การพัฒนาโปรแกรมช่วยจัดแผนการเลือกวิชาลงทะเบียนโดยใช้เทคนิคการค้นหาเฉพาะที่และรายการทาบู

Registration Assistant Application using Local Search and Tabu List Technique

ธาดา หวังธรรมมั่ง[1] เสกสรรค์ สุวรรณมณี[2] ธัชชัย เองฉ้วน[3] และแสงสุรีย์ วสุพงศ์อัยยะ[4*]

Thada Wangthammang[1], Seksun Suwanmanee[2], Touchai Angchuan[3] and Sangsuree Vasupongayya[4*]

บทคัดย่อ

งานวิจัยนี้มีวัตถุประสงค์เพื่อออกแบบและพัฒนาโปรแกรมประยุกต์เว็บเป็นเครื่องมือช่วยจัดแผนการศึกษาให้แก่นักศึกษาในระดับอุดมศึกษาที่ประสบปัญหาทางการเรียน โดยใช้หลักสูตรวิศวกรรมศาสตรบัณฑิต สาขาวิศวกรรมคอมพิวเตอร์ หลักสูตรปรับปรุง พ.ศ. 2553 เป็นกรณีศึกษา การค้นหาทาบูถูกประยุกต์ใช้เป็นเครื่องมือค้นหาหลัก ผู้ใช้สามารถกำหนดวัตถุประสงค์ในการวางแผนการเรียนของตนและโปรแกรมที่พัฒนาขึ้นจะใช้วัตถุประสงค์ดังกล่าวเพื่อค้นหาแผนการศึกษา นอกจากนี้เงื่อนไขทั่วไปในการจัดแผนการศึกษา เช่น จำนวนหน่วยกิต สถานะของนักศึกษา และแผนการศึกษามาตราฐานถูกนำมาผนวกเข้าในระบบเพื่อค้นหาแผนการศึกษาที่เหมาะสมให้ผู้ใช้เลือก ผลการทดสอบระบบเบื้องต้นพบว่าผู้ใช้พึงพอใจในส่วนติดต่อผู้ใช้เนื่องจากมีความคล้ายโปรแกรมประยุกต์ของกูเกิ้ล อย่างไรก็ตามผลการเรียนของนักศึกษาควรถูกนำเข้าระบบแบบอัตโนมัติโดยตรงจากระบบของมหาวิทยาลัยเพื่อลดความผิดพลาดในการกรอกข้อมูล

คำสำคัญ: การจัดกำหนดการ เครื่องมือช่วยนักศึกษา แผนการศึกษา หลักสูตรวิศวกรรมคอมพิวเตอร์

Abstract

This project aims to design and develop a web application as a tool for generating a study plan for higher education students with study issues. The Bachelor of Engineering in Computer Engineering curriculum 2010, Prince of Songkla University was used as a case study. Tabu search was applied as the main search tool. The users can define the objective of their study plan and the proposed program will uses the user defined objective

[1] นักศึกษาปริญญาโท ภาควิชาวิศวกรรมคอมพิวเตอร์ มหาวิทยาลัยสงขลานครินทร์ สงขลา 90110

[2] อ., ภาควิชาวิศวกรรมคอมพิวเตอร์ มหาวิทยาลัยสงขลานครินทร์ สงขลา 90110

[3] ผศ., ภาควิชาวิศวกรรมคอมพิวเตอร์ มหาวิทยาลัยสงขลานครินทร์ สงขลา 90110

[4] ผศ.ดร., ภาควิชาวิศวกรรมคอมพิวเตอร์ มหาวิทยาลัยสงขลานครินทร์ สงขลา 90110

[1] Graduate student, Master degree, Department of Computer Engineering, Prince of Songkla University, Songkhla, 90110

[2] Lecturer, Department of Computer Engineering, Prince of Songkla University, Songkhla, 90110

[3] Asst. Prof., Department of Computer Engineering, Prince of Songkla University, Songkhla, 90110

[4] Asst. Prof. Dr., Department of Computer Engineering, Prince of Songkla University, Songkhla, 90110

[*] Corresponding author: Tel.: 074-287360. E-mail address: sangsuree.v@psu.ac.th

to guide the search. Furthermore, common constraints in creating a study plan such as the number of credits, the student status and the standard study plan are all included into the system in order to find a suitable solution for the user to make his/her selection. The preliminary evaluation results showed that the user satisfied with the user interface because it was similar to that of the Google application interface. However, the study results should be directly downloaded from the university system in order to reduce the data entry mistakes.

