TicketAdvisor：铁路中转换乘方案推荐专家系统

马逸君 17300180070

摘要：本文介绍 TicketAdvisor，一款根据用户出行需求推荐铁路中转换乘方案的专家系统。本文首先分析两项相关工作——基于框架的专家系统的理论和市面上已有的铁路中转换乘方案推荐软件，然后介绍本专家系统的知识表示、推理引擎、解释机制、用户界面，最后用三个典型案例(深圳-重庆、上海-青岛、武汉-济南)说明本专家系统的应用效果。

关键词：专家系统；知识表征

1 简介

铁路是中长途出行的主流交通方式，中转换乘是铁路出行中的一个重要话题。笔者在生活中常常帮同学、朋友制订铁路出行方案，其中不乏中转换乘方案。现在，我们希望设计一个计算机程序，其可以自动完成为用户推荐中转换乘方案的目标，该如何入手呢？因为笔者对铁路有丰富的了解，所以我们可以采用专家系统，将铁路知识显式地表征出来，利用它在全国铁路所有车次数据库中自动进行推理。

我们介绍 TicketAdvisor，一款根据用户出行需求推荐铁路中转换乘方案的专家系统。本专家系统是基于框架的专家系统，关于铁路列车的知识表达在列车时刻表这一框架中；我们还根据实际应用场景，增加了一些规则来与框架交互，例如本专家系统只在预先定义好的全国铁路枢纽站(主要车站)列表中选取中转站，该主要车站列表也是专家知识的体现。

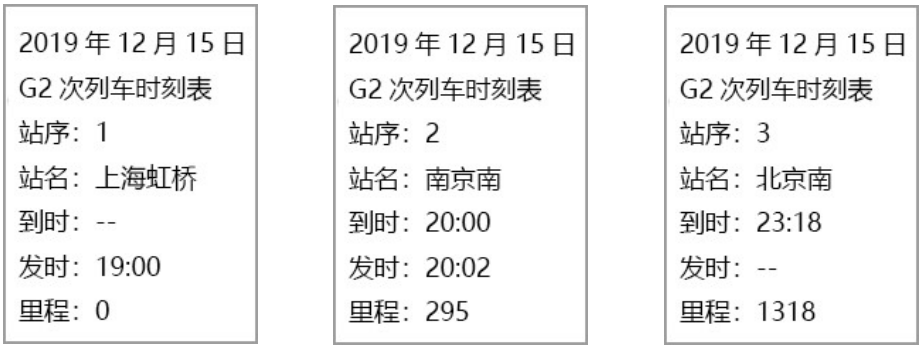
本专家系统提供了图形界面，用户在主界面（如图 5 所示）上完成设置后，点击“查询”按钮，推理引擎(作为“查询”按钮的守护程序)即进行模式匹配，在数据库中查找所有符合用户设定条件的车次实例，然后将推荐中转城市的信息显示给用户。本专家系统还提供解释机制，推理引擎完成一次计算后，用户可点击主界面右侧的“查看所有方案”按钮，查看推理引擎的解释；点击该按钮将显示推理引擎尝试计算过的所有中转城市，作为对推荐方案的解释。

作为测试，我们对三条典型线路，分别是深圳-重庆、上海-青岛、武汉-济南，用本专家系统进行了推理，本专家系统都得到了比直达更优的中转换乘方案。

2 Related Work

专家系统[1]。在人工智能的发展史中，人们认识到，要用机器解决智能问题，必须让其具备特定领域的知识（表现在事实和规则两方面），这些知识往往是该领域的人类专家所具有的。人类专家提供知识后，我们就可以把知识输入计算机，期望计算机能够在特定领域像专家一样解决问题，也就是构建专家系统——在狭窄问题领域具有专家水平的计算机程序。这里的“狭窄”如何界定？学者云：给领域专家打三五个电话，就能把问题解决，这就符合狭窄专业领域的标准。

在专家系统中，**知识有显式的表征。**在基于框架的专家系统中，知识用框架的形式表达。框架是名称和相关属性(槽)的集合，它是带有关于某个对象或概念的典型知识的数据结构，为知识的结构化表达提供了一种自然的方法。在 TicketAdvisor 中，列车时刻表即为表达关于铁路列车知识的框架；其结构如图 1 所示。



a) b) c)

图 1　列车时刻表框架

基于框架的专家系统也用到规则来评估框架中的信息；不同于基于规则的专家系统，在基于框架的专家系统中，规则通常使用模式匹配子句，用于在所有实例框架中找到匹配条件的变量，规则只是辅助角色，框架是知识的主要来源。

基于框架的专家系统可以实现很多高价值应用，如默认推理[2]。

铁路中转换乘方案推荐软件。鉴于我国不充分不平衡的发展现状，地区和地区之间在经济状况、工资及生活水平、科教文卫资源等方面差异甚大，人们前往中心城市务工、求学、发展，导致大规模的人口流动。铁路（包括普速铁路和高速铁路）作为价格和舒适度折中的一种客运方式，是中长途客运的主流选择。

从宏观层面上看，铁路建设是循序渐进的；受硬件条件限制，铁路客运服务难以满足所有的交通需求。我们经常发现这样的情况：主要城市之间，往往有大站快车运行；从普通地级市或县、乡级车站到主要城市，很少有大站快车运行，往往车次较少，且这些车次一般停站较多、耗时较长；从一个地级市到外省的另一个地级市，则很可能没有直达车次。没有直达车次时，到邻近大站中转换乘是一种可行的方案；而对于没有大站快车运行的线路，旅客也可选择到邻近大站换乘快车，总时间可能比直达车次更短。

面对旅客中转换乘的需求，市场上有很多软件提供中转换乘查询。例如，铁路部门在官方网站 12306.cn 和官方 App“铁路 12306”提供了接续换乘查询，如图 2 所示；一些列车时刻表查询 App 如“盛名时刻表”也提供中转查询功能。这些软件检索当前有票的车次，从中为旅客推荐总耗时最短的中转换乘方案。

然而，既有软件提供的接续换乘查询仍有改进空间：运行机制并不透明，自由度较低。例如，旅客不能在高铁动车和普速列车中自由选择；旅客只能按耗时和出发时间、而不能按价格对中转换乘方案排序；旅客无法按个人意愿指定换乘车站。有鉴于此，我们希望能够开发一款自己的中转换乘推荐软件，在以上方面做出改进。

