Interfaces
Articles in Advance, pp. 1–10
ISSN 0092-2102 (print) | ISSN 1526-551X (online)

http://dx.doi.org/10.1287/inte.1110.0598

The Court of Appeals of Virginia Uses Integer Programming and Cloud Computing to Schedule Sessions

J. Paul Brooks
Virginia Commonwealth University, Richmond, Virginia 23284, jpbrooks@vcu.edu

Each spring, a deputy clerk of the Court of Appeals of Virginia manually schedules panel sessions and assigns judges to sessions for the following calendar year. The information technology department for the Supreme Court of Virginia, the head of the judicial branch of government in Virginia, also serves the Court of Appeals of Virginia. In the spring of 2010, the staff of the Court of Appeals of Virginia and the information technology staff contacted Virginia Commonwealth University to explore a computational approach to generating schedules. Together, we developed a tool that uses integer programming to generate schedules; we used the method to generate the 2011 schedule, resulting in savings of up to 150 hours of work annually. The schedule satisfies all the constraints required by the court by properly distributing panel sessions among its districts throughout the year. The court places great importance on its members not becoming parochial; to that end, judges sit in disparate panels to hear litigants, who convene in regions throughout the state, to ensure a more uniform application of the law. The court used industrial-strength integer programming software to generate the 2011 schedule at low cost by using resources available on the Cloud.

Key words: scheduling; integer programming applications; court system; cloud computing.
History: This paper was refereed. Published online in *Articles in Advance*.

Each May, a deputy clerk of the Court of Appeals of Virginia pulls down a wall-sized 12-month calendar with color-coded magnets (see Figure 1). Through mid-August, she spends approximately 150 hours, in addition to her other full-time responsibilities, moving the magnets around in search of a schedule of sessions that meets the case demand, satisfies the court's rules, and accommodates as many requests for avoid dates from the judges as possible. The schedule must satisfy many constraints on the timing and location of sessions and workload considerations for judges. Additionally, each judge must sit with every other judge on a three-judge panel session at least once during the year. The scheduling process involves weeks of constructing near-perfect schedules, discussing the pros and cons with the clerk of the court, and then going back to the magnetic calendar board until a schedule that can be presented to the chief judge is generated. A single change deemed necessary by the clerk or the chief judge can send a cascading effect through the rest of the schedule; simply swapping judges or panels will fail to produce a working schedule. The process of placing the magnets begins seemingly anew, with only the knowledge that the previous arrangements will not work. Thus, scheduling panel sessions and judges is a challenging feasibility problem.

Going forward, the manual scheduling process can be abandoned in favor of a new integer programming (IP)-based method. The method was created by a collaboration between the staff of the Court of Appeals of Virginia, the information technology department of the Supreme Court of Virginia, and the Department of Statistical Sciences and Operations Research at Virginia Commonwealth University (VCU). The deputy clerk is now able to generate a schedule satisfying all the rules of the court, or know that some rules conflict, in less than one day. The time that the deputy clerk dedicates to scheduling is dramatically reduced. The workload is shifted to the Cloud, where shared computing resources on the Internet are used to solve the optimization instances, freeing time for

