バス停時刻表検索システム用データの更新方法の検討

島根大学　総合理工学部　数理・情報システム学科

計算機科学講座　田中研究室

S113091 宮地 朱音

目次

1. 序論

1.1 研究背景

近年、Google Maps[1]等を用いた、視覚的にも便利な各種ルート案内サービスが普及してきている。
先行研究[2]による地図によるバス停時刻表検索システムもそのひとつである。このシステムは Google App Engine[3]を用いて高いスケールアウト性を確保した、島根県松江市のバス停・時刻表及び路線沿いの施設情報を検索し表示するシステムである。
このシステムは検索のために時刻表等のデータをあらかじめ独自の形式でまとめたものを使用しており、内容の更新は手作業で行う必要があるため、ダイヤの改正に伴う作業量は膨大なものとなる。
そこで、このシステムの検索用データをダイヤ改正時に更新する際、その手間を減らすための方法を検討することにした。

1.2 研究概要

先行研究の検索システムに対応した独自形式の時刻表データを、バス会社が公表している PDF ファイルをもとにして CSV ファイルにまとめ、生成するプログラムを作成した。
また、生成した CSV ファイルを Google App Engine[3]上にアップロードするためのプログラムも作成した。
これらのプログラムの作成においては、Eclipse 4.3(Kepler)[4]を使用した。データアップロードの際には Google Plugin for Eclipse[5]も利用し、ウェブアプリケーションのプロジェクト作成、ローカルサーバでの実行等を行った。

1.3 先行研究

先行研究[2]の松江市バス路線時刻表検索システムは Web ページ上で実行される。出発地と目的地をそれぞれ地図上でクリックし、日時を指定すると、クリックしたそれぞれの地点から半径 500m 以内のバス停を探し、それら 2 つのバス停を通るバス路線とその時刻表を検索し、バスの通るルートと共に表示するものとなっている。さらに、沿線情報も検索できるようになっている。

図 1.1 先行研究での実行例

2. 時刻表 CSV ファイル生成

2.1 CSV ファイル生成の流れ

まず対象とする PDF ファイルを開いて読み込み、全体をテキストとして抽出する。次に抽出したテキストから時刻表にあたる部分のみを切り抜き、その部分を一行ずつ整理して CSV に書き込んでいく。すべての行について出力できたらプログラムを終了する。

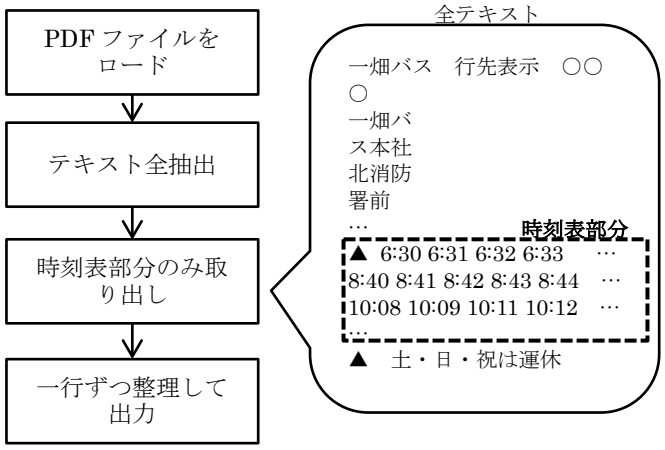

図 2.1 CSV ファイルを生成する流れ

2.2 扱うデータ

今回は、検索システムで必要なデータのひとつであるバス編成テーブルについて生成する。バス編成テーブルは各バス停の時刻表を格納する表である。出発バス停、目的バス停を通るバス路線を検索した後、このテーブルで時刻表を検索する。このテーブルは路線 ID、時刻、バス ID、備考の列で構成される。路線 ID は路線ごとにユニークに割り振った ID である。また、バス ID とは、同じ路線を通るバスを区別するために、その日のその路線の何本目のバスかというのを決定するためのものである。時刻については通過するバス停順に格納されている。このデータは、一畑バス[6]が提供する松江管内のバス時刻表 PDF ファイルを対象に取得した。