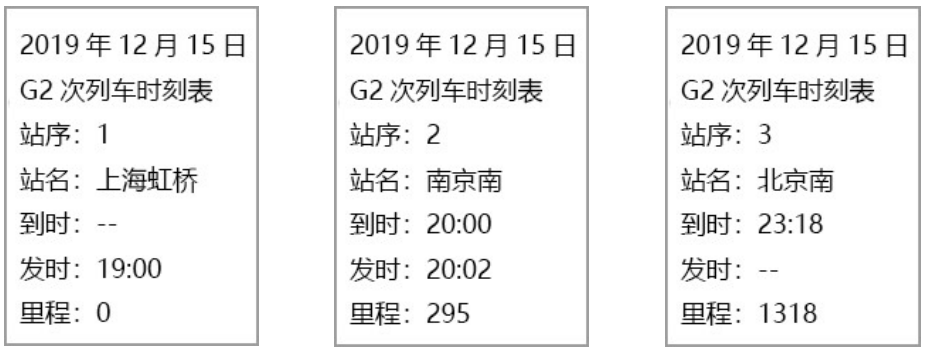
图 2 “铁路 12306”的接续换乘查询

3 实现思路

这一节中，我们首先讨论如何将专家知识在计算机中显式地进行表征，然后讨论利用专家知识如何进行推理计算得到推荐方案，然后是本专家系统对推荐方案的解释和用户界面。

3.1 知识表示

框架化的知识表达。在本专家系统中，关于铁路列车的知识表达在列车时刻表中。鉴于每趟列车的停站数目不同，为保证框架结构的统一，我们不直接存储一趟列车的时刻表，而是将一趟列车的每个经停站都存为一份实例；亦即我们的框架中存储的不是列车时刻表，而是列车时刻表表项，其结构如图 1 所示。



a) b) c)

图 1　本专家系统使用的框架

为了便于维护，便于在需要时添加及删除车次，我们需要一种易于修改并可以从专家系统访问的数据库。得益于该框架结构的统一性，我们可以将车次时刻表表项轻松地存入 MySQL 数据库内，如图 3。我们收集全国铁路所有车次的时刻表信息[3][4]，并整理成这种形式，就得到了我们的知识库。

	id	sn	sta...	arr...	st...	dur...	dis...	train_no
1	80590	1	上海虹桥	09:00	09:00	00:00	-	G2
2	80591	2	南京南	10:07	10:09	01:07	295	G2
3	80592	3	北京南	13:49	13:49	04:49	1318	G2

图 3 MySQL 数据库 train_no_detail 的属性

规则与框架的交互。上面已经定义了知识的框架化表达，我们还可根据实际应用场景，增加一些规则来与框架交互，增强我们的知识库。

理论上，利用收集到的全国铁路全车次时刻表，我们已经足以设计出一个中转换乘方案推荐系统：在进行查询时循环尝试全国所有车站作为中转站，查找所有合法的中转换乘方案。然而这样查询耗时巨大。考虑到我们的设计目的是找出高效、可行的中转换乘方案，而实际操作中存在以下一些限制：一般只有铁路枢纽站（铁路枢纽就是不同铁路线的交汇点）的换乘设施完善，旅客可以快速地、顺利地完成换乘；一般只有铁路枢纽站拥有大站快车，可以

提供高效的中转换乘方案；铁路枢纽站经过的车次较多，只需在全国铁路枢纽站列表中检索就可以覆盖绝大部分的可能方案。考虑到这些影响因素，本专家系统中还作这一限制：推理引擎只在预先定义好的全国铁路枢纽站(主要车站)列表中选取中转站。

而预先定义这一主要车站列表的过程也用到了本人对铁路线路的知识（专家知识）。举个例子，京沪线、京广线、焦柳线、陇海线、沪汉蓉线都是中国重要的铁路干线，京沪线和陇海线的交汇点是徐州，京广线和陇海线的交汇点是郑州，京广线和沪汉蓉线的交汇点是武汉（更严格地说是汉口站），焦柳线和沪汉蓉线的交汇点是宜昌，故这些城市的车站都被列入主要车站列表。利用这样的方法，我们人工定义了 116 个枢纽城市的共计 234 个车站为主要车站，如图 4 所示。

main.py | sql.py | knowledge.py

```
69  major_station_list = {
70      # Wuhan Railway Bureau
71      '武汉': ('武昌', '汉口', '武汉'),
72      '宜昌': ('宜昌', '宜昌东'),
73      '襄阳': ('襄阳', '襄阳东'),  # '襄州'
74      '麻城': ('麻城', '麻城北'),
75      '信阳': ('信阳', '信阳东'),
76
77      # Beijing
78      '北京': ('北京', '北京西', '北京南', '北京东', '北京北', '昌平北', '黄村'),
79      '天津': ('天津', '天津西', '天津南', '天津北', '滨海'),
80      '石家庄': ('石家庄', '石家庄北', '石家庄东'),
81      '唐山': ('唐山', '唐山北', '曹妃甸东', '曹妃甸港'),
82      '秦皇岛': ('秦皇岛',),
83      '山海关': ('山海关',),
84      '衡水': ('衡水', '衡水北'),
85      '邯郸': ('邯郸', '邯郸东'),
86      '德州': ('德州', '德州东'),
87
88      # Shanghai
89      '上海': ('上海', '上海南', '上海虹桥', '上海西', '安亭北'),
```

图 4 利用专家知识人工定义的
全国铁路主要车站列表

该列表将作为本专家系统推理过程中的陪域，推理引擎将只在该列表中选取可能的解。再结合人类专家的一些常识——例如，从出发站到中转城市的一个车次和从中转城市到到达站的一个车次，合起来就是一个合法的中转换乘方案——就可以写出如下的模式匹配语句（此处只展示伪代码，相关的细节将在“推理引擎”一节中讨论）。该语句就是本专家系统最重要的一条规则。

```
FOREACH Transit City IN Major city list:
// 遍历主要车站列表中所有城市
    Train list 1 OF Transit City := {Train
        FOR Train IN MySQL train_no_detail
        WHERE Train: Start Station -> Transit City} //模式匹配
    // 出发站到中转地的所有车次
    Train list 2 OF Transit City := {Train
        FOR Train IN MySQL train_no_detail
        WHERE Train: Transit City -> Dest Station} //模式匹配
    // 中转地到到达站的所有车次
```

此外，因为我们在主界面上只显示最优的一个中转地，所以在推理引擎中还需加入这样一条规则：以代价为比较依据，选出最优的中转地。表达成显式的 IF-THEN 语句就是：

```
Recommended transit city := None //初始化
...... //推理过程，计算下文中 City A 作为中转城市的总花费代价
IF Recommended transit city IS None
OR Cost OF City A < Cost OF Recommended transit city
THEN Recommended transit city := City A //比较代价
```

3.2 推理引擎

因为本系统实现了图形化用户界面，因此推理引擎位于相关的守护程序中。用户在主界面（如图 5 所示）上完成设置后，点击“查询”按钮，推理引擎(守护程序)即进行**模式匹配**，在数据库中查找所有符合用户设定条件的车次实例。

