Film Shoot Scheduling Model Using Constraint Programming

by

Jay Mandyam

Project Report Submitted for EMIS 4395

May 9, 2008

Management Summary

Problem: Typically in Hollywood, the Assistant Director is in charge of creating the shooting schedule and the Unit Production Manager has the task of making sure that each and every crew position is filled for every day of shooting. On smaller, independent, lower budget movies, however, scheduling is at the mercy of the actors and the crew since they usually work for free or for a small stipend. It thus falls on the Producer to create the shooting schedule based on people's availabilities. Typically, the "easiest" way to approach this is to do it manually, with a lot of guessing, testing, and revising.

Analysis and Proposed Solution: I have worked on developing an algorithm to ease the task of the Assistant Director, mainly for a low-budget independent shoot. The algorithm is based on using a Constrained Programming approach to generate all feasible solutions, not necessarily an optimal one. The algorithm takes into account constraints such as scene requirements in terms of actors, crew members, and location, their availability and budget considerations. A software tool using the OPL constraint programming language has been developed and run using a few test cases. The the algorithm will take into account all the availabilities for each requirement of each scene and based on those availabilities will match them up to create all feasible possibilities of the order to be shot in.

1. Introduction

The motivation for this project comes from a movie I was trying to make. I knew that since my actors and my crew would be working for "Please"s and "Thank You"s that I would be at their mercy for whatever time they could devote to working on the movie. In addition, I knew ahead of time that there would only be a small window of opportunity to shoot certain hard-to-get locations. I wanted to see if there was a way that I could use some of the operational research models and methods that I learnt in my classes, to generate a feasible schedule using as constraints, the availability of my actors and crew, as well as the availability of locations. Since there are so many variables and constraints involved with such a project, the Constraint Programming approach seemed to be the better than an Optimization or Linear Programming approach.

2. The Algorthim:

The user inputs the scenes, the equipment, the crew, actors, and the locations and the availabilities of each, in addition to the scene requirements. The scenes also have attributes of setup and breakdown times, that is, how long it will take the crew to set up a scene at a certain location and break it down after shooting is completed. In addition, assuming a shooting rate of one minute per page of the script, the on-screen time is estimated. Other inputs are the number of angles from which the scene is shot, as well as the number of takes per angle.

The scenes are input as strings rather than as integers since there are some scenes in a general movie that might not be worth inputting into such a program since they have no

characters, or the actors are not necessarily significant enough to be put into such a scheduling system.

The program will try and match up all possibilities assuming everything is available. An array is constructed in two parts and then both parts are ORed together. Based on this array, a new array is populated with "1"s in places where actors are available at the appropriate time. The program will then output all possible time blocks on a certain day when a certain scene can be shot. If there is no possibility based on the data, no information will appear for the scene.

2.1 Decision Variables:

The broadest decision variable is the availability of each component. The user can input the availability on a per day basis or on a block basis depending on what the availability of the component is. For example, if the camera being used is available from 9:00 A.M. on a Friday through 9:00 A.M. on a Monday, the user can just input that block of time. Or if the camera is only available from 9:00 A.M. to 5:00 P.M. on a per day basis, the user can then input the availability per day for however many days the camera is budgeted for.

Location Availability: If a location is only available for a certain period of time, the amount of time it takes to set up each scene and break down each scene being shot in that location needs to be factored in since the shoot can only be in that location for that allotted block of time. Therefore an estimated set-up and breakdown time must be input. In addition, the way the program knows how long the shooting time is, is based on the shooting ratio of one page of script is equal to one minute of onscreen time. The user inputs the estimated number of angles and the estimated number of takes per each angle,

and the program will multiply these three numbers together to come up with an estimate of how much time on location will be spent shooting.

Number of takes and number of shots/angles: The number of takes is the number of times that the scene is shot from one particular angle. This program assumes that all takes for any angle are running for the entire length of the scene. However, realistically, directors usually only will run part of the scene from each angle. This assumption in the program would over budget the time spent on that scene. The program also assumes that any rehearsal time is factored into the setup time of the scene.

Time Step: The time step is the smallest number of minutes that can be accounted for. If the time step is set to 60 minutes, the program will schedule things to the nearest hour. If the time step is set to 30 minutes, the program will schedule things to the nearest half an hour. The time step is up to the user, depending on how accurate or specific they want to be. However, the smaller the time step, the longer it will take for the program to run and generate an output.

Inter-cut Scenes: Inter-cut scenes are two scenes that, in the script, are cut together back and forth so that they appear to be multiple scenes. A telephone conversation is a typical example of this type of scene. The program is not smart enough to know that one side of an inter-cut scene can be filmed at one time and treated as one scene. The way to input this type of scene is to either input it the first time the scene occurs and not account for the other scenes, or the user can "fool" the program with the rest of the scenes by giving it a setup time and breakdown time of 0.

The code for the algorithm is given in Appendix A.

Appendix B presents a test case and discusses the results obtained.

3. Conclusions and Recommendations

While the algorithm and the code seem to perform well in producing feasible shooting schedules for the test cases, its application in a real-world situation needs to be evaluated. Whether the shooting schedule determined from this program will work in practice or whether or it can handle a full scale feature length film still needs to be determined.
I will be testing this out over the summer when I use this program to generate a schedule for a feature film which I am planning to shoot and see if the shooting schedule output by the program can actually be used. A modification I am planning to make is give a more accurate time step so that takes and shots can be better accounted for, by not treating them as the whole scene, but instead making it depend on what part of the scene is being shot. Other modifications I intend to make to the program would be to take into account transportation times for people, consumable items such as batteries, light bulbs, and makeup supplies, and rehearsal times.

