Team Members: Arnav Joshi, Charles Cunningham

Title: Train Traffic Rescheduling Problem

Overview of what we have accomplished so far

* We are using this paper to guide our model Link
* We are using this data to test our model Link

1] Loading the data

* We were able to take the data provided on CSPLib for this problem and parse it to create decision variables
* One of the data files deals with the layout of the Courtrai train station which we will use to reschedule trains. The other files describe the trains that are supposed to be running and provide information about their arrival time and itineraries.
* The figures below show the documentation in these files to give a better sense of what they look like

```
/*****************************************************************************
* FILENAME : courtrai_layout.txt
*
* DESCRIPTION :
*       enumerate the components of the station of Courtrai :
*                       * Tracksegment: [TRACKSEGMENT] <name> <length>
*                       * Signal: [TRACKSEGMENT] <name> -1
*                       * Route: [ROUTE] <name> [List of components requested after each
train movement]
*                       * Itinerary: [ITINERARY] <name> <first track segment> <last track
segment> [List of alternative route]
* RESOURCE: Quentin Cappart, Pierre Schaus. Rescheduling Railway Traffic on Real Time
Situations using Time-Interval Variables, 2016
*
* NOTE: Track lengths are not realistically dimensioned (data missing)

* AUTHOR:   Quentin Cappart        START DATE : 23/08/16
* EMAIL: quentin.cappart@uclouvain.be
*
**/
```

```
/*********************************************************************
* FILENAME : courtrai_instance_homo_5_1.dat
*
* DESCRIPTION :
*      Describe an instance
*      each line gives the information for one train:
*                        * <trainId> <speed> <estimated arrival time> <earliest start time>
<planned completion time> <itinerary>
* RESOURCE: Quentin Cappart, Pierre Schaus. Rescheduling Railway Traffic on Real Time
Situations using Time-Interval Variables, 2017
*
* NOTE: - speeds are not realistically dimensioned (data missing)
*       - Estimated arrival time is identical to the earliest start time
*               - horizon time: 1 hour
*               - decision time: 3 minutes
*
* AUTHOR:   Quentin Cappart        START DATE : 23/08/16
* EMAIL: quentin.cappart@uclouvain.be
*
**/
```

2] Creating the decision variables:

* We were able to create the following four decision variables as described in the research paper

* These represent the start time of an activity (s), the duration of an activity (d), the end time of an activity (e) and a boolean variable representing whether or not a particular activity takes place (x)

$$s(A_i^{t,it,r}) \begin{cases} \in [eat_t, horizon] & if\ t\ on\ track\ segment\ ts \\ \in [est_t, horizon] & if\ t\ in\ front\ of\ a\ signal \end{cases} \quad (1)$$

$$d(A_i^{t,it,r}) \begin{cases} = lgt_{ts}/spd_t & if\ t\ on\ track\ segment\ ts \\ \in [0, horizon] & if\ t\ in\ front\ of\ a\ signal \end{cases} \quad (2)$$

$$e(A_i^{t,it,r}) = s(A_i^{t,it,r}) + d(A_i^{t,it,r}) \quad (3)$$

$$x(A_i^{t,it,r}) \in \{0, 1\} \quad (4)$$

3] Creating the constraints

* We were able to code up five constraints that were defined in the research paper

* Attached below is a code snippet of the precedence constraint which gives a sense of what some of our constraints look like

```
def createConstraints(self):
  self.precedenceConstraint()
  self.executionConsistencyConstraint()
  self.alternativeConstraint()
  self.noOverlapConstraint()
  self.trainOrderConsistencyConstraint()

TrainRescheduler.createConstraints = createConstraints
```

```
def precedenceConstraint(self):
  ##Constraint 5 - Precedence e(A(i,t,it,r)) = s(A(i+1,t,it,r))
  model: cp_model.CpModel = self.model
  station = self.station
  e = self.e
  s = self.s
  for t in station.trains:
    it = t.itinerary
    routes = station.itineraries[it][2:]
    for r in routes:
      route = station.routes[r]
      for i in range(0,len(route)-1):
        model.Add(e[i+1,t.id,it,r] == s[i+2,t.id,it,r])

TrainRescheduler.precedenceConstraint = precedenceConstraint
```

4] Results:

* We were able to execute a problem of rescheduling 5 trains. Find the solver stats for our problem below
* Our result is very close to the result obtained by the researchers working on the problem. Our objective function value for the 5 train problem is 279 while their objective function value is 282
* However the routes that our trains choose are significantly different. So we still need to debug our constraints and try to replicate the results of the researchers

```
CpSolverResponse:
status: OPTIMAL
objective: 279
best_bound: 279
booleans: 436
conflicts: 1539
branches: 8232
propagations: 97078
integer_propagations: 219407
walltime: 0.257764
usertime: 0.257764
deterministic_time: 0.0130935
primal_integral: 0
```

Overview of how the scope and goals of the project have been revised

* There have been no changes to the scope and goals of this project
* The goal of our project has been to solve the Train Traffic Rescheduling Problem using the model presented by the researchers
* We will try to replicate the results that the researchers got using our model

Overview of what is left to accomplish and path to doing so

* We need to debug our model so that our results match the results that the researchers got in the 5 train rescheduling problem
* Once we are confident that our model is working correctly we need to solve the 10, 15, 20 and 30 train rescheduling problems as well
* While this won't require any significant changes to our model (it is independent of the number of trains that are being rescheduled), we will need to debug any errors that we come across and check if we are able to replicate the results of the researchers