Keywords: Scheduling, Student Assistant Tool, Study Plan, Computer Engineering Curriculum

Introduction

Each curriculum in the university has a unique set of requirements such as the co-requisite and prerequisite which each student in the curriculum must follow. Furthermore, each university must enforce some rules and regulations on the student registration process in order to ensure the student status and ability to graduate. In addition, each student may not be able to register according to the study plan because he/she has withdrawn some subjects. Thus, this situation will add more constraints to the registration process/planning. As a result, there are lots of rules, regulations and constraints that each student must consider in order to plan his/her subjects for each semester. Some computer engineering students at Prince of Songkla University also face the same difficulty. Even though Prince of Songkla University already provided the study result simulation tool at sis.psu.ac.th [1], the tool is not directly suggested a list of subjects to be taken in each semester. Furthermore, the Faculty of Engineering also provided a web-based application for students to check their study progress according to their respective curriculums [2]. However, the tool shows only the list of subjects in the curriculum and the grade of each subject in order for the students to see what subjects the student are still missing and what subjects the students are already passed. The task of planning the list of subjects to be taken in each semester is still relying on the students.

To solve such problem, this work aims to design and develop a tool that can provide a list of subjects to be taken each semester according to the rules and regulations of the curriculum and the university with an addition of the user preferences. The proposed tool will be based on the 2010 Computer Engineering Undergraduate curriculum of the Department of Computer Engineering at Prince of Songkla University. However, it can be easily adapted to support other curriculums. A Local search with a Tabu list is applied as the main searching technique. A solution that satisfies all the active constraints is returned for the user to make the final decision. However, this version of the software does not consider the out-of-plan subjects that are offered each semester. Tabu search and local search techniques had been successfully applied in many scheduling problems [3][4][5][6]. A combined Tabu search and local search technique had been applied to solve a high school timetabling problem [7]. The goal ensured that each teacher has a no overlapping time slot on his/her schedule. A Tabu search technique had been compared with a genetic algorithm and a backtracking approach on a personal scheduling problem in [8]. Pumpuang et al. [9] designed a model using Bayesian Network techniques to predict the order of subjects to be enrolled by undergraduate computer science or engineering students at

Kasetsart University. The data in their model was the student enrollment information consisting of the student's GPA and grade of each subject for first-year and second-year students with the objective of studying the student enrollment patterns.

The remaining of this paper consists of the methodology on how to design and develop the prototype tool, the results from the prototype on a few examples, the discussions on the preliminary evaluation by the potential users, and the conclusion.

Methodology

This work is a developmental research. The proposed system consists of three components, including the study plan, the search module and the database. Firstly, the study plan is a list of subjects to be taken in each semester with several constraints. The first set of constraints is on the relationships among subjects such as studied-prerequisite, passed-prerequisite, co-requisite and co-current subject. These relationships are commonly found in higher education curriculums. A typical four year undergraduate program of a semester system is divided into eight semesters. Figure 1 shows the relationship of subjects in the case study curriculum that is related to other subjects, while Figure 2 explains the meaning of each symbol in Figure 1. The subjects are listed in its assigned semester. The arrow between the shapes shows the category of relationships enforced on the subjects. The information in Figure 1 is used as an initial solution of our search module.