Reference

Lustig, Irv. "Getting Started: Modeling and Solving Scheduling Problems with Constraint Programming using ILOG OPL Studio".

Acknowledgements

I deeply appreciate the helpful discussions I have had with Dr. Richard Barr during the course of this project.

Appendix A

OPL Model Source Code:

```
/*********************************************
 * OPL 5.5 Model
 *
 * This OPL model creates a scehduling system
 * for a Movie. It takes in a list of Actors,
 * Location, Crew and Equipment and their
 * availability. It also takes a list of Scenes
 * and a list of requirements for each scene
 * (Actors, Locations, Crew and Equipment).
 * It would serach the schedules and come up
 * with a list of possible dates where all the
 * requirements for the scene are met.
 *
 * Note that it takes into account the Scene Setup
 * Scene Breakdown, Shooting Length, # of Shots per scene,
 * and # of Camera angles per shot.

 * Usage: Fill the basic.dat file based on your
 *       data. And hit Run.
 *
 *********************************************/

using CP;

tuple DateTimeAvailable { // Endpoints inclusive
   string start;        //format : 3/12/1997 12:45:00
   string end;          //format : 3/12/1997 12:45:00
}

tuple DateTimeAvailableInt { // Endpoints inclusive
   int start;        //format : time
   int end;          //format : time
}

tuple SceneDetail {
   key string name;   //name of the scene
   int up;         //setup time in minutes
   int down;     //breakdown time in minutes
   int len;          //scene length in minutes
   int shots;         //number of shots / scene
   int angles;       //number of angles / shot
}
```

```
tuple SceneDetailTimestep {
   int up;          //setup time in TimeSteps
   int down;        //breakdown time in TimeSteps
   int shootlen;    //scene shoot length (len*shots*angles) in TimeSteps
   int total;       // total = up + down + len
}

{string} Actor = ...;
{string} Location = ...;
{string} Crew = ...;
{string} Equipment = ...;

{SceneDetail} SceneDetails = ...;
{string} Scene = {n | <n,u,d,l,s,a> in SceneDetails};

SceneDetailTimestep SceneDetailTimesteps[s in Scene];

//show detailed timeslots for scenes.
int showTimeSlots = ...;


// Define availability of Actor, Location, Crew, Equipment

{DateTimeAvailable} actorAvailable[Actor] = ...;
{DateTimeAvailable} locationAvailable[Location] = ...;
{DateTimeAvailable} crewAvailable[Crew] = ...;
{DateTimeAvailable} equipmentAvailable[Equipment] = ...;

{DateTimeAvailableInt} actorAvailableInt[Actor];
{DateTimeAvailableInt} locationAvailableInt[Location];
{DateTimeAvailableInt} crewAvailableInt[Crew];
{DateTimeAvailableInt} equipmentAvailableInt[Equipment];

// Scene Requirements

{string} ActorReqs[Scene] = ...;
{string} LocationReqs[Scene] = ...;
{string} CrewReqs[Scene] = ...;
{string} EquipmentReqs[Scene] = ...;

int stepSize = ...;

int timeStep = 1000*60*stepSize;

int currentTime;
```

```
//
//OPL doesn't support dates. So convert all dates to Integers.
//Here we convert to integer and divide by timestep, this is down to reduce computation.
//The conversion is pessimistic, for example, if the Actor is available from 8:15AM to
10:45AM
// and the timestep is set to 1 hr. The program translate this to availability from
// 9:00AM to 10:00AM.
// If the timestep is set to 30 minutes, the program will translate this to
// 8:30 to 10:30.
// If the timestep is set to 15 minutes, the program will have granularity of 15 minutes and
there
// will be no change in the available time, as the start and end times are at the 15 minute
boundary.
//
execute ConvertDateStringToInt {
  currentTime = Date().getTime() / timeStep;

  for(var a in Actor) {
    for (var l in actorAvailable[a]) {

      actorAvailableInt[a].add(Opl.ftoi(Opl.ceil(Date(l.start).getTime() / timeStep)),
                  Opl.ftoi(Opl.floor(Date(l.end).getTime() / timeStep)));
    }
  }
  for(var a1 in Location) {
    for (var l1 in locationAvailable[a1]) {
      locationAvailableInt[a1].add(Opl.ftoi(Opl.ceil(Date(l1.start).getTime() /
timeStep)),
                  Opl.ftoi(Opl.floor(Date(l1.end).getTime() / timeStep)));
    }
  }
  for(var a2 in Crew) {
    for (var l2 in crewAvailable[a2]) {
      crewAvailableInt[a2].add(Opl.ftoi(Opl.ceil(Date(l2.start).getTime() / timeStep)),
                  Opl.ftoi(Opl.floor(Date(l2.end).getTime() / timeStep)));
    }
  }
  for(var a3 in Equipment) {
    for (var l3 in equipmentAvailable[a3]) {
      equipmentAvailableInt[a3].add(Opl.ftoi(Opl.ceil(Date(l3.start).getTime() /
timeStep)),
                  Opl.ftoi(Opl.floor(Date(l3.end).getTime() / timeStep)));
    }
  }
```

```
 // writeln("  actorAvailableInt ", actorAvailableInt);
}

//
// Convert user specified time in minutes to time in terms of Timesteps.
// This enables the rest of the algorithm to deal with only increments of TimeSteps.
// The program is pessimistic as it takes an upper bound to the
// Scene setup, breakdown and length requirements.
// Shooting Legnth is calculated by multiplying lenght of a scene with #Shots and
#Camera angles.
//
execute CreateSceneDetailTimestep {
  for(var s in Scene) {
    SceneDetailTimesteps[s].up   = Math.ceil(SceneDetails.get(s).up*1000*60/timeStep);
    SceneDetailTimesteps[s].down =
Math.ceil(SceneDetails.get(s).down*1000*60/timeStep);
    SceneDetailTimesteps[s].shootlen  = Math.ceil((SceneDetails.get(s).len*
                                SceneDetails.get(s).shots*
                                SceneDetails.get(s).angles*1000*60)/timeStep);
    SceneDetailTimesteps[s].total = SceneDetailTimesteps[s].up +
                              SceneDetailTimesteps[s].down +
                              SceneDetailTimesteps[s].shootlen;
  }
}

int MAX_INT = maxint;
int MIN_INT = 0;
int rangestart = MAX_INT;
int rangeend = MIN_INT;

//
// The algorithm starts here. It first calculates the time range over which a solution is
sought.
// The range is simply the min and max of the availability specified for any
actor/location/crew/equipment
//
execute FindMinMax {
    for(var a in Actor) {
      for (var l in actorAvailableInt[a]) {
        rangestart = Math.min(rangestart, l.start);
        rangeend   = Math.max(rangeend, l.end);
      }
    }
    for(var a1 in Location) {
      for (var l1 in locationAvailableInt[a1]) {
          rangestart = Math.min(rangestart, l1.start);
```

```
        rangeend   = Math.max(rangeend, l1.end);
      }
    }
    for(var a2 in Crew) {
      for (var l2 in crewAvailableInt[a2]) {
        rangestart = Math.min(rangestart, l2.start);
        rangeend   = Math.max(rangeend, l2.end);
      }
    }
    for(var a3 in Equipment) {
      for (var l3 in equipmentAvailableInt[a3]) {
        rangestart = Math.min(rangestart, l3.start);
        rangeend   = Math.max(rangeend, l3.end);
      }
    }

    //Solution should not be in the past
    // if(rangestart < currentTime) {
    //    rangestart = currentTime;
    // }

    if(rangestart > rangeend) {
      writeln("ERROR: Could not compute the time range of the solution");
