Two of the most important conditions for a competitive learning process are the design of logically connected information flows, guaranteeing quality knowledge acquisition and the possibility for optimal distribution of learning time. The development of the methodology used for formalization of the learning process by Data Structures in the form of an oriented graph is presented and the possibility of using the breadth-first search (BFS) algorithm for planning and optimizing the training time is emphasized.

The ideas presented in the development are based on the experience of the authors in the development of an online training system on Data Structures DSLearning (www.dslearning.eu) and the training of computer science competitors (students and pupils), students who study individually, students who start working as interns in software companies; people who retrain as programmers, novice programmers.

2 Using the SADT methodology to build a model of e-learning by data structures

For each of the above groups of people are characterized by the following features:

Clear goal – striving to acquire specific knowledge of Data Structures.

Different background – the presence of different systems of already acquired knowledge, skills and habits in Data Structures.

Striving to optimize training time – the goal is to achieve the desired results in minimum time.

The preparation of a training plan, consistent with the individual indicators of the learner for the listed characteristics, requires a structured system that integrates the concepts in the respective subject area and the logical connections between them. The logical connections between the concepts must reflect their conditionality – for the assimilation of some concepts the knowledge of others is used.

Therefore, for the formalization of this system it is necessary to choose a methodology that perceives the system through the perspective of the information that flows through it. One such approach is top-down analysis, which underlies the Structured analysis and design technique (SADT) [1]. SADT is a software engineering methodology for describing systems as a hierarchy of functions and is successfully used to solve a wide range of tasks such as long-term strategic planning, automated production and others. SADT is used even in areas such as security in Internet communication [2], providing effective solutions in company management [3], modeling and design of space probes [4].

SADT is used in the present development in order to create a formal model (graph) that implements a specific functional view, describing the concepts of Data Structures and their relationships. The underlying SADT analysis "top-down" allows for the study of concepts, starting with their most general overview, subsequent detailing of the functionality of each concept and hierarchical organization of levels in the resulting system of concepts. In the created formal model for each concept are defined levels of knowledge, corresponding to the specifics of the acquired knowledge and intellectual initiative of the learners. In their most general form, these levels are:

Level A – Formation of a logical idea of the concept, work with the definition and objects.

Level B – Acquisition of knowledge and skills about the properties of the concept, as well as discovery of new ones.

Level C – Formalization of the concept in the respective software system.

Level D – Acquisition of knowledge and skills for free use of the software model.

Level E – Acquisition of knowledge and skills for modeling real practical processes and expanding the functionality of the concept in terms of a specific software system.

Each of these levels has a hierarchical structure and includes other levels. Since the process of building the overall hierarchical system is not the subject of this article, they will not be considered.

The system of concepts, organized in this way, contributes to the successful acquisition of knowledge about the concepts of data structures in the ideology of integrity and interdependence between them. The vertices in the oriented graph, which is a formal model of this system, represent the concepts studied. Two vertices P1 and P2 are connected by an oriented edge from P1 to P2, if the knowledge of the concept P1 is necessary for the acquisition of the knowledge of the concept P2. Figure 1 shows an example of a simplified part of the graph connecting knowledge of several concepts.

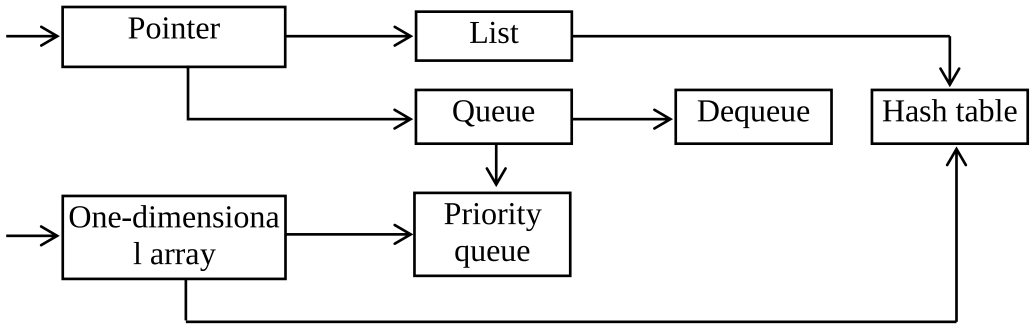

Figure 1: Connection between concepts

The graph shown in Figure 1 do not reflect the considered levels of knowledge acquisition, as the purpose of the present study is to present its use, and not to detail the process of its acquisition. However, the information about the levels of knowledge will be specified in the context of its functionality to the problem under consideration and will be reflected in the software implementation in the next section.