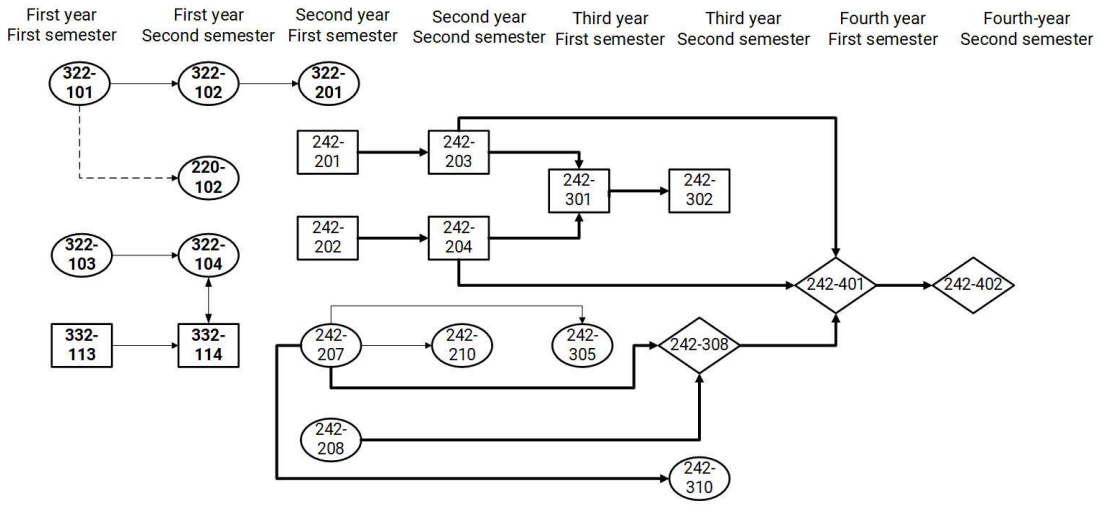

Figure 1 The relationship of subjects in the case study curriculum that is related to other subjects

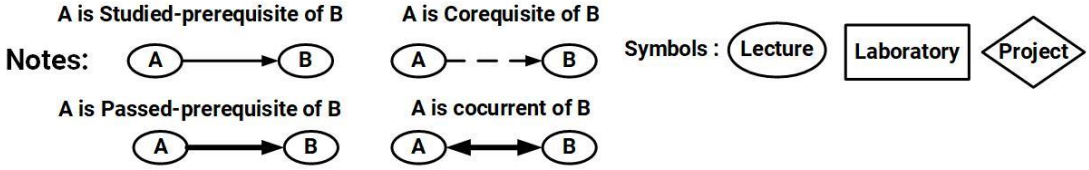

Figure 2 The symbol used in Figure 1

According to the relationship symbol presented in Figure 2, each relation can be explained as follow. A is a studied-prerequisite of B means that a student can enroll in subject B if any only if the student has enrolled and passed subject A with any grade including E. A is a passed-prerequisite of B means that a student can enroll in subject B if and only if the student has enrolled and passed subject A with any grade of at least D or S. A is a co-requisite of B means that a student must take subject B at the same time with subject A or a student has enrolled subject A with any grade of at least D or S before the student can take subject B. A is a co-current of B means that a student must take subject B at the same time with subject A when enrolls these two subjects for the first time. In addition to the relationship among subjects in the curriculum, the regulations according to the number of credits are also considered in the proposed prototype. The number of credits constraints can be caused by the student status and the university regulation. For example, Prince of Songkla University places a limit on the minimum and the maximum number of credits to be taken by a student in a regular semester as 9 and 22 credits and the limit changes to 0 and 9 credits during a summer semester. However, the student status can affect the maximum number of credits allowed. That is, the student with a non-regular status can take a maximum of 16 credits during a regular semester and 6 credits during a summer semester. Similar regulations are also commonly employed by most higher education curriculums. The last part is the scheduled subject of each semester restriction. It is common in any higher education curriculum that some subjects are offered in a specific semester only. Thus, the study plan must consider such restriction as well.

The second component is the search module. A combined local search [10] and Tabu list [9][10][11] is employed as the main searching technique in our proposed system. The main search procedure can be described as follow:

1. An initial solution is generated according to the standard study plan.
2. All conditions are evaluated on the current solution in order to determine the score. The conditions include the student status constraint, the number of credit constraint and the subject constraints according to the curriculum.
3. Add the current solution to the Tabu list. If the list is full, remove the oldest solution from the list.
4. Move to a neighbor solution with a better score.
    a. The next solution to be explored must not be the solution on the Tabu list.
5. Go to step 2 or stop when a stopping criteria is met and return the solutions on the Tabu list

At step 4 of the main search procedure, there are three steps to generate the neighbor solution. The first step is to move the whole subject chain if any subject is violated the condition. For an explanation purpose, Figure 3 shows the initial solution with an issue of a withdrawal subject (denoted by W). For simplicity, we use 6 credits as the limited maximum number of credits. Figure 4 shows the result of applying the moving-the-whole-chain method on the initial solution of Figure 3. According to the solution in this case, the Mecha subject must be moved to the 2nd semester of the 3rd year. Moreover, the number of credits is violated in the 2nd semester of the 2nd year. The second step is the moving some constraint-free subjects around after using the first solution. Figure

5 shows the result of applying both methods. As the results show that three constraint-free subjects are now moved around. The last step is to randomly move any two subjects without considering any constraint in order to throw the solution to a different area in the solution space, in order to prevent the local minima issue.