    }
    // writeln(" range = ", rangestart, " end = ", rangeend);
}

range scene_r = rangestart..rangeend;
int actorsAvailable[s in Scene][hr in scene_r];      //set to 1 when all required actors are
available for scene
int locationsAvailable[s in Scene][hr in scene_r];   //set to 1 when all required locations
are available for scene
int crewsAvailable[s in Scene][hr in scene_r];       //set to 1 when all required crews are
available for scene
int equipmentsAvailable[s in Scene][hr in scene_r];  //set to 1 when all required
equipment is available for scene

//
// Find when the scenes requirements are met.
//
execute findAvailability {
  var available;
  var newLoop;

  for (var s in Scene) {
    // writeln("actorReqs[", s, "] # of actors = ", Opl.card(ActorReqs[s]));
```

```
for(var hr in scene_r) {
 actorsAvailable[s][hr] = 0;
 newLoop = 1;
 if(Opl.card(ActorReqs[s]) > 0) {
  for(var a in ActorReqs[s]) {
   available = 0;
   newLoop = 0;
   for (var t in actorAvailableInt[a]) {
    if(hr >= t.start && hr <= t.end)
     available = 1;
   }
   if(newLoop == 1) {
    actorsAvailable[s][hr] = actorsAvailable[s][hr] * available;
   } else {
    actorsAvailable[s][hr] = available;
   }
  }
 } else {
  actorsAvailable[s][hr] = 1;
 }

 locationsAvailable[s][hr] = 0;
 newLoop = 1;
 if(Opl.card(LocationReqs[s]) > 0) {
  for(var a1 in LocationReqs[s]) {
   available = 0;
   newLoop = 0;
   for (var t1 in locationAvailableInt[a1]) {
    if(hr >= t1.start && hr <= t1.end)
     available = 1;
   }
   if(newLoop == 1) {
    locationsAvailable[s][hr] = locationsAvailable[s][hr] * available;
   } else {
    locationsAvailable[s][hr] = available;
   }
  }
 } else {
  locationsAvailable[s][hr] = 1;
 }

 crewsAvailable[s][hr] = 0;
 newLoop = 1;
 if(Opl.card(CrewReqs[s]) > 0) {
  for(var a2 in CrewReqs[s]) {
  available = 0;
```

```
      newLoop = 0;
      for (var t2 in crewAvailableInt[a2]) {
       if(hr >= t2.start && hr <= t2.end)
        available = 1;
      }
      if(newLoop == 1) {
       crewsAvailable[s][hr] = crewsAvailable[s][hr] * available;
      } else {
       crewsAvailable[s][hr] = available;
      }
      }
     } else {
      crewsAvailable[s][hr] = 1;
     }

     equipmentsAvailable[s][hr] = 0;
     newLoop = 1;

     if(Opl.card(EquipmentReqs[s]) > 0) {
      for(var a3 in EquipmentReqs[s]) {
      available = 0;
      newLoop = 0;
      for (var t3 in equipmentAvailableInt[a3]) {
       if(hr >= t3.start && hr <= t3.end)
        available = 1;
       }
       if(newLoop == 1) {
        equipmentsAvailable[s][hr] = equipmentsAvailable[s][hr] * available;
       } else {
        equipmentsAvailable[s][hr] = available;
       }
      }
     } else {
      equipmentsAvailable[s][hr] = 1;
     }

   }
  }
 //writeln(" actorsAvailable ", actorsAvailable);
 //writeln(" locationsAvailable ", locationsAvailable);
 //writeln(" crewsAvailable ", crewsAvailable);
 //writeln(" equipmentAvailable ", equipmentsAvailable);

}

//
```

```
// Main method to determine when the scene requirements are met.
// Core algorithm: 1. Find when Equipment, Crew, Location is available.
//                 An array (ecla) is constructed in two parts (ecla_a, ecla_b) and then both
parts are ORed together.
//              2. Based on the above array, populate a new array sceneHrs with '1's in places
where actors are availble at the appropriate time.
//                 Appropriate means after the Scene is setup and before the breakdown starts,
ie. during the entire shooting lenght of the scene..
//

dvar int ecla_a[s in Scene][scene_r] in 0..1;   //Equipment, Crew, Location availability
array (Look Ahead)
dvar int ecla_b[s in Scene][scene_r] in 0..1;
dvar int ecla[s in Scene][scene_r] in 0..1;

dvar int sceneHrs[s in Scene][scene_r] in 0..1;


minimize sum (s in Scene) sum (hr in scene_r) (sceneHrs[s][hr]);

subject to {
  forall(s in Scene) {
   forall(i in rangestart..rangeend-(SceneDetailTimesteps[s].total-1)) {
     ecla_a[s][i] == (sum (k in i..i+(SceneDetailTimesteps[s].total-1))
(locationsAvailable[s][k] *
                                                crewsAvailable[s][k] *
                                                equipmentsAvailable[s][k]) ==
SceneDetailTimesteps[s].total);
   }
   forall(q in rangeend-(SceneDetailTimesteps[s].total-1)..rangeend) {
     ecla_b[s][q] == (sum (k in q-(SceneDetailTimesteps[s].total-1)..1)
(locationsAvailable[s][k] *
                                                crewsAvailable[s][k] *
                                                equipmentsAvailable[s][k]) ==
SceneDetailTimesteps[s].total);
   }
   forall (r in rangestart..rangeend) {
      ecla[s][r] == (ecla_a[s][r] == 1 || ecla_b[s][r] == 1);
   }

 }

 // here we find if the actors are available at the appropriate time.
 // The trick is to count the number of 1s in the actorAvailable array during the shooting
length period.
 forall(s2 in Scene) {
```

```
   forall(i2 in rangestart..rangeend-(SceneDetailTimesteps[s2].total-1)) {
     sceneHrs[s2][i2] == ((ecla[s2][i2] == 1) && (sum (m in
i2+(SceneDetailTimesteps[s2].up)..i2+(SceneDetailTimesteps[s2].up+SceneDetailTimest
eps[s2].shootlen-1))
                    actorsAvailable[s2][m] == SceneDetailTimesteps[s2].shootlen));
   }
  }
}

execute POST {
  var printdate;
  var prev_hr;
  writeln("*****************************");
  writeln("All requirements met for scenes at the following times:");
  writeln();
  for (var s in Scene) {
    printdate = 0;
    prev_hr = 0;

    writeln("Scene ", s);
     // Actor availability.
    //writeln(" Actor available : ", SceneDetailTimesteps[s].up*timeStep/(1000*60*60) , "
hours after the start of the scene" );
    //writeln("                    ", SceneDetailTimesteps[s].down*timeStep/(1000*60*60) ,
" hours before the end of the scene" );
    //writeln("  Crew, Location, Equipment availability");

   if (showTimeSlots == 0) {

    for (var hr in scene_r) {
     if( sceneHrs[s][hr] >= 1) {
        if (printdate == 1 || hr == rangestart || hr == rangeend) {
          write("  ", Date(hr*timeStep).toLocaleString(), " - ");
         // if(hr == rangeend) {
         //     writeln("");
        //  }

          printdate = 0;
        }
        prev_hr = hr;
     } else {
        if(prev_hr != 0) {
          //
          //The array SceneHr only stores times at which Scenes can start.
          //The end time needs to be calculated by adding the total length of the scene to the
start time.
```

```
        //
        writeln(Date((prev_hr+SceneDetailTimesteps[s].total-
1)*timeStep).toLocaleString());

        prev_hr = 0;
       }
       printdate = 1;
     }
    }
   } else {
     //if detailed timeslots are required
     for (var hr2 in scene_r) {
      if( sceneHrs[s][hr2] >= 1) {
       writeln("  ", Date(hr2*timeStep).toLocaleString(), " - ",
Date((hr2+SceneDetailTimesteps[s].total)*timeStep).toLocaleString());
      }
     }
   }

  writeln();
 }
 writeln("****************************");

}
```