表 2.1 バス編成テーブル

路線 ID (STRING)	平日 (STRING)	休日 (STRING)	備考 (STRING)	バス ID (INT)	時刻 1 (INT)	時刻 2 (INT)	…	時刻 64 (INT)	時刻 (STRING)
10020101	1	0	休日運休	1	730	732	…	0	7:30/7:32/…/8:10
⋮	⋮	⋮	⋮	⋮	⋮	⋮	…	⋮	⋮
10460101	1	0	休日運休	1	1020	1021	…	0	10:20/10:21/…/11:01
10460101	1	0	休日運休	2	1120	1121	…	0	11:20/11:21/…/12:01
10460101	1	1		3	1220	1221	…	0	12:20/12:21/…/13:01
⋮	⋮	⋮	⋮	⋮	⋮	⋮	…	⋮	⋮

2.3 プログラム作成のための開発環境

プログラム作成にあたり開発環境を整えた。1.2 研究概要でも述べた通り Eclipse 4.3(Kepler)[4]を用い、開発言語は Java とした。PDF ファイルを扱うにあたり Java ライブラリである Apache PDFBox[7]を使用した。

2.4 時刻表 CSV ファイル生成プログラム

まず対象となる PDF ファイル名を指定して開く。続いて、開いたファイルの全文をテキスト抽出し、その中から時刻表部分のみを取り出す。以下にプログラムを示す。

```
35  >  >  >  /* PDFドキュメントをロード */
36  >  >  >  PDDocument doc = PDDocument.load("time_" + rosen +".pdf");
37  >  >  >  PrintWriter outFile = new PrintWriter
38  >  >  >  >  >  (new BufferedWriter(new OutputStreamWriter(new FileOutputStream("time_" + rosen + ".csv"),"MS932")));
39  >  >  >  PDFTextStripper stripper = new PDFTextStripper();
40  >  >  >  String pdfData = stripper.getText(doc).trim();
41
42  >  >  >  /* PDFデータから読み込んだテキストの時刻表テーブル部分のみを抜き出す */
43  >  >  >  int j = pdfData.indexOf(":"); //左側からの検索
44  >  >  >  String tmp = pdfData.substring(0,j); //全文の最初から":"出現直前まで
45  >  >  >  int t_start = tmp.lastIndexOf("¥n"); //最後に出現する改行文字の位置
46  >  >  >  String pdfDataTmp = pdfData.substring(t_start);
47
48  >  >  >  int k = pdfDataTmp.lastIndexOf(":"); //右側からの検索
49  >  >  >  tmp = pdfDataTmp.substring(k+1);
50
51  >  >  >  int t_end = tmp.indexOf("¥n");
52  >  >  >  String timeTable = pdfDataTmp.substring(0,k+t_end);//timeTable:時刻表部分だけ取り出した文字列
53  >  >  >  timeTable = timeTable.trim(); //最初と最後の改行文字を削除
```

図 2.2 PDF ファイルから時刻表部分をテキスト抽出するための記述

時刻表部分は、時刻を記述する際「6:30」のように":"(コロン)を用いているため、コロンが含まれる最初の行から最後の行までを取り出すように指定している。

次に取り出した時刻表部分を 1 行ずつ整理し、バス路線 ID やバス ID、平日運行・休日運行といった情報も付加して CSV ファイルに書き込んでいく。このとき、”▲”（土日祝運休）の印が行頭にあるかないかで処理を分岐するようにしている。以下にプログラムを示す。

```
            int loop = countStringInString(timeTable,"¥n") + 1; //時刻表部分の行数カウント

            for(int i = 1 ; i < loop ; i++){

                int tab = timeTable.indexOf("¥n"); //時刻表で初めて改行が登場する位置(＝一行目終端)

                if((timeTable.substring(0,tab)).indexOf("▲") == -1){ //一行目に▲(土日祝運休)がなければ
                    ttline = timeTable.substring(0,tab+1); //一行目
                    ttlineStr = (ttline.replaceAll(" ","/")).trim();
                    ttline = (ttline.replaceAll(":", "")).trim();
                    outFile.write(bus_id + ",1,1,," + "一畑バス," + i + ",");

                    ttSplit = ttline.split("[¥¥s]+"); //半角スペースごとに区切る

                    int a;
                    for (a = 0;a < ttSplit.length ; a++){
                        outFile.write(ttSplit[a] + ",");
                    }

                    for(int b = 63-a;b>0;b--){
                        outFile.write(",");
                    }
                    outFile.write(ttlineStr + "¥n");

                } else{ //一行目に▲があれば
                    ttline = timeTable.substring(2,tab+1); //一行目から▲を抜いたもの
                    ttlineStr = (ttline.replaceAll(" ","/")).trim();
                    ttline = (ttline.replaceAll(":", "")).trim();
                    outFile.write(bus_id + ",1,0,休日運休," + "一畑バス," + i + ",");
                    ttSplit = ttline.split("[¥¥s]+");

                    int a;
                    for (a = 0;a < ttSplit.length ; a++){
                        outFile.write(ttSplit[a] + ",");
                    }

                    for(int b = 63-a;b>0;b--){
                        outFile.write(",");
                    }
                    outFile.write(ttlineStr + "¥n");
                }
                timeTable = timeTable.substring(tab + 1); //CSVに書き込み済みの一行を削除する
            }
```