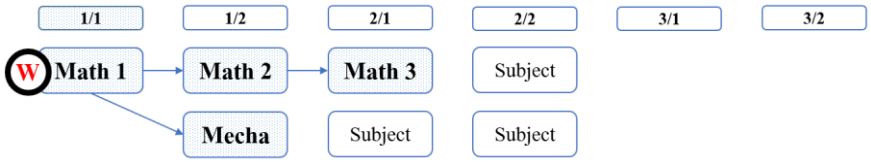

Figure 3 The initial solution with an issue of a withdrawal subject

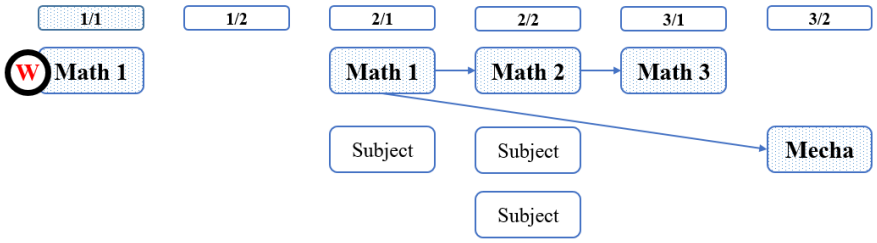

Figure 4 The results of moving the whole chain

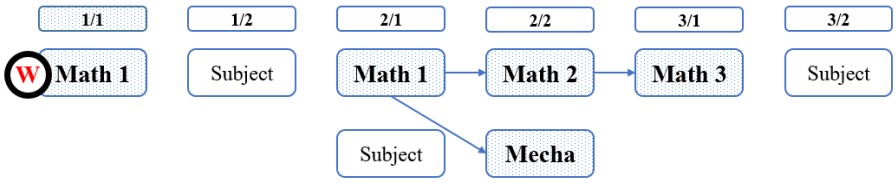

Figure 5 The results of applying the first two steps

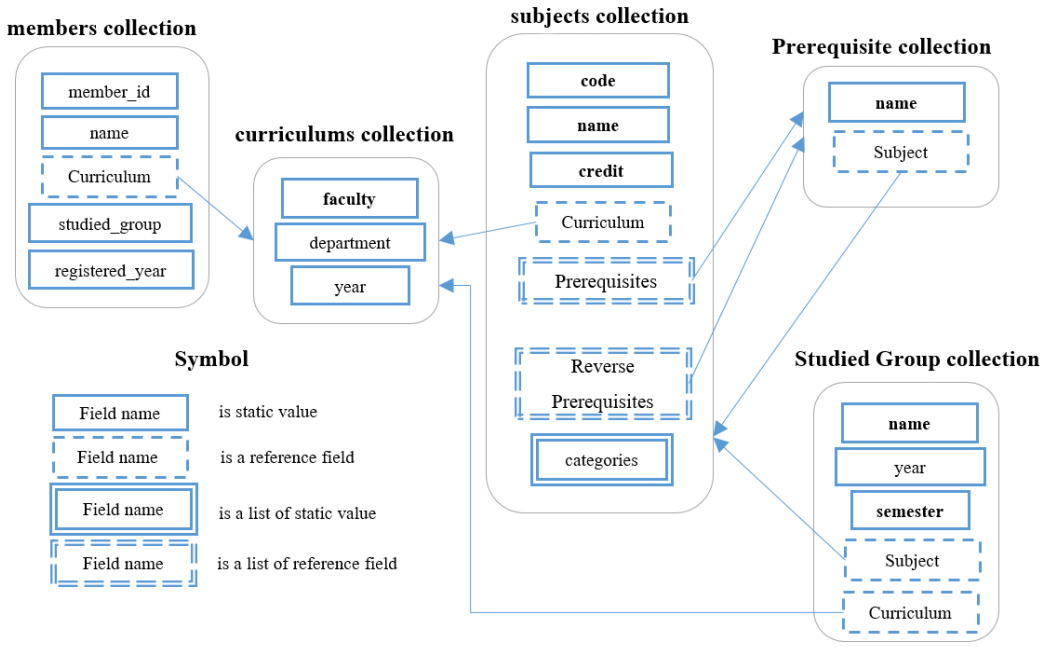

Figure 6 The MongoDB document schema

The stopping criteria at step 5 in this work is when there is no new neighbor solution found or the number of explored solutions exceeds a pre-defined value. The last component of the prototype is the databases. Figure 6 shows the MongoDB schema of the database used in this work. The database implementation is written in Python language in order to connect the MongoDB database. The Mongoengine [12] is used to bridge between the Python and the MongoDB.