OPL Data Sample Format:

```
/*********************************************
 * OPL 5.5 Data
 *********************************************/

//stepSize is the smallest time increment in minutes that the program should consider.
//The smaller this number is, the more calculations/memory is needed to get to an answer.
//
stepSize = 30;
showTimeSlots = 1; // 1 to show detailed time slots, 0 to show consolidated times.

//Add all actors here
Actor = {"A1", "A2", "A3", "A4", "A5"};

//Add all locations
Location = {"L1", "L2", "L3", "L4", "L5"};

//Add all Crew
Crew = {"C1", "C2", "C3", "C4", "C5", "C6", "C7", "C8", "C9", "C10"};
```

```
//Add all Equipment
Equipment = {"E1", "E2", "E3", "E4", "E5", "E6", "E7"};

//          <name,  setup, breakdown, length,   shots, angles>
//                    in     in      in      #     #
//                  minutes minutes  minutes
SceneDetails = {
            <"S1",    120,     240,    600,     1,     1>,
            <"S2",    25,      100,    11,      1,     1>,
            <"S3",    15,      10,     2,       1,     1>
          };

// Scene requirements (Actor, Location, Crew, Equipment)

//Actors required for specific scenes
ActorReqs = #[
          "S1" : { "A1" },
          "S2" : { "A1", "A2" },
          "S3" : { "A1" }
         ]#;

//Location required for specific scenes
LocationReqs = #[
          "S1" : {"L1","L2"},
          "S2" : { "L1", "L4"},
          "S3" : {"L1"}
         ]#;

//Crew required for specific scenes
CrewReqs = #[
          "S1" : { "C1" },
          "S2" : { "C2"},
         ]#;

//Location required for specific scenes
EquipmentReqs = #[
          "S1" : { },
          "S2" : { "E2", "E3"},
         ]#;

//Actor Availability
actorAvailable = #[
            "A1":{<"4/16/2009 03:00:00","4/17/2009 08:00:00">,
                <"4/17/2009 05:00:00","4/25/2009 07:00:00">},
            "A2":{<"4/17/2009 00:00:00","4/23/2009 00:00:00">,
```

```
            <"4/24/2009 00:00:00","4/25/2009 00:00:00">,
            <"4/28/2009 00:00:00","4/29/2009 00:00:00">}
        ]#;

// Location Availability
locationAvailable = #[
        "L1":{<"4/16/2009 02:00:00","4/17/2009 04:00:00">,
            <"4/17/2009 08:00:00","4/26/2009 03:00:00">},
        "L2":{<"4/17/2009 00:00:00","4/24/2009 00:00:00">,
            <"4/24/2009 00:00:00","4/25/2009 08:00:00">,
            <"4/28/2009 00:00:00","4/29/2009 00:00:00">},
        "L3":{<"4/15/2009 00:00:00","4/17/2009 00:00:00">,
            <"4/25/2009 00:00:00","4/26/2009 00:00:00">},
        "L4":{<"4/17/2009 00:00:00","4/24/2009 00:00:00">,
            <"4/24/2009 00:00:00","4/25/2009 00:00:00">,
            <"4/28/2009 00:00:00","4/29/2009 00:00:00">}
        ]#;

// Crew Availablility
crewAvailable = #[
        "C1":{<"4/16/2009 01:00:00","4/17/2009 02:00:00">,
            <"4/17/2009 05:00:00","4/26/2009 08:00:00">},
        "C2":{<"4/17/2009 00:00:00","4/23/2009 00:00:00">,
            <"4/24/2009 00:00:00","4/25/2009 00:00:00">,
            <"4/28/2009 00:00:00","4/29/2009 00:00:00">}
        ]#;

// Equipment Availability
equipmentAvailable = #[
        "E1":{<"4/16/2009 00:00:00","4/17/2009 00:00:00">,
            <"4/25/2009 00:00:00","4/26/2009 00:00:00">},
        "E2":{<"4/17/2009 00:00:00","4/24/2009 00:00:00">,
            <"4/24/2009 00:00:00","4/25/2009 00:00:00">,
            <"4/28/2009 00:00:00","4/29/2009 00:00:00">},
        "E3":{<"4/15/2009 00:00:00","4/17/2009 00:00:00">,
            <"4/25/2009 00:00:00","4/26/2009 00:00:00">},
        "E4":{<"4/17/2009 00:00:00","4/24/2009 00:00:00">,
            <"4/24/2009 00:00:00","4/25/2009 00:00:00">,
            <"4/28/2009 00:00:00","4/29/2009 00:00:00">}
        ]#;
```