3 BFS algorithm as an algorithmic strategy for learning process management

After the already created formal model of functional dependence between the concepts in Data Structures, the question arises for assessing the way of acquiring knowledge about a given concept from the respective level. In order to optimize the training time, this time should not contain unnecessary waypoints. This draws attention to finding the minimum path in a graph by the number of vertices involved. The problem of finding the minimum path in a graph by the number of vertices naturally leads to the use of the BFS algorithm [5].The BFS algorithm is the basis of many developments with high practical value such as optimizing difficult to execute queries in a large database [6], creating an algorithm for searching for bird food [7], mapping the best route to achieve fast and adequate home movements for the elderly [8], efficient use of cluster resources in CPU-GPU based cluster calculations [9] and many others.

The use of the BFS algorithm to optimize the time to obtain a certain knowledge of a given level requires appropriate formalization of the necessary information. This information will be presented in the form of Java classes. Each Java class will be described only with the data/methods needed to understand the overall concept of using BFS to solve the problem in question.

Each concept is represented as an object of class Concept:

```
public class Concept {
  // term/name of the concept
  private String term;
  // description of the acquired competencies after mastering the concept
  private String description;
  public Concept(String term, String description) {
    this.term = term;
    this.description = description;
  }
  ...
}
```

Each level of the hierarchical structure of knowledge for a given concept is represented as an object of class LevelOfKnowledge, which inherits from class Concept:

```
public class LevelOfKnowledge extends Concept{
  // level of knowledge
  private String level;
  // list of tasks that must be covered to acquire knowledge
  private ArrayList<String> tasks;
  // list of required levels of knowledge required for this level of knowledge
  private HashSet<LevelOfKnowledge> requiredKnowledge;
  public LevelOfKnowledge(String term, String description, int level,
                              ArrayList<LevelOfKnowledge> list) {
      super(term,description);
      this.level = level;
      tasks = new ArrayList<String>();
      requiredKnowledge = new HashSet<LevelOfKnowledge>(list);
  }
  ...
}
```

The adjacency list for a concept (vertex in the graph) is presented as an object of class AdjacencyList. All neighbors of a vertex present the concepts directly related to it (knowledge of the respective level). The edges have weights equal to the time to acquire knowledge of these concepts. The quantitative assessment, which is assigned to each edge from vertex P1 to vertex P2, is the value of time in minutes for acquiring knowledge and skills from the respective level for the concept P2 with already acquired knowledge and skills from the required level for the concept P1. To estimate this time, the authors use information from the DSLearning Data Structures online training system created by them. At each acquisition of new knowledge by each trainee registered in the system, information about the time spent by the trainee for acquiring the new knowledge is kept in the administrator panel of the system. The quantitative assessment of each of the edges in the graph is assumed to be equal to the arithmetic mean of the times of all registered users so far. The concepts from the adjacency list and the times for their assimilation are associated in pairs <level of knowledge about the concept, time>, which are integrated in HashMap. The time component is not a criterion for minimizing the path, but it is then used to calculate an estimate of the training time in the shortest path found by the number of vertices.

```
public class AdjacencyList{
  private HashMap<LevelOfKnowledge,Integer> list;
  public AdjacencyList() {
      list = new HashMap<LevelOfKnowledge,Integer>();
  }
  ...
}
```