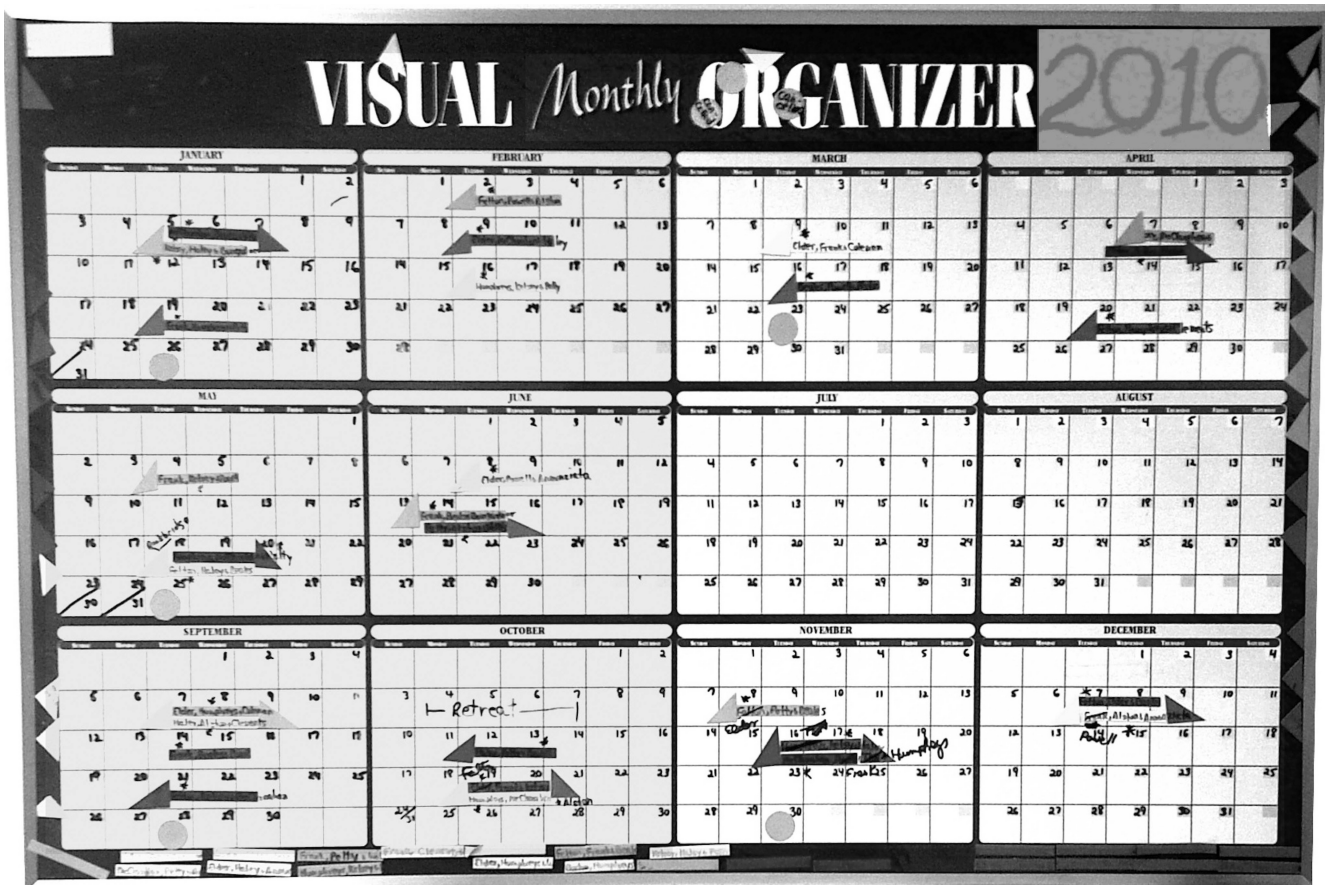

Figure 1: This graphic shows a magnetic calendar board for scheduling panel sessions and judges for the Court of Appeals of Virginia. Each color-coded magnet represents a panel session to which three judges are assigned. The color of a magnet indicates the district in which the session occurs.

the deputy clerk, clerk, and chief judge to perform other court responsibilities.

In court systems throughout the country, the process of scheduling panel sessions, assigning judges to panel sessions, and assigning cases to panel sessions is time consuming because of the care needed to ensure that the scheduling process does not affect the administration of justice. This paper presents the IP-based method we developed for scheduling sessions and assigning judges to sessions. The method was used to generate the 2011 schedule for the Court of Appeals of Virginia, and resulted in a savings of approximately 150 hours of work. The use of the computational resources on the Cloud saved the court approximately $10,000 that it would otherwise have had to spend on computer hardware and a commercial license for optimization software. The schedule satisfies all the rules of the court, which are designed to ensure a uniform application of the law.

Review of Previous Related Work

The court scheduling problem (CSP) discussed in this paper shares similarities with staff-scheduling and sports- and event-scheduling problems, which are often solved by using IP; however, the CSP has some important differences. It requires simultaneous satisfaction of constraints on time and geography for both staff (judges) and events (sessions).

Staff scheduling is applied in diverse areas including nurse scheduling (Burke et al. 2004), airline fleet scheduling, crew scheduling (Gao et al. 2009), and tennis umpire crew scheduling (Farmer et al.

2007). According to the classification system for staff scheduling and rostering presented by Ernst et al. (2004), the CSP is based on (1) task-based demand, in which a well-defined set of tasks must be completed; (2) days-off scheduling, which defines minimum rest requirements between tasks; and (3) line-of-work construction, in which schedules for individuals must satisfy specific constraints. The tasks for the court are the sessions that must be held so that all cases are heard. The CSP departs from this framework in that each judge must sit with every other judge at least once in a three-judge panel; this is similar to a round-robin tournament schedule.

Rasmussen and Trick (2008) survey previous work in the area of round-robin tournament sports scheduling, and identify classes of side constraints that are often enforced. These side constraints include location constraints and avoid-date constraints, which are also applicable to the CSP. The differences between the CSP and tournament scheduling include the following: (1) court sessions include either three or at least eight judges (when sitting *en banc*), whereas tournament games consist of two teams; (2) multiple judges can be from the same home district, whereas tournaments assume a one-to-one correspondence between teams and home locations; and (3) a session may occur in a district even if no judge from that district is participating, whereas a tournament usually specifies a home team and a visiting team for each match.

The CSP shares similarities with the scheduling of academic exams (Carter et al. 1994), classes (Miranda 2010), and dance showcases (Lejeune and Yakova 2008). As with the CSP, the academic scheduling problems require that events (courses and exams) occur with no conflicts and favor solutions with adequate spacing between events. Lejeune and Yakova (2008) describe the scheduling of dance showcases as a process of the simultaneous scheduling of heats and assignment of teachers and students to heats, which mirror the scheduling of sessions and assignment of judges to sessions in the CSP.

Scheduling and Justice

An analyst at the Supreme Court of Virginia initiated communication with faculty at VCU to discuss the problem of scheduling sessions for the Court of Appeals of Virginia. He presented a six-page document detailing the constraints that a schedule must satisfy and asked, "Is this the kind of thing that you guys do?" The constraints, which are rigid and deterministic, were in an itemized list, cataloged by type and priority. A quick glance at the document revealed that the language of law can be remarkably similar to the language of optimization.