Appendix B

A sample test case is presented below:

Test Case:
"Trust Me I'm Lying"
Southern Methodist University Student Filmmakers' Association
Producer: Kenneth J. Morris

Crew	Availability
Director	3:00 PM 3/14/2008 – 8:00 PM 3/16/2008
Producer	3:00 PM 3/14/2008 – 8:00 PM 3/16/2008
Cinematographer	3:00 PM 3/14/2008 – 8:00 PM 3/16/2008
Gaffer	3:00 PM 3/14/2008 – 8:00 PM 3/16/2008
Sound Mixer	3:00 PM 3/14/2008 – 8:00 PM 3/16/2008
Boom Operator	3:00 PM 3/14/2008 – 8:00 PM 3/16/2008
Production Assistant	3:00 PM 3/14/2008 – 8:00 PM 3/16/200

Location	Availability
Parking Lot	1:00 PM 3/14/2008 – 5:00 PM 3/14/2008
HT Market	10:00 AM 3/15/2008 – 5:00 PM 3/15/2008
Coffee Shop	1:00 PM 3/16/2008 – 8:00 PM 3/16/2008
Sidewalk	Daytime
Park	Daytime

Cast	Availability
Joe	1:00 PM 3/14/2008 – 5:00 PM 3/14/2008 9:00 AM 3/15/2008 – 5:00 PM 3/15/2008 3:00 PM 3/16/2008 – 8:00 PM 3/16/2008
Karoline	1:00 PM 3/14/2008 – 5:00 PM 3/14/2008 9:00 AM 3/15/2008 – 5:00 PM 3/15/2008 3:00 PM 3/16/2008 – 8:00 PM 3/16/2008
Monica	1:00 PM 3/14/2008 – 5:00 PM 3/14/2008 9:00 AM 3/15/2008 – 5:00 PM 3/15/2008 3:00 PM 3/16/2008 – 8:00 PM 3/16/2008
Mac	9:00 AM 3/16/2008 – 5:00 PM 3/16/2008
Brandon	1:00 PM 3/14/2008 – 2:45 PM 3/14/2008 3:00 PM 3/16/2008 – 5:00 PM 3/16/2008