図 2.3 時刻表データを CSV ファイルにまとめ出力するための記述

3.Google App Engine へのアップロード

3.1 Google App Engine とは

Google App Engine[3]とは、Google のインフラ上で、開発した Web アプリケーションを実行できる Web アプリホスティングサービスである。データストアは Key-Value Store である Bigtable で、負荷分散を自動で行うため、アクセスが集中してもパフォーマンスが悪化しないという特長を持つ。
また、ある程度までの使用は無料でできる点や、サーバ構築の必要がない点から、アプリケーションの公開が容易に行えるという利便性もある。

3.2 アップロードに際した開発環境

Google App Engine 上へのデータアップロードに際し、Google App Engine と Eclipse の連携を取る必要があったため、Google App Engine SDK for Java[8]のパッケージの「appengine-java-sdk-1.9.5.zip」をダウンロードし、インストールした。そして Eclipse の機能を用いて Google Plugin for Eclipse[5]をインストールした。さらに、CSV ファイルアップロードのために
「commons-fileupload-1.3.1.jar」を追加した。
また、データの格納には Google App Engine でデータストアの API 標準としてサポートされている Java Data Objects を利用した。

3.3 検索用データのアップロード

Google App Engine の管理画面には Datastore Viewer があり、データストアにあるデータの確認やアップロード等はそこから行えるが、用意したデータをひとつひとつアップロードするには膨大な手間がかかる。そこで、CSV ファイルにまとめたデータをアップロードするプログラムを、先行研究に従って作成した。

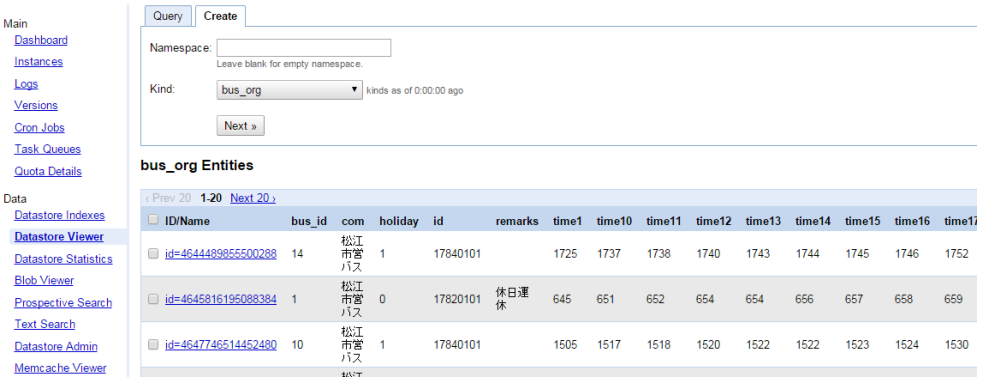

図 3.1 Google App Engine の管理画面の一部

3.3.1 エンティティの定義

Google App Engine では Java Data Objects をサポートしているため、これを用いてオブジェクトの永続化を行う。

```
11  @PersistenceCapable(identityType = IdentityType.APPLICATION)
12  public class bus_org {
13
14      @PrimaryKey
15      @Persistent(valueStrategy = IdGeneratorStrategy.IDENTITY)
16      private Key key;
17
18      @Persistent
19      private String id; //バス路線ID
20
21      @Persistent
22      private String weekday; //平日運行かどうか(0/1)
23
24      @Persistent
25      private String holiday; //休日運行かどうか(0/1)
26
27      @Persistent
28      private String remarks; //備考
29
30      @Persistent
31      private String com; //バス会社名
32
33      @Persistent
34      private int bus_id; //バスID
35
36      @Persistent
37      private int time1; //時刻1
38
39      @Persistent
40      private int time2; //時刻2
41
42      @Persistent
43      private int time3;
44
45      @Persistent
46      private int time4;
```