图 5 主界面

定义在“查询”按钮上的守护程序(推理引擎)的行为是，一旦被触发，首先从用户界面(UI)获得用户的输入值；然后循环遍历预定义的主要车站列表，对于其中每个城市（若用户有强制中转设定，则只计算强制中转的城市），在数据库中检索符合用户设置的中转方案并计算代价；取总代价最小的城市为推荐中转城市；最后将推荐中转城市的信息显示给用户。该守护程序的伪代码如下：

```
DEMON 1
IF selected OF 查询 pushbutton
THEN BEGIN
    // 从用户界面(UI)取得用户输入值
    Start station := Text OF 出发站 textbox OF Ui
    Dest station := Text OF 到达站 textbox OF Ui
    Preferred train := Value OF 偏好车型 combo box OF Ui
    Set transit station := Text OF 强制中转 textbox OF Ui
    Weight 1 := (Value OF 里程权重 number box OF Ui) / 100
    Weight 2 := (Value OF 时间权重 number box OF Ui) / 100
```

```
    // 里程和时间在“代价”中的占比
    // 用户输入的值是实际系数的 100 倍（因为输入框的后面有个百分号），故除以 100。

    // 循环检索预定义的主要车站列表，找出代价最小的中转城市
    Recommended transit city := None
    FOREACH Transit City IN (Major city list IF Set transit station IS None
      ELSE Set transit station)
      // 遍历主要车站列表中所有城市，若用户设定了强制中转，则只检索强制中转的城市

        Train list 1 OF Transit City := {Train
            FOR Train IN MySQL train_no_detail
            WHERE Train: Start Station -> Transit City}
        // Train list 1 是从出发站到中转地所有车次的集合
        Train list 2 OF Transit City := {Train
            FOR Train IN MySQL train_no_detail
            WHERE Train: Transit City -> Dest Station}
        // Train list 2 是从中转地到到达站所有车次的集合

        IF Train list 1 IS Empty OR Train list 2 IS Empty
        THEN CONTINUE
        // 如果 Train list 1 和 Train list 2 中至少一个为空，则不继续执行下面的语句

        Cost 1 := min {Weight 1 * Distance + Weight 2 * Time
            OF Train FOR Train in Train list 1}
        Cost 2 := min {Weight 1 * Distance + Weight 2 * Time
            OF Train FOR Train in Train list 2}
        // 使用用户输入的 Weight 1 和 Weight 2 计算车次的“代价”
        Cost OF Transit City := Cost 1 + Cost 2
        // 取所有的前后车次组合中代价最小的一对车次的代价，作为当前枚举的中转城市的代价

        IF Recommended transit city IS None
        OR Cost OF Transit city < Cost OF Recommended transit city
        THEN Recommended transit city := Transit city
        // 每轮循环的最后，比较当前检索的城市与 Recommended transit city 的代价
        // 所有循环结束后，Recommended transit city 就是代价最小的中转城市

    // 将推荐中转城市的信息显示给用户
    Text OF 推荐中转地 textbox OF Ui := Recommended transit city
    Content OF 前车车次 table OF Ui := Train list 1
    Content OF 后车车次 table OF Ui := Train list 2
    // 还会显示中转总时间、总里程，为简单起见略去
END
```

其中值得讨论的是“代价”的计算方式。用户在用户界面上输入的 Weight 1 作为里程的权重值，Weight

2 作为时间的权重值，其中 Weight 1, 2 均为正，且 Weight 1 + Weight 2 = 1；车次的代价定义为 Weight 1 * Distance + Weight 2 * Time，此处 Distance 和 Time 为本次列车的里程(km)和时间(分钟)的绝对数值。

取绝对数值的原因是，对目前全国铁路的所有车次，里程和时间两个数量都在相同量级，且主流的 160km/h - 250km/h 的速度级别的列车的均速大约是 1km/分钟，这些车次的里程和时间在数值上更为接近，所以直接取里程和时间的绝对数值并加权计算本车次的代价是合理的。

我们让用户自行输入数值的目的，是让用户根据自己的喜好来分配对里程和时间两个量的侧重程度。这是基于这样一个事实：全国各铁路线的路况、建设标准等差异较大，导致不同铁路线路上运行的列车均速差异较大，这样就可能导致里程较短的线路耗时较长。例如，两个主要城市徐州和青岛之间，有两条线路可选择，一条是经京沪线到济南，再经胶济线到青岛，如图 6 中黑色曲线所示；另一条是经枣临线(枣庄至临沂铁路)、胶新线(胶州至新沂铁路)到青岛，如图 6 中蓝色曲线所示。显而易见，第二种交通方案的里程较近；但由于京沪线和胶济线为国家干线，技术标准较高，为国铁 I 级复线铁路，设计时速 160km/h，而枣临线和胶新线的技术标准相对较低，仅为 120km/h 单线铁路；所以实际上途经两条线路的耗时差异不大，走前一条线路的车次运行时间最短为 485 分钟(Z274)，后一条线路最短为 454 分钟(K1050)。若再考虑到铁路路况的因素，干线铁路的路况显然会更好，列车运行时更平稳，所以前一种交通方案的舒适度会更高。此外还有宝成线、宁安线等多个类似的例子，此处略去。

图 6 徐州-青岛的两条线路，两者里程相差较大，但运行时间相差不大

黑色曲线(京沪线-胶济线):总里程 697km；160km/h 国家干线，线路路况较好

蓝色曲线(枣临线-胶新线):总里程 490km；120km/h 单线铁路，线路工程标准低

这样一来，用户可以根据自己的需求分配对里程和时间两个指标的侧重程度，必须说明的是，铁路的计价规则是(相同速度等级的车次)按里程计费，因此用户实质上也就是在时间和票价中做出权衡。若旅客希望旅行耗时尽可能低，而不在意里程(票价)，则可将时间的权重设为 100%；若希望票价尽可能低，则可将里程的权重设为 100%；若对两者都不太敏感，则可将时间和里程的权重都设为 50%；考虑到可能出现图 5 中的时间相差不大但里程相差较大的情况，还可将时间设为 90%，里程设为 10%，这样就可以在总时间接近时筛选出里程较短的列车。我们提供这一设置项也可认为是专家知识的体现。

另一个值得讨论的问题是其中“模式匹配”语句的具体实现方式。以上推理引擎源码中，我们使用了这样两行伪代码来表示模式匹配子句：

```
Train list 1 OF Transit City := {Train
    FOR Train IN MySQL train_no_detail
    WHERE Train: Start Station -> Transit City}
Train list 2 OF Transit City := {Train
    FOR Train IN MySQL train_no_detail
    WHERE Train: Transit City -> Dest Station}
```

这两行的含义是，查找所有从出发站到中转地的车次，装入集合 Train list 1；查找所有从中转地到到达站的车次，装入集合 Train list 2。这是伪代码表示，具体应该如何实现呢？

考虑到我们已经使用 MySQL 数据库存储车次信息[5][6][7]，我们可以用 SQL 语句进行该模式匹配，将具体的查询工作交给数据库来完成：[8]

```
SELECT train_no
FROM train_no_detail
WHERE station in (?, ?)
GROUP BY train_no
HAVING count(*) = 2
```

前面已经讨论过，数据库中的每个条目是列车时刻表的一个表项，也就是列车的一个经停站。我们在调用这一 SQL 语句时，就在上面的问号处分别填入出发站和中转站的站名(也可以是中转站和到达站的站名)，则这一 SQL 语句的直观含义就是，在数据库 train_no_detail 中找出所有站名等于出发站的条目和站名等于中转站的条目，按车次号分组，筛选出那些在查找结果中有两个条目的车次，就得到了经过出发站和中转站的车次。