Results

The overall architecture of the prototype is presented in Figure 7. The architecture of the prototype consists of two sides, including the client and the server sides. The client side is a web application, interacting with the user and displaying the study plans from the server. The server site contains the main components of the prototypes including the web service, the search engine and the database. When the user inputs his/her study results into the web application via the interface shown in Figure 8, the web application sends the HTTP request with the input in a JSON format to the web service API of the server side. Then, the web service executes the search engine to find a set of optimal study plans, and send the resulting study plans, which is also in a JSON format, to the web application. Lastly, the web application decodes the JSON and displays the resulting study plans as shown in Figure 9. The study plan can be shown in cozy view as shown in Figure 10. Moreover, the JSON results can also be seen as shown in Figure 11. The user interface of the web application is developed using Polymer [13] which applies MVC (Model-View-Controller) concept to control the web application. The App Router [14] module is also applied.

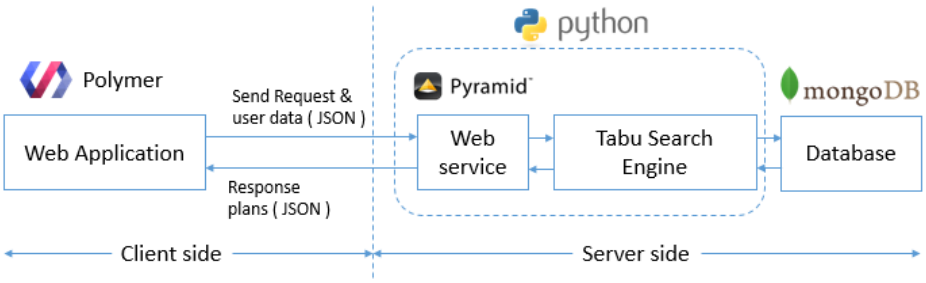

Figure 7 The architecture of the proposed prototype

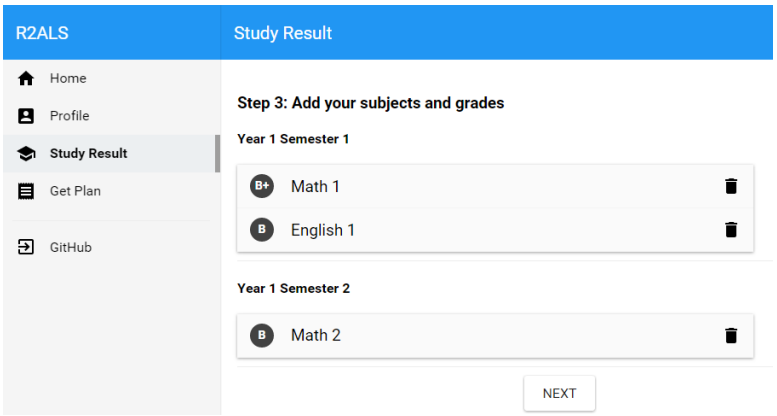

Figure 8 The study result input screen

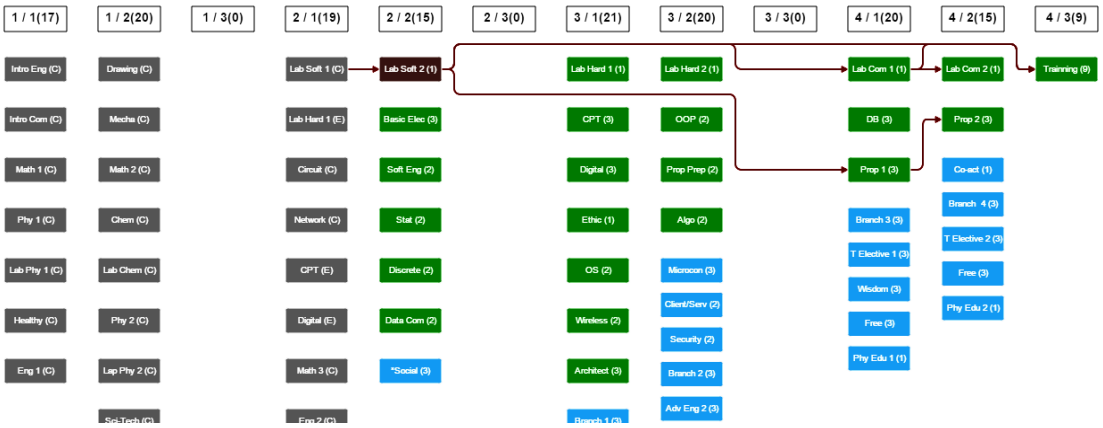

Figure 9 The result study plans

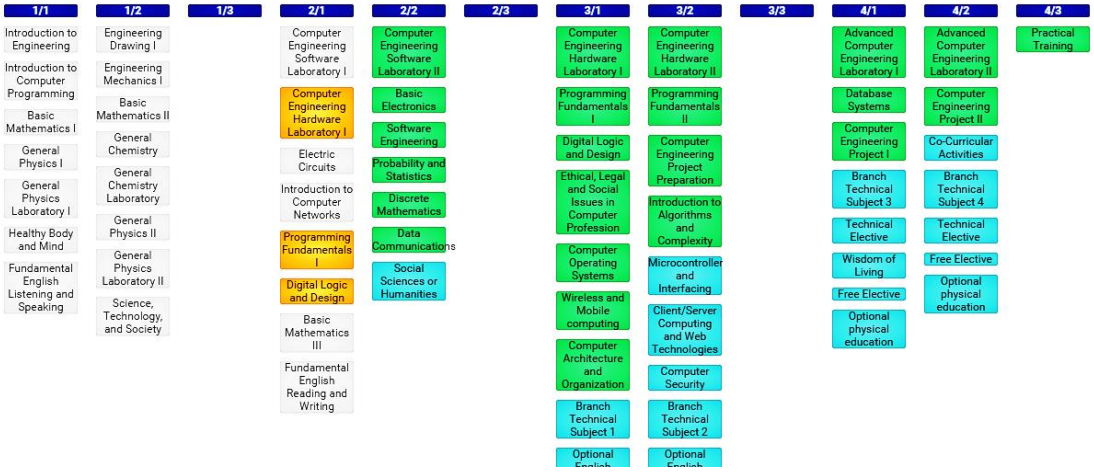

Figure 10 The cozy view of the same study plan shown in Figure 9

Input


```
    {
"member": {
    "name": "5410110070_1",
    "subject_group": "second-group"
},
"type": "array-of-semester",
"is_testing": true,
"semesters": [
    {
        "year": "1",
        "semester": "1",
        "subjects": [
            {
                "gradeOption": "credit",
                "name": "215-111 Engineering Drawing I",
                "id": "552e3b206e9552070c71a6d9",
                "year": "1",
                "semester": "1",
                "grade": "C"
            },
            {
                "gradeOption": "credit",
                "year": "1",
                "semester": "1",
                "name": "322-101 Basic Mathematics I",