図 3.2 エンティティの定義

@PersistenceCapable で永続化対象のクラスであることを宣言し、@Persistent で永続化対象のフィールドであることを宣言、@PrimaryKey でエンティティの主キーを定義している。

3.3.2 データの追加

データの追加を行うプログラムを以下に示す。

```
            FileItemIterator itemIterator = fileUpload.getItemIterator(req);
               while (itemIterator.hasNext()) {
               FileItemStream itemStream = itemIterator.next();
               InputStream inputStream = itemStream.openStream();
               String contentType = itemStream.getContentType();
               if (contentType == null) {
                    contentType = "";
               }
               if (contentType.contains("text")  || itemStream.getName().endsWith(".csv")) {
                    resp.setContentType("text/html; charset=UTF-8");
                    BufferedReader buffer = new BufferedReader(new InputStreamReader(inputStream, "UTF-8"));
                    String line = null;
                    int i = 0;
                    while ((line = buffer.readLine()) != null) {
                        String[] split = line.split(",",-1);

                        // csvからデータを取り出す
                            String id = split[0].trim();
                            String weekday = split[1].trim();
                            String holiday = split[2].trim();
                            String remarks = split[3].trim();
                            String com = split[4].trim();
                            int bus_id = Integer.parseInt(split[5].trim());
                            int time1 = Integer.parseInt(split[6].trim());
                            int time2 = Integer.parseInt(split[7].trim());
                            int time3 = Integer.parseInt(split[8].trim());
                            int time4 = Integer.parseInt(split[9].trim());

                                                ⋮

                            int time63 = Integer.parseInt(split[68].trim());
                            int time64 = Integer.parseInt(split[69].trim());
                            String time_table = split[70].trim();


                            bus_org per = new bus_org(id, weekday, holiday, remarks, com, bus_id,
                                time1, time2, time3, time4, time5, time6, time7, time8, time9, time10,
                                time11, time12, time13, time14, time15, time16, time17, time18, time19, time20,
                                time21, time22, time23, time24, time25, time26, time27, time28, time29, time30,
                                time31, time32, time33, time34, time35, time36, time37, time38, time39, time40,
                                time41, time42, time43, time44, time45, time46, time47, time48, time49, time50,
                                time51, time52, time53, time54, time55, time56, time57, time58, time59, time60,
                                time61, time62, time63, time64, time_table);

                            PersistenceManager pm=pmf.getPersistenceManager();

                            if (pm != null) {
                                try{
                                 pm.makePersistent(per);
                                }finally{
                                 pm.close();
                                }
                            }
```

図 3.3 データの追加を行うプログラム

String 型配列「split」に CSV ファイルのデータを一行ずつ取得し、trim メソッドでその各パラメータを取り出し、インスタンス化してデータストアに格納している。

3.3.3 アップロード作業

3.3.2 節のプログラムをサーブレットとし、簡単な HTML ファイルを作成した。アップロードはこの HTML のページ上で実行する。

```
<!DOCTYPE html>

<html>
    <head>
      <meta http-equiv="content-type" content="text/html; charset=UTF-8">
      <title>File Upload</title>
    </head>
    <body>
      <h1>File Upload(バス編成)</h1>
      <form action="/fileuploadbusorg" method="post" enctype="multipart/form-data">
          <input type="file" name="file" /><br />
          <input type="submit" value="アップロード" />
      </form>
    </body>
</html>
```