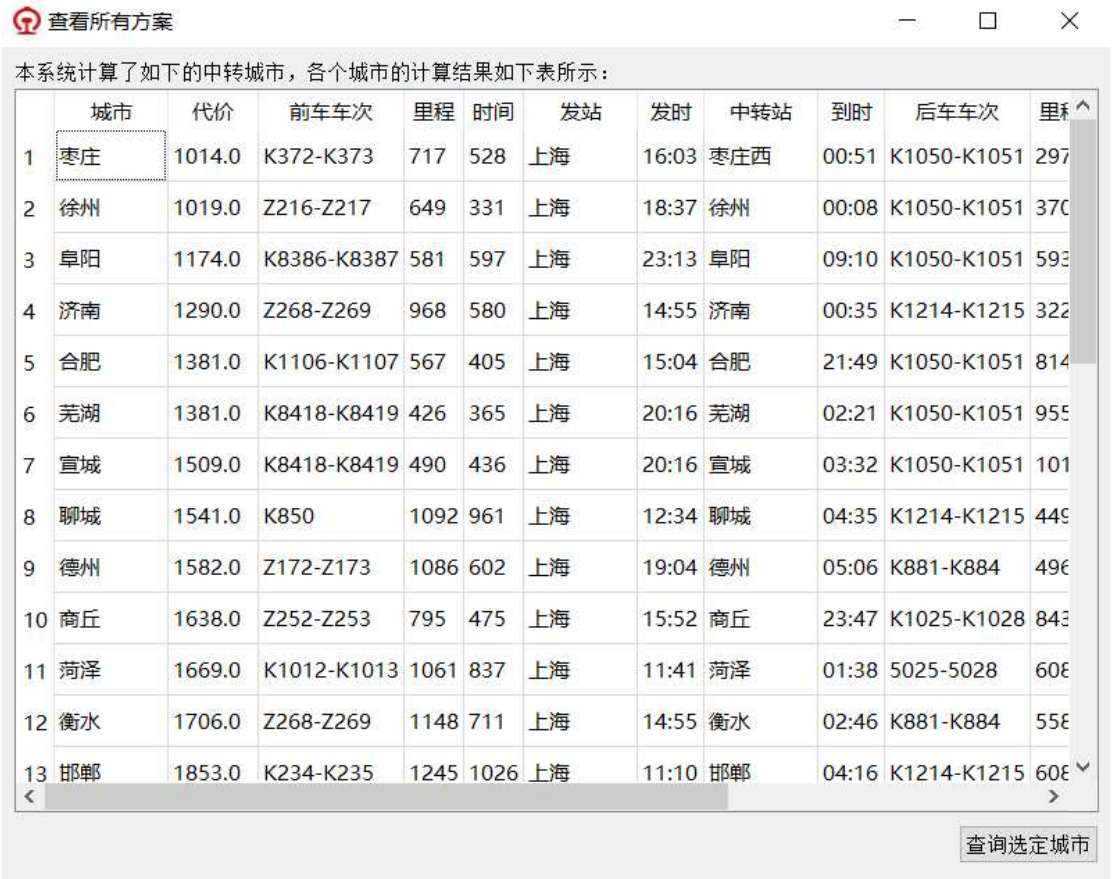
当然实际场景中还需要处理反向车次、环形车次、用户设定的偏好车型等问题，但都只需要在上面 SQL 语句的基础上扩展就可以了。

以上就是本专家系统的推理引擎的工作原理和一些实现细节。

3.3 解释机制

推理引擎完成一次计算后，用户可点击主界面（如图 5 所示）右侧的“查看所有方案”按钮，以查看推理引擎的解释（如图 7 所示）。

点击该按钮将显示推理引擎尝试计算过的所有中转城市。因为我们的推荐中转地是从所有中转地中选取代价最小的一个，所以此处还会显示推理引擎计算出的所有城市的代价；代价的计算式中有用户输入的权重、里程、时间这些项，其中用户输入的权重对用户而言是已知的，我们需要展示各个城市作为中转城市的总里程和时间；对一个城市而言，其作中转城市的代价是定义为其作中转城市的所有中转换乘方案中代价最小的一个，所以我们还需要展示这代价最小的一个方案对应的具体车次是哪两趟。这样就完成了从已知数据到推理结果的解释。



	城市	代价	前车车次	里程	时间	发站	发时	中转站	到时	后车车次	里
1	枣庄	1014.0	K372-K373	717	528	上海	16:03	枣庄西	00:51	K1050-K1051	297
2	徐州	1019.0	Z216-Z217	649	331	上海	18:37	徐州	00:08	K1050-K1051	370
3	阜阳	1174.0	K8386-K8387	581	597	上海	23:13	阜阳	09:10	K1050-K1051	593
4	济南	1290.0	Z268-Z269	968	580	上海	14:55	济南	00:35	K1214-K1215	322
5	合肥	1381.0	K1106-K1107	567	405	上海	15:04	合肥	21:49	K1050-K1051	814
6	芜湖	1381.0	K8418-K8419	426	365	上海	20:16	芜湖	02:21	K1050-K1051	955
7	宣城	1509.0	K8418-K8419	490	436	上海	20:16	宣城	03:32	K1050-K1051	101
8	聊城	1541.0	K850	1092	961	上海	12:34	聊城	04:35	K1214-K1215	449
9	德州	1582.0	Z172-Z173	1086	602	上海	19:04	德州	05:06	K881-K884	496
10	商丘	1638.0	Z252-Z253	795	475	上海	15:52	商丘	23:47	K1025-K1028	843
11	菏泽	1669.0	K1012-K1013	1061	837	上海	11:41	菏泽	01:38	5025-5028	608
12	衡水	1706.0	Z268-Z269	1148	711	上海	14:55	衡水	02:46	K881-K884	558
13	邯郸	1853.0	K234-K235	1245	1026	上海	11:10	邯郸	04:16	K1214-K1215	608

图 7 解释界面

3.4 用户界面

为提高易用性，本系统实现了图形化用户界面(GUI)（参考了[9][10][11]），其中也包括开发人员界面。主界面如图 5 所示。

	车次	里程	时间	发站	发时	到站	到时
1	Z216-Z217	649	331	上海	18:37	徐州	00:08
2	Z164-Z165	649	359	上海	20:10	徐州	02:09
3	Z268-Z269	649	373	上海	14:55	徐州	21:08
4	Z172-Z173	649	376	上海	19:04	徐州	01:20
5	Z40-Z41	649	383	上海	19:40	徐州	02:03
6	Z252-Z253	649	391	上海	15:52	徐州	22:23
7	T138-T139	649	392	上海	15:52	徐州	22:24
8	T116-T117	649	405	上海	15:32	徐州	22:17