What is the basis for court scheduling rules that prescribe an optimization-based approach? Law experts claim that implementing strict rules and removing judicial discretion from their scheduling process increases the likelihood of justice, fairness, and efficiency in the administration of court decisions (Brown and Lee 2000a, Samaha 2009). Norwood (1996) advocates restrictions on the ability of litigants to choose the judges on their panels, and equates such practices to jury shopping and law shopping, both of which undermine fundamental principles in our system of law. Samaha (2009) asserts that the assignment of cases to panels of judges or judges to panel sessions should be random in that an assignment should not be made with consideration of the particular experience or personal biases of judges. Brown and Lee (2000a) describe how panel packing during the civil rights era facilitated integration efforts in the US Court of Appeals for the Fifth Circuit. The practices now used in scheduling sessions and cases, with similarities to experimental design and random assignment, have facilitated empirical studies. Kastellec (2011) demonstrates that judges from different political parties affect each others' decision making. Samaha (2009) notes that judges denounce the use of randomness in other areas of applying justice. Imagine a judge that decides that both sides of a case have equal merit, and then flips a coin to determine the winner. Judges are charged with administering justice that is tailored to the particular details of the case at hand. Most courts therefore strive to assign judges to panels and cases to panels in such a way that (1) a case is equally likely to have any combination of judges on its panel, (2) the assignment process is not subject to exploitation by litigants, and (3) the court can operate efficiently and fairly with respect to the judges' schedules.

In the appendix to Brown and Lee (2000a, b), they review the scheduling practices of each circuit of the

US Court of Appeals. The circuits differ in the number of panel sessions per year, the number of appointed judges, and the number of panel sessions for each judge in a year. Many circuits separate the processes of scheduling sessions and assigning cases to sessions. Scheduling panels has the feel of a designed experiment. Some circuits create a matrix of every possible three-judge combination and sample without replacement from the rows of the matrix. Some circuits have computer programs that assist in creating schedules, which they then manually manipulate. Schedulers try to satisfy a number of constraints, including enforcing lower bounds on intervals between panel sessions for judges, upper bounds on consecutive months with sessions for judges, and geographic considerations, and also ensuring that each judge sits with every other judge in a three-judge panel session. The assignment of cases to panels usually occurs after the schedule of panels has been fixed and is conducted based on the output of a random process.

The scheduling of panel sessions is generally regarded as a housekeeping measure and not subject to law (Brown and Lee 2000a). Samaha (2009) suggests that other methods for scheduling panel sessions and cases are possible, including using politics, markets, a first-come first-served system, and scheduling based on the expertise of individual judges. The prevailing sentiment seems to be that the best approach for assigning judges to panels, in terms of justice, fairness, and efficiency, is to approximate an experimental design that controls for each attribute of judges and panels.

Scheduling Judges and Sessions for the Court of Appeals of Virginia

The constraints associated with scheduling sessions for the Court of Appeals of Virginia fall into roughly two categories: the scheduling of sessions and the assignment of judges to sessions. The number of sessions scheduled for this court is based on a forecast of the number of cases. For the past few years, the court has scheduled 29 three-judge panel sessions and five full-court (*en banc*) sessions, which usually consist of 11 members of the court, but must by statute consist of no fewer than eight judges. No panel sessions may be scheduled during the week of a full-court session. One of the 29 panel sessions is designated as a last panel—a panel that has an open spot to be filled by a member of the court. The chief judge is assigned to this panel as a placeholder for the open spot, which will be assigned later. Panel sessions are held in each of four districts, and constraints are implemented to allow at least three weeks between sessions in a district. Full-court sessions are required to be at least 45 days apart. At most, one panel session can be held during the weeks of sessions of the Supreme Court of Virginia and must be located in District 2. If two sessions are held in the same week, then one must be held in District 2. Each district must have a session in September, which follows two months in which no panel sessions are scheduled. The weeks of conferences, retreats, and certain holidays must also be avoided.

The assignment of judges to panels is as complex as is scheduling the panels. Each judge must sit with every other judge on a panel session at least once. Any two judges may sit together on a three-judge panel at most three times. There must be at least three weeks between panel sessions for each judge. A judge may not have panel sessions for more than three consecutive months. Each judge must have a panel session in each district and at least two sessions in his or her home district. Each judge can have at most two sessions in a district other than his (her) home district. Each full-time judge is assigned seven panel sessions. Each judge has three months of the year, including July and August, during which no panel sessions are scheduled. Part-time judges are senior judges who sit on one panel session in each half of the year. One part-time judge can at most be assigned to a panel session. Judges may also submit avoid-date requests.

Hearing the oral arguments of litigants, which is the function of the court sessions, is one of the most important aspects of the work of the Court of Appeals. The sessions occupy a limited portion of the judges' time annually. Extensive preparation for each case is required before sessions, because each case reflects a significant interpretation of state law. Judges are also continuously involved in activities to study case history and new laws. Creating schedules that maximize the court's utilization of judicial resources is critical so that members of the court can address

the many other responsibilities with which they are tasked in a given year.

Our initial model development focused on recreating the 2009 schedule in an automated fashion using mathematical programming. I led a team of VCU graduate students through the modeling process as part of a class project. Based on my previous experience with a nurse-scheduling problem, we began by defining a set of judges J and sessions S and using double-indexed binary variables of the form given in Equation (1). With this definition, we would build a set of judges J with properties such as home district, avoid-date requests, and full-time versus part-time status. We also maintain a set of sessions S with properties such as district, month, and week.