図 3.4 アップロード作業を行うための HTML ページ

図 3.5 アップロード作業画面

フォームで CSV ファイル名を指定し、アップロードボタンを押すとアップロードを開始する。

4.結論

4.1 実行結果

一畑バスの各時刻表 PDF ファイルについて、それぞれ CSV ファイル生成が出力できた。

20000000	1	0	休日運休	一畑バス	1	630	631	632
20000000	1	0	休日運休	一畑バス	2	640	641	642
20000000	1	0	休日運休	一畑バス	3	650	651	652
20000000	1	0	休日運休	一畑バス	4	710	711	712
20000000	1	1		一畑バス	5	740	741	742
20000000	1	0	休日運休	一畑バス	6	800	801	802
20000000	1	1		一畑バス	7	840	841	842
20000000	1	1		一畑バス	8	940	941	942
20000000	1	1		一畑バス	9	1040	1041	1042
20000000	1	1		一畑バス	10	1140	1141	1142
20000000	1	1		一畑バス	11	1240	1241	1242
20000000	1	1		一畑バス	12	1340	1341	1342
20000000	1	1		一畑バス	13	1440	1441	1442
20000000	1	1		一畑バス	14	1540	1541	1542
20000000	1	0	休日運休	一畑バス	15	1605	1606	1607

図 4.1 出力された CSV データの一部

また、Google App Engine へのアップロードが成功した。
実行に用いた HTML ページのアドレスは
http://1-dot-s113091-su.appspot.com/fileuploadbusorg.html
である。

4.2 今後の課題

PDF ファイルの情報をテキストとして抜き出す際、下図のように空白テーブルがあるとその部分が抜け落ちるようになってしまう。このような抜け落ちのないプログラムに改良するのが今後の課題である。
また、今回は一畑バスのデータを対象に時刻表データ生成プログラムを作成したため、他形式の PDF ファイルには対応していない。よって、他社の時刻表 PDF ファイルの形式にも対応するようなシステムの検討も課題として挙げられる。

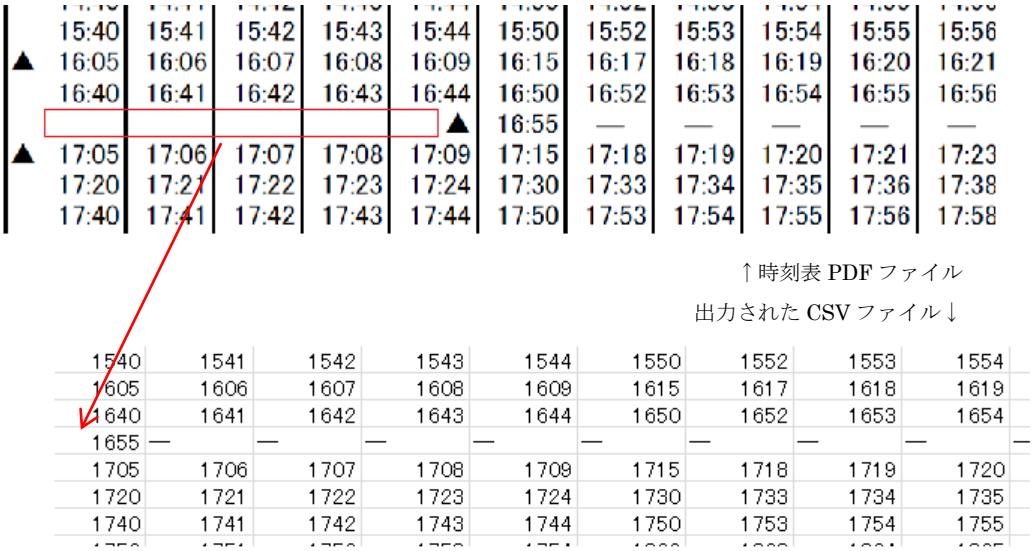

図 4.2 テーブルの空白部分の欠落

謝辞

本研究を進めるにあたり、最後まで熱心なご指導をして頂きました田中章司郎教授には心より御礼申し上げます。
また、数々のご協力とご助言をして頂きました同研究室の皆様にも、厚く御礼申し上げます。
なお、本論文、本研究で作成したプログラム及びデータ、並びに関連する発表資料等の全ての知的財産権を本研究の指導教官である田中章司郎教授に譲渡致します。

参考文献

[1] http://maps.google.co.jp/

[2]三島健太,「JTS を用いた沿線検索システムの Google App Engine 上での実装」,島根大学 総合理工学研究科 数理・情報システム学専攻 修士論文,2012

[3] https://cloud.google.com/appengine/docs

[4] http://www.eclipse.org/

[5] https://cloud.google.com/appengine/docs/java/tools/eclipse

[6] http://www.ichibata.co.jp/bus/

[7] http://pdfbox.apache.org/

[8] https://cloud.google.com/appengine/downloads