	车次	里程	时间	发站	发时	到站	到时
1	K1050-K1051	370	454	徐州	22:48	青岛北	06:10
2	K1025-K1028	697	507	徐州	23:10	青岛北	07:22
3	K1633-K1636	697	606	徐州	01:54	青岛北	11:41
4	1563-1566	697	607	徐州	01:51	青岛北	11:42

图 5 主界面

用户可在上半部分修改出发站、到达站、偏好车型、强制中转地、“里程”和“时间”两项属性在计算代价时的占比等设置，该代价为推理引擎选出最佳中转城市的依据（参见 3.2 节的源码）。其中，出发站、到达站、强制中转三项设置是以文本框的形式，用户可直接输入站名和地名；偏好车型的设置通过一个组合框来完成，用户在全部列车、高铁动车、普速列车中选择一项；代价计算设置部分，“里程”和“时间”前面各有一个单选按钮，后面也各有一个数字输入框，用户点击单选按钮可将对应项的权重置为 100%，另一项置为 0%，也可根据自己的喜好手动输入权重，当鼠标指针指向“代价的计算方式？”字样时还会显示提示信息。如图 8 所示。

a) b) c)

图 8 主界面用户设置部分

用户完成设置后，点击“查询”按钮，即触发推理引擎开始工作。推理引擎计算过程中，主界面底部的

进度条会显示进度。推理引擎计算完毕后，会将主要车站列表中代价最小的中转城市及在该地中转的总里程、总时间、可选车次列表显示到用户界面的下半部分。为与系统默认字体区别，推理引擎计算的结果会以绿色字体显示，如图 9 所示。

	车次	里程	时间	发站	发时	到站	到时
1	Z216-Z217	649	331	上海	18:37	徐州	00:08
2	Z164-Z165	649	359	上海	20:10	徐州	02:09
3	Z268-Z269	649	373	上海	14:55	徐州	21:08
4	Z172-Z173	649	376	上海	19:04	徐州	01:20
5	Z40-Z41	649	383	上海	19:40	徐州	02:03
6	Z252-Z253	649	391	上海	15:52	徐州	22:23
7	T138-T139	649	392	上海	15:52	徐州	22:24
8	T116-T117	649	405	上海	15:32	徐州	22:17

	车次	里程	时间	发站	发时	到站	到时
1	K1050-K1051	370	454	徐州	22:48	青岛北	06:10
2	K1025-K1028	697	507	徐州	23:10	青岛北	07:22
3	K1633-K1636	697	606	徐州	01:54	青岛北	11:41
4	1563-1566	697	607	徐州	01:51	青岛北	11:42

图 9 主界面查询结果显示部分

用户可以点击右侧“查看所有方案”按钮，以查看推理引擎的解释。本系统将弹出一个子窗口，显示推理引擎尝试计算过的所有中转城市，如图 7 所示。在该界面上选中一个城市并点击确定，将自动把该城市填入“强制中转”中，并查询，从而显示在该城市中转换乘的详细结果，供用户参考，如图 10 所示；这样就允许用户在各个中转城市中手动选择自己较熟悉或较偏好的城市作为中转地。

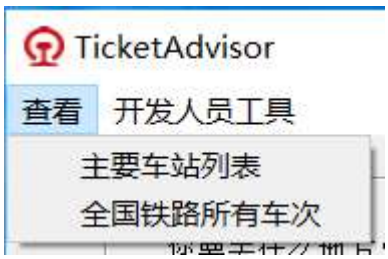
	城市	代价	前车车次	里程	时间	发站	发时	中转站	到时	后车车次	里程
1	枣庄	1014.0	K372-K373	717	528	上海	16:03	枣庄西	00:51	K1050-K1051	297
2	徐州	1019.0	Z216-Z217	649	331	上海	18:37	徐州	00:08	K1050-K1051	370
3	阜阳	1174.0	K8386-K8387	581	597	上海	23:13	阜阳	09:10	K1050-K1051	593
4	济南	1290.0	Z268-Z269	968	580	上海	14:55	济南	00:35	K1214-K1215	322
5	合肥	1381.0	K1106-K1107	567	405	上海	15:04	合肥	21:49	K1050-K1051	814
6	芜湖	1381.0	K8418-K8419	426	365	上海	20:16	芜湖	02:21	K1050-K1051	955
7	宣城	1509.0	K8418-K8419	490	436	上海	20:16	宣城	03:32	K1050-K1051	101
8	聊城	1541.0	K850	1092	961	上海	12:34	聊城	04:35	K1214-K1215	449
9	德州	1582.0	Z172-Z173	1086	602	上海	19:04	德州	05:06	K881-K884	496
10	商丘	1638.0	Z252-Z253	795	475	上海	15:52	商丘	23:47	K1025-K1028	843
11	菏泽	1669.0	K1012-K1013	1061	837	上海	11:41	菏泽	01:38	5025-5028	608
12	衡水	1706.0	Z268-Z269	1148	711	上海	14:55	衡水	02:46	K881-K884	558
13	邯郸	1853.0	K234-K235	1245	1026	上海	11:10	邯郸	04:16	K1214-K1215	608

图 7 “查看所有方案”界面(解释界面)

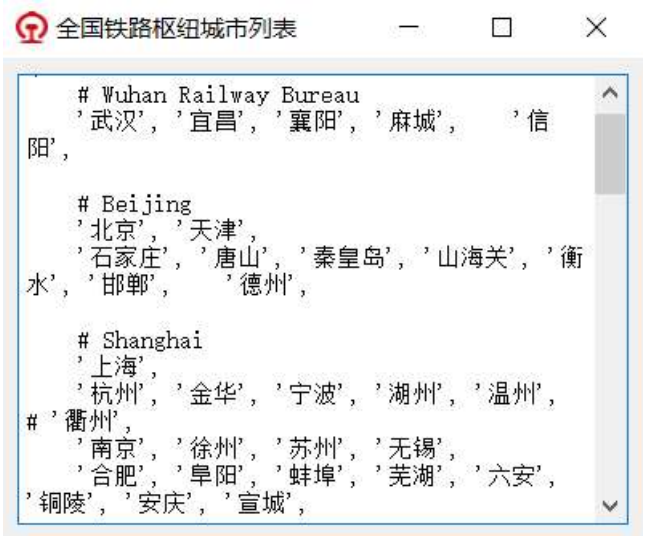
图 10 在解释界面选定城市，并点击“查询选定城市”

这样一来，将允许如下一种应用场景：用户查询了上海-青岛北的普速列车，权重设为“100%里程”，推荐的中转城市将是枣庄（见图 7），此时用户查看所有方案，发现在徐州中转的代价与枣庄相差不大，但总时间有所缩减，用户可选定第二行，点击“查询选定城市”，就可以看到在徐州中转换乘的信息了（如图 10）。

除此之外，主界面上方菜单栏内提供了“查看”和“开发人员工具”两个菜单。“查看”菜单提供给用户查看知识库的功能，其中的两个选项为查看主要车站(城市)列表和查看所有车次列表。用户点击后，本系统将弹出一个子窗口，显示对应的列表，如图 11 所示。在所有车次列表中，用户可以按下 Ctrl+F，在弹出的对话框中键入车次，查看所需的单车次的信息。