                "id": "552e3b206e9552070c71a6cf",
                "grade": "D+"
            },
```


Output


```
    {
"last_year_plan": 4,
"total_credits": [
    17,
    20,
    0,
    19,
    15,
    0,
    21,
    20,
    0,
    20,
    15,
    9
],
"semesters": [
    {
        "semester": 1,
        "subjects": [
            {
                "semester": 1,
                "name": "Introduction to Engineering",
                "tags": [
                    "theory",
                    "choice"
                ],
                "id": "552e3b206e9552070c71a6cb",
```


Figure 11 The JSON view of the same study plan shown in Figure 9

Discussions

The preliminary evaluation was conducted on two phases. The initial results were collected from the project day preparation period by several volunteers. The initial results showed that the students without issues would focus on the user interface that showed the study plan. There were twenty study plans for each request. According to the initial results, we selected three specific subjects with study issues for a final evaluation by a method of interview. The final results were collected from three specific cases by a way of individual interview after the test subject had used the system. The results showed that the user interfaces of the prototype was friendly, clear and easy to use because it was similar style as that of the Google design. The prototype was able to show twenty different solutions for the students. However, the prototype should add the automatic download features so that the user did not have to enter their study results in order to reduce the time and the mistakes. In comparison with the existing systems including the study result simulation tool [1], and the Faculty of Engineering study progress check tool [2], the prototype was more satisfying. Firstly, the study result simulation tool was not directly suggested a list of subjects to be taken in each semester. Secondly, the study progress check tool showed only the list of subjects in the curriculum and the grades of each subject in order for the students to see what subjects the student were still missing and what subjects the students were already passed. The task of planning the list of subjects to be taken in each semester was still relying on the students.