Equipment	Availability
Camera	9:00 AM 3/14/2008 – 9:00 AM 3/17/2008
Lights	9:00 AM 3/14/2008 – 9:00 AM 3/17/2008
Sound Mixer	9:00 AM 3/14/2008 – 9:00 AM 3/17/2008
Boom Mic	9:00 AM 3/14/2008 – 9:00 AM 3/17/2008

Scene Breakdowns:

Scene Number	Location	Cast
1	Coffee Shop	Joe
2	HT Market	Joe, Monica
3	Parking Lot	Joe
4	Parking Lot	Joe
5	HT Market	Joe, Karoline
6	Coffee Shop	Joe
7	Park	Joe
8	Sidewalk	Joe, Karoline
9	Coffee Shop	Joe
10	Parking Lot	Joe, Karoline, Brandon
11	Coffee Shop	Joe, Mac

```
/*********************************************
 * OPL 5.5 Data
 *********************************************/

//stepSize is the smallest time increment in minutes that the program should consider.
//The smaller this number is, the more calculations/memory is needed to get to an answer.
//
stepSize = 30; //30 mins
//stepSize = 15; // 15 min
showTimeSlots = 1; // 1 to show detailed time slots, 0 to show consolidated times.

//Add all actors here
Actor = {"Joe", "Karoline", "Monica", "Mac", "Brandon"};

//Add all locations
Location = {"Parking Lot", "HT Market", "Coffee Shop", "Park"};

//Add all Crew
Crew = {"Director", "Producer", "Cinematographer", "Sound Mixer", "Boom Operator",
"Gaffer", "Production Assistant"};

//Add all Equipment
Equipment = {"Camera", "Boom Mic", "Sound Mixer", "Lights"};

//          <name,  setup, breakdown, Shooting length,   # of angles/shots,  # of takes>
//                    in     in          in
//                  minutes  minutes     minutes
SceneDetails = {
            <"S1",    90,     90,    1,          2,    2>,
            <"S2",    60,     60,    1,          5,    6>,
            <"S3",    30,     30,    1,          2,    2>,
            <"S4",    30,     30,    1,          2,    2>,
            <"S5",    60,     60,    2,          5,    6>,
```

```
            <"S6",    0,      0,    1,         2,    2>,
            <"S7",    30,     30,    2,          4,    5>,
            <"S8",    30,     30,    2,          3,    4>,
            <"S9",    0,      0,    1,         2,    2>,
            <"S10",   45,     45,  2, 5, 6>,
            <"S11",    0,      0,    2,         5,    6>

};

// Scene requirements (Actor, Location, Crew, Equipment)

//Actors required for specific scenes
ActorReqs = #[
          "S1" : { "Joe" },
          "S2" : { "Joe", "Monica" },
          "S3" : { "Joe" },
          "S4" : { "Joe" },
          "S5" : { "Joe", "Karoline" },
          "S6" : { "Joe" },
          "S7" : { "Joe", "Karoline" },
          "S8" : { "Joe", "Karoline" },
          "S9" : { "Joe" },
          "S10" : { "Joe", "Karoline", "Brandon" },
          "S11" : { "Joe", "Mac" }

        ]#;

//Location required for specific scenes
LocationReqs = #[
          "S1" : {"Coffee Shop" },
          "S2" : {"HT Market"},
          "S3" : {"Parking Lot"},
          "S4" : {"Parking Lot"},
          "S5" : {"HT Market"},
          "S6" : {"Coffee Shop"},
          "S7" : {"Park"},
          "S8" : {"Parking Lot"},
          "S9" : {"Coffee Shop"},
          "S10" : {"Parking Lot"},
          "S11" : {"Coffee Shop"},
        ]#;

//Crew required for specific scenes
CrewReqs = #[
          "S1" : {"Director", "Producer", "Cinematographer", "Sound Mixer", "Boom
Operator", "Gaffer", "Production Assistant"  },
```

```
        "S2" : {"Director", "Producer", "Cinematographer", "Sound Mixer", "Boom
Operator", "Gaffer", "Production Assistant"},
        "S3" : {"Director", "Producer", "Cinematographer", "Sound Mixer", "Boom
Operator", "Gaffer", "Production Assistant"},
        "S4" : {"Director", "Producer", "Cinematographer", "Sound Mixer", "Boom
Operator", "Gaffer", "Production Assistant"},
        "S5" : {"Director", "Producer", "Cinematographer", "Sound Mixer", "Boom
Operator", "Gaffer", "Production Assistant"},
        "S6" : {"Director", "Producer", "Cinematographer", "Sound Mixer", "Boom
Operator", "Gaffer", "Production Assistant"},
        "S7" : {"Director", "Producer", "Cinematographer", "Sound Mixer", "Boom
Operator", "Gaffer", "Production Assistant"},
        "S8" : {"Director", "Producer", "Cinematographer", "Sound Mixer", "Boom
Operator", "Gaffer", "Production Assistant"},
        "S9" : {"Director", "Producer", "Cinematographer", "Sound Mixer", "Boom
Operator", "Gaffer", "Production Assistant"},
        "S10" : {"Director", "Producer", "Cinematographer", "Sound Mixer", "Boom
Operator", "Gaffer", "Production Assistant"},
        "S11" : {"Director", "Producer", "Cinematographer", "Sound Mixer", "Boom
Operator", "Gaffer", "Production Assistant"},
]#;

//Equipment required for specific scenes
EquipmentReqs = #[
        "S1" : {"Camera", "Boom Mic", "Sound Mixer", "Lights"},
        "S2" : {"Camera", "Boom Mic", "Sound Mixer", "Lights"},
        "S3" : {"Camera", "Boom Mic", "Sound Mixer", "Lights"},
        "S4" : {"Camera", "Boom Mic", "Sound Mixer", "Lights"},
        "S5" : {"Camera", "Boom Mic", "Sound Mixer", "Lights"},
        "S6" : {"Camera", "Boom Mic", "Sound Mixer", "Lights"},
        "S7" : {"Camera", "Boom Mic", "Sound Mixer", "Lights"},
        "S8" : {"Camera", "Boom Mic", "Sound Mixer", "Lights"},
        "S9" : {"Camera", "Boom Mic", "Sound Mixer", "Lights"},
        "S10" : {"Camera", "Boom Mic", "Sound Mixer", "Lights"},
        "S11" : {"Camera", "Boom Mic", "Sound Mixer", "Lights"}
      ]#;

//Actor Availability
actorAvailable = #[
          "Joe":{<"3/14/2008 13:00:00","3/14/2008 17:00:00">,
              <"3/15/2008 9:00:00","3/15/2008 17:00:00">,
              <"3/16/2008 15:00:00","3/16/2008 20:00:00">},
          "Karoline":{<"3/14/2008 13:00:00","3/14/2008 17:00:00">,
              <"3/15/2008 9:00:00","3/15/2008 17:00:00">,
              <"3/16/2008 15:00:00","3/16/2008 20:00:00">},
          "Monica":{<"3/14/2008 13:00:00","3/14/2008 17:00:00">,
```

```
        <"3/15/2008 9:00:00","3/15/2008 17:00:00">,
        <"3/16/2008 15:00:00","3/16/2008 20:00:00">},
    "Mac":{<"3/16/2008 9:00:00", "3/16/2008 17:00:00">},
    "Brandon":{<"3/14/2008 13:00:00", "3/14/2008 14:45:00">,
    <"3/16/2008 15:00:00", "3/16/2008 17:00:00">}
      ]#;

// Location Availability
locationAvailable = #[
    "Parking Lot":{<"3/14/2008 13:00:00","3/14/2008 17:00:00">},
    "HT Market":{<"3/15/2008 10:00:00","3/15/2008 17:00:00">},
    "Coffee Shop":{<"3/16/2008 13:00:00","3/16/2008 20:00:00">},
    "Park":{<"3/14/2008 09:00:00","3/15/2008 23:00:00">}
    ]#;

// Crew Availablility
crewAvailable = #[
    "Director":{<"3/14/2008 15:00:00","3/16/2008 20:00:00">},
    "Producer":{<"3/14/2008 15:00:00","3/16/2008 20:00:00">},
    "Cinematographer":{<"3/14/2008 15:00:00","3/16/2008 20:00:00">},
    "Sound Mixer":{<"3/14/2008 15:00:00","3/16/2008 20:00:00">},
    "Boom Operator":{<"3/14/2008 15:00:00","3/16/2008 20:00:00">},
    "Gaffer":{<"3/14/2008 15:00:00","3/16/2008 20:00:00">},
    "Production Assistant":{<"3/14/2008 15:00:00","3/16/2008 20:00:00">}
    ]#;

// Equipment Availability
equipmentAvailable = #[
    "Camera":{<"3/14/2008 09:00:00","3/17/2008 09:00:00">},
    "Boom Mic":{<"3/14/2008 09:00:00","3/17/2008 09:00:00">},
    "Sound Mixer":{<"3/14/2008 09:00:00","3/17/2008 09:00:00">},
    "Lights":{<"3/14/2008 09:00:00","3/17/2008 09:00:00">}
    ]#;
```