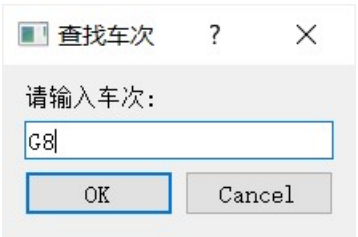
a)



b)

	车次	站序	站名	到时	发时	运行时间	里程
1	1095	1	太原	-	19:10	00:00	0
2	1095	2	榆次	19:31	19:36	00:21	27
3	1095	3	太谷	20:02	20:04	00:52	63
4	1095	4	祁县	20:21	20:23	01:11	86
5	1095	5	平遥	20:38	20:41	01:28	108
6	1095	6	介休	21:06	21:09	01:56	139
7	1095	7	灵石	21:34	21:36	02:24	166
8	1095	8	霍州	22:12	22:14	03:02	209
9	1095	9	洪洞	22:51	22:53	03:41	251
10	1095	10	临汾	23:16	23:22	04:06	274
11	1095	11	侯马	00:10	00:14	05:00	334

c)



d)

	车次	站序	站名	到时	发时	运行时间	里程
1	G8	1	上海虹桥	-	19:00	00:00	0
2	G8	2	南京南	20:07	20:09	01:07	295
3	G8	3	北京南	23:49	23:49	04:49	1318

e)

图 11 知识库查看界面

“开发人员工具”菜单提供了一个开发人员界面，其中的内容为编辑主要车站列表、编辑车次。在“编辑主要车站列表”中，领域专家可以直接编辑源代码中的相关常量：界面与用户的查看主要城市列表界面(图 10 b)相似，但添加了一个“确定”按钮，如图 12 所示，在窗口中编辑后点击“确定”，对相关常量的修改将直接同步到源文件中。因为是开发人员界面，此处可编辑的内容比用户查看知识库的界面更进一步，可以编辑各城市所有同城车站的站名。这样一来，领域专家就可以在铁路运行图调整时通过该界面简单地修改车站列表了，不必修改源码。“编辑车次”的界面也是与用户的查看所有车次界面(图 10 c)相似，增加了一个“确定”按钮，领域专家对车次的增删修改可通过该界面或 SQL 语句完成，不必修改源码。这样就在本系统中实现了知识内容和推理引擎的独立性。

图 12 开发人员界面

4 案例

以下以三个典型案例说明本专家系统的应用。

深圳-重庆。珠三角是我国经济增长的“火车头”，是我国经济最发达的地区之一；川渝地区是我国人口最为密集的地区之一，外出人口较多。每年春运期间，从珠三角地区开往川渝地区的直达列车车票常常是一票难求，由此可见这两个地区之间客运需求之旺盛。

我们尝试一下我们的专家系统能否在这条线路上求出一个较优的解。出发站和到达站输入“深圳北”和“重庆西”，偏好车型选择“高铁动车”，权重设为“时间 100%”，点击查询，结果如图 13 所示：

	车次	里程	时间	发站	发时	到站	到时
1	G2926-...	969	300	深圳北	07:58	贵阳北	12:58
2	G2922-...	969	301	深圳北	15:25	贵阳北	20:26
3	G2918	969	306	深圳北	08:58	贵阳北	14:04

	车次	里程	时间	发站	发时	到站	到时
1	D1870	344	129	贵阳东	17:41	重庆西	19:45
2	D1826	344	129	贵阳东	17:46	重庆西	19:50
3	D1807-...	344	134	贵阳东	13:45	重庆西	15:55
4	D1882-...	344	134	贵阳东	14:08	重庆西	16:18
5	D1868	344	135	贵阳东	16:54	重庆西	19:04
6	D1876	344	136	贵阳东	20:57	重庆西	23:08
7	D1878	344	139	贵阳东	21:17	重庆西	23:32
8	D1822	344	142	贵阳东	16:27	重庆西	18:43

图 13 深圳北-重庆西，本专家系统给出的方案

本专家系统的建议是，由深圳出发，沿广深高铁、贵广线到达贵阳，在贵阳中转，沿渝贵线到达重庆。中转总里程 1313 公里，中转总时间 7 时 9 分，总票价为 526.5 元。

作为对比，我们来看深圳到重庆唯一直达高铁车次 G1312 的情况，如图 14 所示：

车次	站次	站名	到达时间	开车时间	停留时间
G1312/G1313	1	深圳北	始发站	07:37	
G1312/G1313	2	广州南	08:07	08:10	3分钟
G1312/G1313	3	韶关	09:01	09:03	2分钟
G1312/G1313	4	郴州西	09:34	09:36	2分钟
G1312/G1313	5	株洲西	10:31	10:33	2分钟
G1312/G1313	6	长沙南	10:52	10:57	5分钟
G1312/G1313	7	武汉	12:18	12:25	7分钟
G1312/G1313	8	汉口	12:46	13:05	19分钟
G1312/G1313	9	汉川	13:28	13:30	2分钟
G1312/G1313	10	潜江	14:02	14:04	2分钟
G1312/G1313	11	荆州	14:30	14:32	2分钟
G1312/G1313	12	宜昌东	15:07	15:13	6分钟
G1312/G1313	13	恩施	16:57	17:03	6分钟
G1312/G1313	14	利川	17:38	17:41	3分钟
G1312/G1313	15	重庆北	19:27	终点站	

图 14 深圳到重庆唯一直达高铁 G1312 时刻表

该车次的路线是，从深圳出发，沿广深高铁、武广高铁、沪汉蓉线运行，经由广州、长沙、武汉、宜昌等地到达重庆。总里程为 2052 公里，总时间 11 时 50 分，总票价 827.5 元。

我们可以在地图上更直观地感受到差异，如图 15。

图 15 深圳-重庆，本专家系统推荐中转方案和直达车次对比图
直达方案：黑色曲线(广深高铁-武广高铁-沪汉蓉线)，
总里程 2052km，总耗时 11 时 50 分，全程票价 827.5 元
中转方案：蓝色曲线(广深高铁-贵广线-渝贵线)，
总里程 1313km，总耗时 7 时 9 分，总票价 526.5 元

即使是加上 1 小时的中转换乘间隔，本专家系统推荐方案的总耗时(8 时 9 分)也远远短于直达车次 G1312。由此可见，本专家系统确实计算出了更优的方案。

上海-青岛。长三角地区的人口密度和经济水平均在全国前列，是我国最为发达的地区之一；胶东半岛也是我国重要的经济区，其中心城市青岛的经济指标 2018 年在全国所有城市中排名第 11 位。受制于地形因素，胶东地区的地面交通并不发达，其开往长三角方向的唯一普速列车 K1184 的时刻表如图 16 所示：