At an initial meeting with the staffs of the Supreme Court of Virginia and the Court of Appeals of Virginia, faculty and students from VCU gave a presentation on the flexibility and limitations of modeling using mathematical programming. The VCU team promised a solution to the clients that would generate at least 75 percent of a schedule; the remainder could be generated manually. However, all the attendees knew that 75 percent of a schedule with the possibility of small manual fixes cannot be generated. A small constraint violation in a proposed schedule could lead to a cascading array of changes that might require a total reworking. Because of the requirement of strict adherence to the rules of the court, the major challenge in generating schedules is finding a feasible solution. The nature of the problem does not lend itself to heuristic solution methods, which can be useful in helping one to choose from among a large number of feasible solutions.

After developing an initial model, we began implementation using a freely available optimization solver and modeling language. The instances, which had more than 80,000 variables and millions of constraints, quickly became too big for our hardware and software to handle. Therefore, we began implementing the model using the Gurobi Optimizer (http://www.gurobi.com) with the C callable library. After implementing a few constraints, we realized that the number of potential sessions was dramatically overestimated. With a simple change to S, the set of sessions, we reduced the number of potential sessions by a factor of 30. The number of variables decreased to approximately 10,000 and the number of constraints to about 100,000. Although we still had a few more constraints to model and implement, we could feel that a computable solution was imminent.

In our first attempt to model the constraint, "each judge must have three weeks between panel sessions," we had a constraint for each judge j and pair of sessions s_1 and s_2 separated by less than three weeks. Using the concept of clique inequalities in the conflict graph (Atamtürk et al. 2000, Padberg 1973), these constraints can be strengthened to the form of Equation (18) so that there are constraints only for each judge and week combination.

Making similar adjustments to the remainder of the model yielded a reduction in the number of constraints to just under 20,000. With all constraints implemented, and with the parameters set to aggressively generate cutting planes, the Gurobi Optimizer took about 10 hours to find a feasible solution to the 2009 scheduling problem. We created a Microsoft Access database for visually checking solutions and formatting schedules. Although the program was able to implement all the constraints listed in the initial document from the court, additional constraints were needed to ensure proper spacing between sessions and limits on the number of sessions for judges in districts other than their home districts. In July and August 2010, we added these additional constraints and some constraints specific to the 2011 calendar; the program successfully generated the 2011 schedule in early August 2010.

Implementation of the Court Scheduling Process

Figure 2 shows the steps for the court-scheduling system. The initial and final phases of formatting data are conducted with the help of Python scripts, Microsoft Access, and Microsoft Excel. The intermediate steps, which require industry-standard optimization software, are conducted by accessing the Gurobi Optimizer on the Amazon elastic computing platform (EC2), which we refer to as "the Cloud."

The first step in the new scheduling process involves collecting data. The clerk and a deputy clerk forecast the number of cases in each district based on the previous year's caseloads and determine the

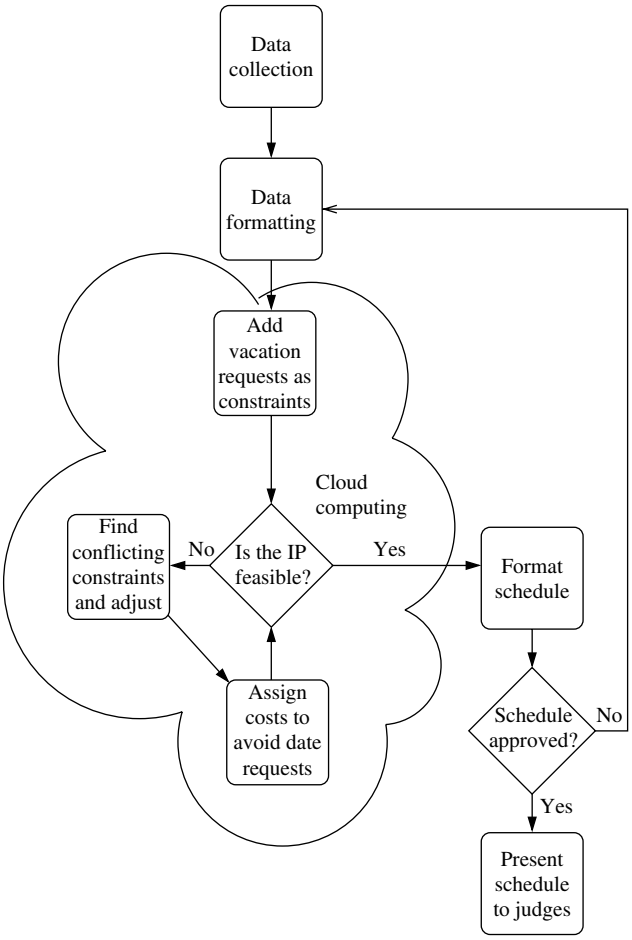

Figure 2: The flowchart shows the steps in the process of using optimization to generate a schedule for the Court of Appeals of Virginia.

number of sessions that must be scheduled for each district. They consult the calendar to determine eligible weeks during which sessions may be scheduled. Holiday weeks include the week of Thanksgiving, the week of Christmas, and the week before Christmas. The court does not schedule sessions during the months of July and August or during the week of Election Day. During five weeks scattered throughout the year, the judges attend various conferences and business meetings; these weeks must also be avoided. The sessions of the Supreme Court of Virginia are noted. The deputy clerk collects requests for individual avoid dates and preferred months in which no panels are scheduled from each judge. Each judge's home district is noted.