Output:

```
Post-processing final solution:
**************************
All requirements met for scenes at the following times:

Scene S1
  03/16/2008 13:30:00 000 - 03/16/2008 17:00:00 000
  03/16/2008 14:00:00 000 - 03/16/2008 17:30:00 000
  03/16/2008 14:30:00 000 - 03/16/2008 18:00:00 000
  03/16/2008 15:00:00 000 - 03/16/2008 18:30:00 000
  03/16/2008 15:30:00 000 - 03/16/2008 19:00:00 000
```

03/16/2008 16:00:00 000 - 03/16/2008 19:30:00 000
03/16/2008 16:30:00 000 - 03/16/2008 20:00:00 000
03/16/2008 17:00:00 000 - 03/16/2008 20:30:00 000

Scene S2
03/15/2008 10:00:00 000 - 03/15/2008 12:30:00 000
03/15/2008 10:30:00 000 - 03/15/2008 13:00:00 000
03/15/2008 11:00:00 000 - 03/15/2008 13:30:00 000
03/15/2008 11:30:00 000 - 03/15/2008 14:00:00 000
03/15/2008 12:00:00 000 - 03/15/2008 14:30:00 000
03/15/2008 12:30:00 000 - 03/15/2008 15:00:00 000
03/15/2008 13:00:00 000 - 03/15/2008 15:30:00 000
03/15/2008 13:30:00 000 - 03/15/2008 16:00:00 000
03/15/2008 14:00:00 000 - 03/15/2008 16:30:00 000
03/15/2008 14:30:00 000 - 03/15/2008 17:00:00 000
03/15/2008 15:00:00 000 - 03/15/2008 17:30:00 000

Scene S3
03/14/2008 15:00:00 000 - 03/14/2008 16:30:00 000
03/14/2008 15:30:00 000 - 03/14/2008 17:00:00 000
03/14/2008 16:00:00 000 - 03/14/2008 17:30:00 000