车次	站次	站名	到达时间	开车时间	停留时间
K1181/K1184	1	烟台	始发站	09:30	
K1181/K1184	2	桃村	10:21	10:24	3分钟
K1181/K1184	3	徐家店	10:39	10:42	3分钟
K1181/K1184	4	莱阳	11:06	11:09	3分钟
K1181/K1184	5	莱西	11:27	11:30	3分钟
K1181/K1184	6	蓝村	12:20	12:28	8分钟
K1181/K1184	7	潍坊	13:49	13:54	5分钟
K1181/K1184	8	淄博	14:58	15:04	6分钟
K1181/K1184	9	大明湖	16:40	16:54	14分钟
K1181/K1184	10	泰山	17:53	17:56	3分钟
K1181/K1184	11	兖州	18:54	19:00	6分钟
K1181/K1184	12	邹城	19:15	19:48	33分钟
K1181/K1184	13	枣庄西	20:33	20:37	4分钟
K1181/K1184	14	徐州	21:37	22:03	26分钟
K1181/K1184	15	宿州	22:50	22:53	3分钟
K1181/K1184	16	蚌埠	23:52	00:05	13分钟
K1181/K1184	17	南京	02:04	02:12	8分钟
K1181/K1184	18	常州	03:32	03:36	4分钟
K1181/K1184	19	无锡	04:03	04:07	4分钟
K1181/K1184	20	苏州	04:38	04:42	4分钟
K1181/K1184	21	上海南	06:20	06:51	31分钟

图 16 胶东地区和长三角地区间唯一普速列车 K1184 时刻表

乘 K1184 次列车从青岛(蓝村)到上海南站，需从济南绕行，票价为 280.5 元，里程为 1346km，时间为 17 时 52 分。

下面我们来尝试本专家系统是否可以提出更好的普速列车交通方案。出发站和到达站输入“上海”和“青岛北”，偏好车型选择“普速列车”，权重设为“时间 100%”，点击查询，结果如图 17 所示：

为您推荐的中转地： 徐州　　中转总里程(公里)： 1019　　中转总时间(分钟)： 785　　查看所有方案

车次一览：

上海-徐州

	车次	里程	时间	发站	发时	到站	到时
1	Z216-Z217	649	331	上海	18:37	徐州	00:08
2	Z164-Z165	649	359	上海	20:10	徐州	02:09
3	Z268-Z269	649	373	上海	14:55	徐州	21:08
4	Z172-Z173	649	376	上海	19:04	徐州	01:20
5	Z40-Z41	649	383	上海	19:40	徐州	02:03
6	Z252-Z253	649	391	上海	15:52	徐州	22:23
7	T138-T139	649	392	上海	15:52	徐州	22:24
8	T116-T117	649	405	上海	15:32	徐州	22:17

徐州-青岛北

	车次	里程	时间	发站	发时	到站	到时
1	K1050-K1051	370	454	徐州	22:48	青岛北	06:10
2	K1025-K1028	697	507	徐州	23:10	青岛北	07:22
3	K1633-K1636	697	606	徐州	01:54	青岛北	11:41
4	1563-1566	697	607	徐州	01:51	青岛北	11:42

图 17 上海-青岛北，本专家系统给出的方案

本专家系统推荐的方案是，从上海出发，沿京沪线到徐州，在徐州中转换乘，沿枣临线、胶新线到青岛。中转总里程 1019 公里，中转总时间 13 时 5 分，总票价为 214 元。

地图对比如图 18 所示。由此可见本专家系统又一次给出了总里程、总时间、总票价都更低的方案。

图 18 上海-青岛，本专家系统推荐中转方案和直达车次对比图

直达方案：黑色曲线(京沪线-胶济线)，票价为 280.5 元，里程为 1346km，时间为 17 时 52 分

中转方案：蓝色曲线(京沪线-枣临线-胶新线)，总票价 214 元，里程为 1019km，中转总时间 13 时 5 分

武汉-济南。本案例来源于笔者的一个同学，家在武汉，在山东大学上学。受制于安徽北部线路条件等诸多因素，目前从武汉开往济南的所有高铁动车都必须从郑州绕行，从郑州开出后又需到达徐州再转向济南，在地图上画出两个大锐角，非常不经济；且列车还需在郑州和济南换向，列车换向时旅客必须旋转座位，降低了体验感。

以 G258 为例，其时刻表如图 19 所示。里程 1182 公里，时间 5 时 11 分，票价 524.5 元。

车次	站次	站名	到达时间	开车时间	停留时间
G258/G259	1	武汉	始发站	07:11	
G258/G259	2	信阳东	07:54	07:56	2分钟
G258/G259	3	漯河西	08:34	08:36	2分钟
G258/G259	4	郑州东	09:10	09:14	4分钟
G258/G259	5	徐州东	10:44	11:11	27分钟
G258/G259	6	曲阜东	11:48	11:50	2分钟
G258/G259	7	济南西	12:22	12:24	2分钟

图 19 一趟典型的武汉到济南直达高铁 G258 时刻表

尝试运行本专家系统，在出发站和到达站输入“武汉”和“济南西”，偏好车型选择“高铁动车”，权重设为“里程 90%、时间 10%”，点击查询，结果如图 20 所示：

为您推荐的中转地： 合肥　　中转总里程(公里)： 847　　中转总时间(分钟)： 256　　查看所有方案

车次一览：

武汉-合肥

	车次	里程	时间	发站	发时	到站	到时
1	G597-G600	357	116	武汉	15:01	合肥南	16:57
2	G4827-G4830	357	118	武汉	20:28	合肥南	22:26
3	G575-G578	357	121	武汉	15:11	合肥南	17:08
4	G675-G678	357	125	武汉	13:35	合肥南	15:40
5	G4823-G4826	357	126	武汉	19:33	合肥南	21:39
6	D3093-D3096	357	129	武汉	08:15	合肥南	10:24
7	G1128	357	130	武汉	19:49	合肥南	21:54
8	G593-G596	357	135	武汉	14:20	合肥南	16:35

合肥-济南西

	车次	里程	时间	发站	发时	到站	到时
1	G272	490	158	合肥南	18:17	济南西	20:55
2	G352	490	166	合肥南	09:58	济南西	12:41
3	G262	490	176	合肥南	07:06	济南西	10:02
4	G270	490	176	合肥南	16:31	济南西	19:22
5	G162	490	181	合肥南	12:00	济南西	14:55
6	G348	490	202	合肥南	19:17	济南西	22:32
7	G268	490	205	合肥南	13:02	济南西	16:27
8	G4908	490	215	合肥南	18:36	济南西	22:04

图 20 武汉-济南西，本专家系统给出的方案

本专家系统推荐的方案是，从武汉出发，在合肥中转换乘，最终到达济南。中转总里程 847 公里，中转总时间 4 时 16 分，总票价为 397 元。

地图对比如图 21 所示。虽然考虑到中转换乘的间隔，耗时相差不大，但因总里程的缩减，我们的票价降低了将近 25%，这对学生而言是相当经济的；而且换乘的方案不需换向，也提高了舒适度。