The data are then formatted for input to a program written in C, which uses the Gurobi Optimizer callable library. We access Gurobi using the EC2 platform to avoid the cost and maintenance of a local single-machine license. A single-machine license for software such as Gurobi plus hardware would cost approximately $10,000. After registering an EC2 account, the user is able to log in to a virtual machine with access to the Gurobi libraries and compiler programs. The fee for using this service is less than $10 per hour, plus some minor fees for transferring data. The C code and input files are transferred to the virtual machine, the code is compiled, and the program is run. The week, district, and judges for each session in the optimal schedule are written to a text file, which is downloaded to a local computer.

The appendix contains the core IP model. To generate the 2011 schedule, we began by strictly enforcing all avoid-date requests and preferred months with no panels from judges. These constraints are enforced by fixing the appropriate variables to zero. The optimization solver determined that the problem was infeasible. We used the Gurobi irreducible infeasible subsystem (IIS) finder to determine a minimum-sized set of conflicting constraints (Guieu and Chinneck 1999). The results indicated that the following constraints were in conflict: (1) each district must have a session in September, (2) at most one panel session (in District 2) can be held during the week of September 11 because of a session of the Supreme Court of Virginia, (3) no sessions can be scheduled during a judicial conference during the week of September 25, and (4) if two panel sessions are held in the same week, one must be held in District 2. Upon the approval of the clerk and deputy clerk, we allowed two panel sessions during the same week in September without the requirement that one is in District 2.

With the modified constraints, the problem remained infeasible. The IIS finder indicated that the infeasibility was because of the inability to satisfy multiple judges' requests for no panels in the month of June. We then placed a penalty on avoid dates and preferred months with no panels by adding a cost coefficient for appropriate variables in the objective function. We used a cost of 1 for avoid dates and a cost of 100 for preferred months with no panels. We were able to obtain a feasible solution to this

problem after approximately 11 hours with objective value 101 because of one violation of a request for a preferred month with no panels and one violation of an avoid-date request. The problem was not solved to provable optimality after an additional seven hours of computation.

A Microsoft Access database formats the schedules and facilitates checking that constraints are satisfied. The database aids in organizing input data to the CSP, and enables evaluating and printing schedules. Two options are given for formatting a schedule. The first option lists each session in chronological order, with the location and participating judges. The second option provides a list of the weeks and information about each week's sessions. Other views are included in the database to help verify that specific constraints are satisfied.

The deputy clerk reviews the schedule using the Microsoft Access application. If necessary, constraints are added or removed. If the deputy clerk approves the schedule, she reviews it with the clerk. If the clerk approves the schedule, then it is presented to the chief judge for final approval.

Future Directions

The court-scheduling system described here saves the deputy clerk nearly 150 hours per year and transfers much of the hard work in creating a schedule to the Cloud. Using computer-generated schedules reduces human discretion in the process, increasing the likelihood of disparate panels and thus ensuring a more uniform application of the law. Although streamlining the data-formatting process can be further improved, a one-click application for generating schedules is unlikely. Each calendar year brings new challenges because of the way that dates are arranged and because of new avoid-date requests from judges. Even in this highly constrained and seemingly deterministic environment, the scheduling process is necessarily a feedback loop (see Figure 2), as in any modeling exercise.

Improvements are possible in developing solution methods for the IP instances. Tailored cutting planes and branching schemes may help generate solutions and prove optimality more quickly. Techniques from constraint programming may also help to intelligently enumerate aspects of feasible schedules.

Appendix

Let J be the set of judges, $J_F \subseteq J$ the set of full-time judges including the chief judge, and $J_P \subseteq J$ the set of part-time judges. Let $c \in J_F$ indicate the chief judge. Let D be the set of districts, and n_d the number of scheduled sessions in each district. For each Judge j, h_j is the home district. Let S be the set of potential sessions, $S_P \subseteq S$ the potential panel sessions (except for the potential last-panel sessions), S_F the full-court sessions, and $S_l \subseteq S$ the potential last-panel sessions. Let $S_d \subseteq S$ be the panel sessions in district d. Let H be the set of sessions of the Supreme Court of Virginia (high court). For each session $s \in S \cup H$, w_s, m_s, and d_s indicate the week, month, and district. Let W and M be the set of weeks and months of potential sessions. Let a_{js} be the cost of having Judge j sit on session s, which is zero unless session s falls during one of Judge j's requested avoid weeks. Similarly, let b_{jk} be the cost of having Judge j work during month k, which is zero unless Judge j has requested to avoid sessions in month k.