The graph itself is represented as a HashMap of associated pairs <knowledge at a given level for a concept, adjacency list>. The BFSPath method of the Graph class adapts the BFS algorithm to the presented formal knowledge model. The method returns a HashMap from associated pairs. The first component contains the vertex of the graph through which it passes. The second component in each pair contains the precursor in the path found at the vertex of the first component. In this way, the found path can be restored.

```
public class Graph{
  private HashMap<LevelOfKnowledge, AdjacencyList > graph;
  public AdjacencyList() {
     list = new HashMap<LevelOfKnowledge,Integer>();
  }
  public HashMap<LevelOfKnowledge,LevelOfKnowledge> BFSPath(LevelOfKnowledge
                start, LevelOfKnowledge goal) throws InterruptedException {
     LinkedBlockingQueue<LevelOfKnowledge> queue =
                               new LinkedBlockingQueue<LevelOfKnowledge>();
    HashMap<LevelOfKnowledge,LevelOfKnowledge> used =
                          new HashMap<LevelOfKnowledge,LevelOfKnowledge>();
    queue.put(start);
    used.put(start,null);
    while(!queue.isEmpty() && !used.containsKey(goal)) {
      int sizeQueue = queue.size();
      for(int currentNum = 0; currentNum < sizeQueue; currentNum++) {
        LevelOfKnowledge currentConcept = queue.poll();
        AdjacencyList list = graph.get(currentConcept);
        if(list != null) {
           Iterator<Entry<LevelOfKnowledge, Integer>> it =
                                   list.getList().entrySet().iterator();
           while(it.hasNext()) {
             Entry<LevelOfKnowledge, Integer> item = it.next();
             if(!used.containsKey(item.getKey())) {
                 queue.put(item.getKey());
                 used.put(item.getKey(),currentConcept);
             }
           }
        }
      }
    }
    return used;
   }
  ...
}
```

The BFSPath method returns a HashMap from associated pairs. The first component contains the vertex of the graph through which it passes. The second component in each pair contains the precursor in the path found at the vertex of the first component. In this way, starting from the final vertex in the path and following the predecessors, all the vertices of the path can be obtained. The following program fragment does this.

```
HashMap<LevelOfKnowledge,LevelOfKnowledge> map = g.BFSPath(start,goal);
LevelOfKnowledge current = goal;
while(current != start) {
     action with the vertex current
     current = map.get(current);
}
```

4 Conclusion

The received training plan is in accordance with the profile of the trainee. This profile contains information about all already acquired levels of knowledge, which allows to prepare an individual training plan. This approach cannot completely replace the role of the instructor in this process, as in many cases quite subjective characteristics for the learner have to be considered. However, it frees the instructor from the routine and laborious task of determining the sequence of basic topics based on the complex and voluminous connections in the knowledge column. One possible further improvement may be to consider the time weights of the edges in the knowledge graph. The BFS algorithm can be integrated with the idea of partially minimizing actual training time.

References

[1] Marca, D., A., McGowan, C., L. (1987) SADT: Structured Analysis and Design Techniques, McGraw-Hill.

[2] Quan, L., Guo, Q., Chen, H., Xie, X., Li, X., Liu, Y., Hu, J., SADT: Syntax-Aware Differential Testing of Certificate Validation in SSL/TLS Implementations, 35th IEEE/ACM International Conference on Automated Software Engineering (ASE), 21-25 Sept. 2020, Melbourne, VIC, Australia.

[3] LAKHOUA, M., N., KAROUI, M., F., Monitoring of a Production System based on Functional and Dysfunctional Analysis, Journal of Computer Science and Control Systems, 2019, Volume 12, Issue 1.

[4] Lobanov, A., Strogankova, N., Bolbakov R., Meta-modeling of Space Probe On-Board Computing Complexes, International Conference on High-Performance Computing Systems and Technologies in Scientific Research, Automation of Control and Production, 15-16 May 2020, Barnaul, Russia.

[5] Nakov, P., Dobrikov, P. (2018) Programming = ++ Algorithms.

[6] Mondal, S., Mukherjee, N., A BFS-Based Pruning Algorithm for Disease-Symptom Knowledge Graph Database, Information and Communication Technology for Intelligent Systems, Smart Innovation, Systems and Technologies, vol 107. Springer, Singapore. Online ISBN 978-981-13-1747-7.

[7] Zhang, Z., Huang, C., Dong, K., Huang, H., Birds foraging search: a novel population-based algorithm for global optimization, Memetic Computing, 11, 221–250 (2019).

[8] Pinheiro, PR, Pinheiro, PGCD, Filho, RH, Barrozo, JPA, Rodrigues, JJPC, Pinheiro, LICC, Pereira, MLD, Integration of the Mobile Robot and Internet of Things to Monitor Older People, IEEE Access, Volume 8, 2020, ISSN**:** 2169-3536.

[9] Chandrashekhar, BN, Sanjay, HA, Srinivas, T., Performance Analysis of Parallel Programming Paradigms on CPU-GPU Clusters, International Conference on Artificial Intelligence and Smart Systems (ICAIS), 25-27 March 2021, Coimbatore, India.