图 21 武汉-济南，本专家系统推荐中转方案和直达车次对比图

直达方案：黑色曲线(京武高铁-徐兰高铁-京沪高铁)，里程 1182 公里，时间 5 时 11 分，票价 524.5 元

中转方案：蓝色曲线(京沪线-枣临线-胶新线)，里程 847 公里，时间 4 时 16 分，票价 397 元

5 讨论

实际因素的影响，和本专家系统的可改进之处。我们的专家系统根据列车运行耗时和里程为旅客推荐代价较低的中转换乘方案，但在实际操作中，还面临各种各样实际因素的影响。

例如，若需计算票价，考虑到旅客的实际需求，白天运行的列车应按硬座计价，夜晚运行的列车应按硬卧计价，而不仅仅是按照简单的票价里程公式来计算，或也可让旅客自行指定坐席类型。（因为笔者未能获得带票价信息的车次数据库，本专家系统尚未添加该功能）

再如，每个车站的结构都不是完全相同的，旅客在陌生车站换乘时，往往需要预留更长的时间；一个极端的例子是重庆北站，其北广场和南广场并不互通，需乘一站轻轨才能抵达，因此创造过一日导致三千人误车的记录[？？？]，更有段子声称“有到重庆北站南广场的乘客，请在重庆北站下车；有到重庆北站北广场的乘客，请在龙头寺站下车；有到龙头寺汽车站的乘客，请在重庆北站下车；有到龙头寺汽车北站的乘客，请在龙头寺站下车；有到重庆汽车北站的乘客，请在红旗河沟下车”。由此我们可见在陌生车站换乘状况的复杂程度，可搜集全国所有这类车站的信息，在本专家系统添加在相关车站换乘时的提醒信息，让旅客提前知晓。

再如列车晚点的问题，这是中转换乘上一个非常重要的影响因素，若前车晚点，可能会导致旅客赶不上后车。为了避免这种情况发生，笔者平时帮同学朋友设计转车方案时，都需在“高铁管家”等 app 中查询前车的晚点情况（如图 22 所示），若前车的晚点率高，则必须为后车预留更多的时间。这一因素是相当重要的，但因为这类软件和网站没有提供 API，我们没有很好的办法来将晚点查询功能集成到本专家系统中。后期若能找到相关组件，则可以为本专家系统添加晚点查询功能。

再如区间限售和票额限售，因为该因素涉及到专家系统知识表示的问题，笔者将在下一页单独讨论这一因素。

图 22 “高铁管家“app 中的晚点率查询功能

铁路售票中的区间限售和票额限售问题，及其体现出的知识表示难点。区间限售指的是，为了保证长途旅客的利益，短途区间不放票，即，旅客购买长途车车票时，若查询较短的区间，则显示为无票，旅客无法购买短途区间的车票（如图 23 所示）。

图 23 区间限售

至于票额限售，首先讲一下票额的概念：每一个车站、每一个车次，根据车站大小，都有车站票额限制。例如，截至 2017-10-6 19:40:51，查询 10 月 20 日 Z54 次列车，武汉-北京区间的硬卧余票 0 张，长沙-北京区间的硬卧余票 35 张。晚上车的较短区间，余票数量反而较少，就是因为有票额限制。有的硬卧铺位只售长沙站，有的只售武昌站，这样的席位数量就是这个车站在当次车上的票额。票额限售就是，因票额限制，旅客从非始发站上车可能买不到票，这样就照顾了始发站的利益。

对于被限售的车次，如果进一步观察，往往还会有一个固定的放票规律。例如，2019 年 9 月 29 日汉口-乌鲁木齐的 Z292 次列车，初放票时在汉口-兰州西这个区间是买不到票的。经过持续的观察我们发现，在 9 月 25 日时，9 月 27 日的 Z292 解除了区间限售，汉口-兰州西区间可以买到票；9 月 26 日时，9 月 28 日的 Z292 解除了区间限售，到兰州可以买到票；由此我们就可以猜测，该车次极有可能是提前两天解除区间限售，因而在 9 月 27 日我们顺利买到了 9 月 29 日 Z292 次列车在汉口-兰州西区间的车票。

不同车次、不同区间、不同路局还有很多不同的情况，笔者对这类知识很有研究。但因为这类知识相当复杂、总量很大，而且不到特定的任务场景下很难想到，所以很难被形式化并加到本专家系统中。这也是人工智能界共同的挑战——我们知道，人工智能有三个坎：我们具有知识、我们会表达知识、我们能够把知识形式化后输入给计算机。常识的数据量巨大，而很难越过后两个坎，因为其难以形式化，而且不到特定的任务场景下我们很难想到它，如鸽子与方糖的实验。现在一般采取的做法是让计算机学习大量的事例，但是提供的例子很难穷尽所有可能的场景，所以仍然难以避免所有漏洞。知识表示（尤其是常识表示）的这一问题，现在仍然是有待解决的。

6 总结

本文介绍了 TicketAdvisor，一款根据用户出行需求推荐铁路中转换乘方案的专家系统。本专家系统的知识有显示的表征，将铁路知识表达在列车时刻表这一框架中；推理引擎使用模式匹配语句，在数据库中查找所有符合用户设定条件的车次实例；还提供了图形化的用户界面、开发人员界面和解释机制。本专家系统在三个典型案例(深圳-重庆、上海-青岛、武汉-济南)中都取得了较好的应用效果，有一定的实用价值。

参考文献

[1] M. Negnevitsky, “Artificial Intelligence: A Guide to Intelligent Systems”. 北京:机械工业出版社, 2012: 16, 86-108.

[2] http://blog.sina.com.cn/s/blog_842bf5cd010168yt.html

Marvin Minsky--"人工智能之父"和框架理论的创立者

[3] https://gitee.com/zsyoung01/train

train: 12306 爬虫，抓取指定城市，始发和经过的所有车次信息

[4] https://github.com/metromancn/Parse12306

Parse12306: 分析 12306 获取全国列车数据

[5] https://www.cnblogs.com/ma6174/archive/2013/02/21/2920126.html

python 操作 mysql 方法和常见问题

[6] https://www.cnblogs.com/andu99/p/8980456.html

pycharm 连接 mysql 数据库

[7] https://www.baidu.com/link?url=UsNduKNqkgE-EbXL2Ypu5OQzwho-bh3MkfgMWJEge6nSuh6jtjFhqWF_csq8JagIgupXTjW05TMlNwoAEQUelX5_vls1IUTMXCzU3PtaEne&wd=&eqid=d69d79ee0000fed7000000035e0afb72

学习常见问题——PyCharm 连接 MySql 出现时区错误问题

[8] https://bbs.csdn.net/topics/90186227 列车时刻中转算法

[9] https://blog.csdn.net/panrenlong/article/details/79959069 PyQt5 的表格创建

[10] https://blog.csdn.net/jia666666/article/details/81627589

PyQt5 高级界面控件之 QTableWidget

[11] https://blog.csdn.net/weixin_39449466/article/details/81008711

[PyQt5]点击主窗口弹出另一个窗口