The integer program contains four sets of binary variables. Let

$$x_{js} = \begin{cases} 1 & \text{if Judge } j \text{ is assigned to session } s, \\ 0 & \text{otherwise,} \end{cases} \quad \text{for } j \in J \text{ and } s \in S, \tag{1}$$

$$y_s = \begin{cases} 1 & \text{if session} s \text{ is held,} \\ 0 & \text{otherwise,} \end{cases} \quad \text{for } s \in S, \tag{2}$$

$$z_{jk} = \begin{cases} 1 & \text{if Judge } j \text{ works in month } k, \\ 0 & \text{otherwise,} \end{cases} \quad \text{for } j \in J \text{ and } k \in M, \text{ and} \tag{3}$$

$$g_{ils} = \begin{cases} 1 & \text{if Judge } i \text{ works with Judge } \ell \text{ on session } s, \\ 0 & \text{otherwise,} \end{cases} \quad \text{for } i \in J_F \text{ and } \ell \in J_F,\ i \neq \ell, \text{ and } s \in S. \tag{4}$$

The IP model is then

$$\min \sum_{j \in J} \sum_{s \in S} a_{js} X_{js} + \sum_{j \in J} \sum_{k \in M} b_{jk} z_{jk},$$

subject to

$$\sum_{s\in S_P} y_s = 28, \tag{5}$$

$$\sum_{s\in S_F} y_s = 5, \tag{6}$$

$$\sum_{s\in S_l} y_s = 1, \tag{7}$$

$$\sum_{s\in S_d} y_s = n_d, \quad d\in D, \tag{8}$$

$$\sum_{s\in S_F:\, k\le w_s\le k+5} y_s \le 1, \quad k\in\{1,2,\ldots,|W|-5\}, \tag{9}$$

$$\sum_{s\in S_P\cup S_F:\, w_s=w_k} y_s \le 1, \quad k\in H, \tag{10}$$

$$\sum_{s\in S:\, d_s=d,\, k\le w_s\le k+2} y_s \le 1, \quad d\in D,\, k\in\{1,2,\ldots,|W|-2\}, \tag{11}$$

$$\sum_{s\in S_P:\, m_s=9,\, d_s=d} y_s \ge 1, \quad d\in D, \tag{12}$$

$$y_s + \sum_{k\in S_F:\, w_k=w_s} y_k \le 1, \quad s\in S_P\cup S_l, \tag{13}$$

$$\sum_{s\in S_P\cup S_l:\, w_s=k \text{ and } d_s\neq 2} y_s \le 1, \quad k\in W, \tag{14}$$

$$\sum_{j\in J} x_{js} = 3y_s, \quad s\in S_P\cup S_l, \tag{15}$$

$$\sum_{j\in J_F} x_{js} \ge 8y_s, \quad s\in S_F, \tag{16}$$

$$\sum_{j\in J_P} x_{js} \le 1, \quad s\in S, \tag{17}$$

$$\sum_{s\in S:\, k\le w_s\le k+2} x_{js} \le 1, \quad j\in J,\, k\in\{1,2,\ldots,|W|-2\}, \tag{18}$$

$$\sum_{i=k}^{k+3} z_{ji} \le 3, \quad k\in\{1,\ldots,|M|-3\}, \tag{19}$$

$$\sum_{k\in M} (1-z_{jk}) \ge 2, \quad j\in J_F, \tag{20}$$

$$\sum_{s\in S_P\cup S_l:\, m_s\le 6} x_{js} = 1, \quad j\in J_P, \tag{21}$$

$$\sum_{s\in S_P\cup S_l:\, m_s\ge 9} x_{js} = 1, \quad j\in J_P, \tag{22}$$

$$\sum_{s\in S_P\cup S_l:\, d_s=d} x_{js} \ge 1, \quad j\in J_F,\, d\in D:\, h_j\neq d, \tag{23}$$

$$\sum_{s\in S_P\cup S_l:\, d_s=d} x_{js} \le 2, \quad j\in J_F,\, d\in D:\, h_j\neq d, \tag{24}$$

$$\sum_{s\in S_P\cup S_l:\, d_s=d} x_{js} \ge 2, \quad j\in J_F,\, d\in D:\, h_j=d, \tag{25}$$

$$x_{is} + x_{\ell s} \ge 2g_{i\ell s}, \quad s\in S_P\cup S_l,\, i,\ell\in J_F \tag{26}$$

$$x_{is} + x_{\ell s} \le 1 + g_{i\ell s}, \quad s\in S_P\cup S_l,\, i,\ell\in J_F \tag{27}$$

$$\sum_{s\in S_P\cup S_l} g_{i\ell s} \ge 1, \quad i,\ell\in J_F \tag{28}$$

$$\sum_{s\in S_P\cup S_l} g_{i\ell s} \le 3, \quad i,\ell\in J_F \tag{29}$$

$$\sum_{s\in S_P\cup S_\ell} x_{js} = 7, \quad j\in J_F, \tag{30}$$

$$\sum_{s\in S_\ell} x_{cs} = 1, \tag{31}$$

$$x_{js} \le y_s, \quad j\in J_F,\, s\in S, \text{ or } j\in J_p,\, s\in S/S_\ell, \tag{32}$$

$$x_{js} \le z_{jk}, \quad j\in J_F,\, S\in S_P\cup S_\ell,\, k\in M, \tag{33}$$

$$\sum_{s\in S_P\cup S_l:\, m_s=k} x_{js} \ge z_{jk}, \quad j\in J_F,\, k\in M. \tag{34}$$