Scene S4
03/14/2008 15:00:00 000 - 03/14/2008 16:30:00 000
03/14/2008 15:30:00 000 - 03/14/2008 17:00:00 000
03/14/2008 16:00:00 000 - 03/14/2008 17:30:00 000

Scene S5
03/15/2008 10:00:00 000 - 03/15/2008 13:00:00 000
03/15/2008 10:30:00 000 - 03/15/2008 13:30:00 000
03/15/2008 11:00:00 000 - 03/15/2008 14:00:00 000
03/15/2008 11:30:00 000 - 03/15/2008 14:30:00 000
03/15/2008 12:00:00 000 - 03/15/2008 15:00:00 000
03/15/2008 12:30:00 000 - 03/15/2008 15:30:00 000
03/15/2008 13:00:00 000 - 03/15/2008 16:00:00 000
03/15/2008 13:30:00 000 - 03/15/2008 16:30:00 000
03/15/2008 14:00:00 000 - 03/15/2008 17:00:00 000
03/15/2008 14:30:00 000 - 03/15/2008 17:30:00 000

Scene S6
03/16/2008 15:00:00 000 - 03/16/2008 15:30:00 000
03/16/2008 15:30:00 000 - 03/16/2008 16:00:00 000
03/16/2008 16:00:00 000 - 03/16/2008 16:30:00 000
03/16/2008 16:30:00 000 - 03/16/2008 17:00:00 000
03/16/2008 17:00:00 000 - 03/16/2008 17:30:00 000
03/16/2008 17:30:00 000 - 03/16/2008 18:00:00 000

03/16/2008 18:00:00 000 - 03/16/2008 18:30:00 000
03/16/2008 18:30:00 000 - 03/16/2008 19:00:00 000
03/16/2008 19:00:00 000 - 03/16/2008 19:30:00 000
03/16/2008 19:30:00 000 - 03/16/2008 20:00:00 000
03/16/2008 20:00:00 000 - 03/16/2008 20:30:00 000

Scene S7
03/14/2008 15:00:00 000 - 03/14/2008 17:00:00 000
03/14/2008 15:30:00 000 - 03/14/2008 17:30:00 000
03/14/2008 16:00:00 000 - 03/14/2008 18:00:00 000
03/15/2008 08:30:00 000 - 03/15/2008 10:30:00 000
03/15/2008 09:00:00 000 - 03/15/2008 11:00:00 000
03/15/2008 09:30:00 000 - 03/15/2008 11:30:00 000
03/15/2008 10:00:00 000 - 03/15/2008 12:00:00 000
03/15/2008 10:30:00 000 - 03/15/2008 12:30:00 000
03/15/2008 11:00:00 000 - 03/15/2008 13:00:00 000
03/15/2008 11:30:00 000 - 03/15/2008 13:30:00 000
03/15/2008 12:00:00 000 - 03/15/2008 14:00:00 000
03/15/2008 12:30:00 000 - 03/15/2008 14:30:00 000
03/15/2008 13:00:00 000 - 03/15/2008 15:00:00 000
03/15/2008 13:30:00 000 - 03/15/2008 15:30:00 000
03/15/2008 14:00:00 000 - 03/15/2008 16:00:00 000
03/15/2008 14:30:00 000 - 03/15/2008 16:30:00 000
03/15/2008 15:00:00 000 - 03/15/2008 17:00:00 000
03/15/2008 15:30:00 000 - 03/15/2008 17:30:00 000
03/15/2008 16:00:00 000 - 03/15/2008 18:00:00 000

Scene S8
03/14/2008 15:00:00 000 - 03/14/2008 16:30:00 000
03/14/2008 15:30:00 000 - 03/14/2008 17:00:00 000
03/14/2008 16:00:00 000 - 03/14/2008 17:30:00 000

Scene S9
03/16/2008 15:00:00 000 - 03/16/2008 15:30:00 000
03/16/2008 15:30:00 000 - 03/16/2008 16:00:00 000
03/16/2008 16:00:00 000 - 03/16/2008 16:30:00 000
03/16/2008 16:30:00 000 - 03/16/2008 17:00:00 000
03/16/2008 17:00:00 000 - 03/16/2008 17:30:00 000
03/16/2008 17:30:00 000 - 03/16/2008 18:00:00 000
03/16/2008 18:00:00 000 - 03/16/2008 18:30:00 000
03/16/2008 18:30:00 000 - 03/16/2008 19:00:00 000
03/16/2008 19:00:00 000 - 03/16/2008 19:30:00 000
03/16/2008 19:30:00 000 - 03/16/2008 20:00:00 000
03/16/2008 20:00:00 000 - 03/16/2008 20:30:00 000

Scene S10

Scene S11
03/16/2008 13:00:00 000 - 03/16/2008 14:00:00 000
03/16/2008 13:30:00 000 - 03/16/2008 14:30:00 000
03/16/2008 14:00:00 000 - 03/16/2008 15:00:00 000
03/16/2008 14:30:00 000 - 03/16/2008 15:30:00 000
03/16/2008 15:00:00 000 - 03/16/2008 16:00:00 000
03/16/2008 15:30:00 000 - 03/16/2008 16:30:00 000
03/16/2008 16:00:00 000 - 03/16/2008 17:00:00 000
03/16/2008 16:30:00 000 - 03/16/2008 17:30:00 000

****
Done post-processing

Test Case Conclusion: There are many options for when to shoot each scene. It would be up to the user to mix and match which combination work best for the shoot. As shown by the output, Scene 10 is not feasible to be shot given the constraints at hand, so one of the availabilities will have to be looked into and adjusted if possible. Otherwise, alternate arrangements would need to be made.