Conclusions

A study planning tool for higher education was proposed in this work. A prototype of the tool was developed and evaluated. The prototype overall architecture consisted of two parts, including the client side and the server side. The client side was developed using polymer and the data exchange between the two parts was in a JASON format. The server side was the main engine of the system including the web service, the search engine and the database. The database implementation was written in Python language and the Mongoengine was used to bridge between the Python and the MongoDB. The search engine applied a local search with a Tabu list. The Bachelor of Engineering in Computer Engineering curriculum 2010, Prince of Songkla University was used as a case study. The prototype could follow the regulations that were commonly found on any higher education curriculum including the constraints on the subjects, the constraints on the minimum and maximum credits, and the student status. The general feedback from other students (without study issues) showed that the user interfaces of the prototype was friendly and clear and easy to use. The preliminary results by a method of individual interview three specific subjects with study issues showed that the prototype performed a better job than any existing tool currently provided by the university and the faculty. However, the prototype should directly connect to the university database in order to download all necessary information for the process in order to reduce the human mistakes during the data entry process.

References

[1] Prince of Songkla University (2018). **Student Information System** (Online). Retrieved on 28 Jan 2018, from http://sis.psu.ac.th.

[2] Faculty of Engineering, Prince of Songkla University (2018). **Study Result Tracking System**. (Online). Retrieved on 28 Jan 2018, from http://infor.eng.psu.ac.th/advisev2.

[3] Jiawei L. (2017). "Airport Intelligent Dispatching Model Based on Double Queue and Tabu Search", **International Symposium on Distributed Computing and Applications to Business, Engineering and Science**. 215-218. 13-16 October 2017. Anyang, China.

[4] Carcangiu S, Caboni M., Fanni A., Montisci A., Cardelli E., and Faba A. (2017). "Magnetic Materials Characterization by Tabu Search Optimization". **AEIT International Annual Conference**. 1-6. 20-22 September 2017. Cagliari, Italy.

[5] Kobubo T. and Fukuyama Y. (2017). "Generation Methods of Neighborhood Schedules for Practical Train Crew Scheduling Problems using Tabu Search". **IEEE International Workshop On Computational Intelligence and Applications**. 39-44. 11-12 November 2017. Hiroshima, Japan.

[6] Kawaguchi S. and Fukuyama Y. (2017). "Reactive Tabu Search for Job-shop Scheduling Problems Considering Energy Management". **IEEE International Workshop on Computational Intelligence and Applications**. 9-14. 11-12 November 2017. Hiroshima, Japan.

[7] Schaerf A. (1999). "Local Search Techniques for Large High School Timetabling Problems". **IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans**. 29(4). 368-377.

[8] Amol A.C. and Rajankumar B.S. (2012). "Tabu Search for Solving Personnel Scheduling Problem". **International Conference on Communication, Information & Computing Technology**. 1-6. 19-20 October 2012. Mumbai, India.

[9] Pumpuang P., Srivihok A., Praneetpolgrang P. and Numprasertchai S. (2008). "Using Bayesian Network for Planning Course Registration Model for Undergraduate Students". **IEEE International Conference on Digital Ecosystems and Technologies**. 492-496. 26-29 February 2008. Phitsanulok, Thailand.

[10] Gendreau M. (2003). "An Introduction to Tabu Search". In Glover F., Kochenberger G.A. (eds) **Handbook of Metaheuristics. International Series in Operations Research & Management Science**, 37-54. Boston, MA : Springer.

[11] Glover F. (1995). **Tabu Search Fundamentals and Uses**. (Online). Retrieved on 28 Jan 2018, from https://www.researchgate.net/publication/249776329_Tabu_Search_Fundamentals_and_Uses.

[12] Lawley R. (2018). **mongoengine**. (Online). Retrieved on 28 Jan 2018, from http://mongoengine.org.

[13] Polymer Authors. (2017). **Polymer**. (Online). Retrieved on 28 Jan 2018, from https://www.polymer-project.org.

[14] Ringsmuth E. (2018). **App-router**. (Online). Retrieved on 28 Jan 2018, from https://erikringsmuth.github.io/app-router/.