The objective presented here minimizes the cost of violating avoid-date requests. We use penalties of 100 for violating a request for a preferred month with no panel sessions and a cost of 1 for violating a request for a particular date with no panel sessions. As described in the text, we can express avoid-date requests as constraints by fixing appropriate variables to zero, and seek a feasible solution to the integer program with a zero for the objective function. Constraints (5)–(14) are concerned with scheduling sessions. Constraints (5)–(7) require that 28 panel sessions, five full-court sessions, and one last-panel session are scheduled. Constraint (8) dictates the number of sessions in each district. Constraint (9) ensures that full-court sessions are at least 45 days (6 weeks) apart. Constraint (10) limits the number of panel sessions during weeks of high court sessions to one. Constraint (11) ensures that each district has at least three weeks between panel sessions, and Constraint (12) ensures that each district has a session in the first month after the summer break. Constraint (13) disallows panel sessions to be scheduled during the week of full-court sessions. Constraint (14) limits the number of panel sessions per week to two and requires that if two sessions are scheduled, then one must be in District 2.

Constraints (15)–(31) are concerned with the assignment of judges to sessions. Constraints (15) and (16) require that three judges sit on each panel session and at least eight full-time judges sit on full-court sessions. Constraint (17) limits the number of part-time judges on each panel session to one. Constraint (18) ensures that each judge has three weeks between panel sessions, and Constraint (19) ensures that judges cannot have panel sessions in three consecutive months. Constraint (20) ensures that each full-time judge has three months without panel sessions during the year. Constraints (21) and (22) require that each part-time judge sits on one panel session in the first six months of the year, and one panel session in the last four months. Constraints (23)–(25) require that each full-time judge has at least one panel session in each district, at least two panel sessions in his (her) home district, and at most two panel sessions in each non-home district. Constraints (26)–(29) ensure that any two full-time judges sit together on a three-judge panel session at least once and at most three times. Constraint (30) requires that each full-time judge sits on seven panel sessions. Constraint (31) ensures that the chief judge is assigned to sit on the designated last-panel session, which serves as a placeholder for a judge to be assigned nearer to the date of the last panel.

Constraints (32)–(34) ensure that proper relationships between the binary variables are maintained. Constraint (32) requires that if a judge is assigned to a session, then that session must be held. Constraints (33) and (34) require that a judge is assigned to a session if and only if the judge is scheduled to work in the month of that session.

The number of variables in the model is $|J||S| + |S| + |J||M| + \binom{|J_F|}{2}|S|$. The number of constraints is quadratic in the number of full-time judges and increases as $2(|S_P| + |S_\ell|)|J_F|^2$. The general complexity of the problem is most affected by the number of judges and the number of possible sessions. The problem size is largely determined by the variables and constraints associated with ensuring that each pair of judges sits together on a three-judge panel at least once; namely, the variables $g_{i\ell s}$ indicating whether judge i sits with judge ℓ in session s and Constraints (26)–(29). These constraints are similar in nature to the problem of finding an edge coloring of a graph (West 2001) that is known to be NP-hard (Garey and Johnson 1979). The relationship between graph coloring and scheduling is well studied, and adding side constraints usually adds complexity to the problem. For example, Easton et al. (2003) investigate the traveling tournament problem, which has elements of graph coloring, and the traveling salesperson problem (TSP). Their experience has been that the traveling tournament problem becomes intractable for smaller instances than either graph coloring or the TSP. Constraints (9)–(11), (13) and (14) are the constraints of a set-packing problem in the variables y_s, which is also known to be NP-hard (Garey and Johnson 1979).

Acknowledgments

The author is grateful to the employees of the Court of Appeals of Virginia and the Supreme Court of Virginia for their eagerness to develop efficient systems. From the Court of Appeals of Virginia: the Honorable Walter S. Felton Jr., Chief Judge; Cynthia L. McCoy, Clerk; Deborah A. F. Uitvlucht, Deputy Clerk. From the Supreme Court of Virginia: Jerry Berman, Appellate Information Technology Team Leader; Delois Jeffers, Contract Business Analyst; Artie O'Connor, Contract Programming Analyst; Ben Grannan, Analyst. The author thanks the members of the Spring 2010 OPER 639 Practical Optimization class at Virginia Commonwealth University for initial model development and creation of the Microsoft Access database application.

References

Atamtürk, A., G. L. Nemhauser, M. W. P. Savelsbergh. 2000. Conflict graphs in solving integer programming problems. *Eur. J. Oper. Res.* **121**(1) 40–55.

Brown, J., Jr., A. H. Lee. 2000a. Neutral assignment of judges at the Court of Appeals. *Texas Law Rev.* **78**(5) 1037–1116.

Brown, J., Jr., A. H. Lee. 2000b. Circuit practices. Accessed July 21, 2011, http://law.du.edu/images/uploads/neutral-assignment/Neutral_assignment_links.pdf.

Burke, E. K., P. de Causmaecker, G. vanden Berghe, H. van Landghem. 2004. The state of the art of nurse rostering. *J. Scheduling* **7**(6) 441–499.

Carter, M. W., G. Laporte, J. W. Chinneck. 1994. A general examination scheduling system. *Interfaces* **24**(3) 109–120.

Easton, K., G. Nemhauser, M. Trick. 2003. Solving the travelling tournament problem: A combined integer programming and constraint programming approach. *Practice and Theory of Automated Timetabling IV*. Lecture Notes in Computer Science, Vol. 2740. Springer-Verlag, Berlin, 100–109.

Ernst, A. T., H. Jiang, M. Krishnamoorthy, D. Sier. 2004. Staff scheduling and rostering: A review of applications, methods and models. *Eur. J. Oper. Res.* **153**(1) 3–27.

Farmer, A., J. S. Smith, L. T. Miller. 2007. Scheduling umpire crews for professional tennis tournaments. *Interfaces* **37**(2) 187–196.

Gao, C., E. L. Johnson, B. C. Smith. 2009. Integrated airline fleet and crew robust planning. *Transportation Sci.* **43**(1) 2–16.

Garey, M. R., D. S. Johnson. 1979. *Computers and Intractability.* Freeman, New York.

Guieu, O., J. W. Chinneck. 1999. Analyzing infeasible mixed-integer and integer linear programs. *INFORMS J. Comput.* **11**(1) 63–77.

Kastellec, J. P. 2011. Hierarchical and collegial politics on the U.S. Courts of Appeals. *J. Politics* **73**(2) 345–361.

Lejeune, M. A., N. Yakova. 2008. Showcase scheduling at Fred Astaire East Side Dance Studio. *Interfaces* **38**(3) 176–186.

Miranda, J. 2010. eClasSkeduler: A course scheduling system for the executive education unit at the Universidad de Chile. *Interfaces* **40**(3) 196–207.

Norwood, K. J. 1996. Shopping for a venue: The need for more limits on choice. *Univ. Miami Law Rev.* **50**(2) 267–334.

Padberg, M. 1973. On the facial structure of set packing polyhedra. *Math. Programming* **5**(1) 199–215.

Rasmussen, R. V., M. A. Trick. 2008. Round robin scheduling—A survey. *Eur. J. Oper. Res.* **188**(3) 617–636.

Samaha, A. M. 2009. Randomization in adjudication. *William and Mary Law Rev.* **51**(1) 1–86.

West, D. B. 2001. *Introduction to Graph Theory*. Pearson Education, Urbana, IL.

Cynthia L. McCoy, Clerk of Court, Court of Appeals of Virginia, Richmond, Virginia, writes: "I am writing, as Clerk of the Court of Appeals of Virginia, with regard to the scheduling program developed for the Court, as the result of a collaborative effort of Dr. J. Paul Brooks of Virginia Commonwealth University (VCU), his Spring 2010 Practical Optimization class, the Application Development Division of the Supreme Court of Virginia, and the Clerk's Office of the Court of Appeals.

"The challenge presented was one that had been discussed by members of the Clerk's Office and the appellate team of the application development division for many years. Every year, when it came time to prepare the Court's annual schedule, a task that could take up to 150 hours of labor-intensive work each year, the question would reemerge of whether it would be possible to utilize the efficiency and vast calculating resources of a computer environment to perform the time-consuming, up-front calculations that were necessary in order to create a workable schedule for the Court. An added challenge was that, due to budgetary constraints, there were no funds available for this project. Thus, it was imperative to find a creative way to bring together and utilize existing state resources.

"The broad framework of the Court of Appeals' annual schedule is that the Court hears its appellate cases during 29 docket sessions, each of which consists of a panel of three judges, convening in four regions throughout the Commonwealth of Virginia. In addition, the full Court (eleven judges) convenes five times a year. However, it is the many specific requirements involved in creating a schedule that cause the greatest complications when manually attempting to arrive at an acceptable solution. When members of the clerk's staff and the appellate team of the application development division met to identify the most important factors, constants, and variables that were taken into consideration in setting the Court's schedule each year and, thus, needed to be incorporated into any computer program, the resulting document was a six-page single-spaced list of specific requirements.

"Over the years, we had become convinced that an "off-the-shelf" program that would encompass all that we were asking for did not exist. Nonetheless, in 2009 we decided to pursue the goal of having such a program created, as it was becoming increasingly evident to all involved that the current way of producing the annual schedule was an inefficient method that committed a very large portion of manpower hours and resources each year. Fortuitously, Ben Grannan, a contract analyst for the Supreme Court of Virginia, became aware of what we were seeking and arranged to put us in touch with Dr. Brooks, who was teaching a class in practical optimization at VCU. We approached Dr. Brooks with the idea of a collaboration of resources to determine whether a program could be developed to accomplish the creation of an annual schedule for an appellate court. He was eager to take on the challenge and enthusiastic to find a solution that would do the majority of the "heavy lifting" for us. From that initial contact, the project rapidly moved forward. For six months in 2010, the parties involved in this project were in constant communication with one another, as step by step the program was built and completed. Even after the class's semester was concluded, Dr. Brooks continued to devote multiple hours of his own time to the project.

"From my perspective as Clerk of Court, the program that resulted from this collaborative effort is a highly valuable tool to be used in the creation of the Court's annual schedule. I believe that the value of this program is not only that it is extremely efficient and saves countless hours of work, but that it produces absolutely neutral, objective scheduling solutions that, at the very least, provide a workable framework upon which to base the Court's annual schedule. In addition, this project brought together distinct state resources in a collegial problem-solving venture that realized budgetary benefits for the Commonwealth. I look forward to utilizing this program to develop future schedules for the Court."