Generating College Conference Basketball Schedules by a SAT Solver

Hantao Zhang*
Computer Science Department
The University of Iowa
Iowa City, IA 52242
hzhang@cs.uiowa.edu

May 20, 2003

1 Introduction

Major college conferences basketball matches draw millions of viewers each year. Besides team members care about the fairness of a schedule, there are also loyal fans and die-hard team followers who are upset with inconvenient shceduling, not to mention leaders of other sports, such as swimming and wresting, who have to schedule around basketball games. For instance, sportscasters might mention that that Ohio State had a tough schedule in 2002, because after playing five out of six conference games at home, they had to play four road games in a row. For a schedule to be fair, we may require that no teams play three consecutive home games or three consecutive road games. According to this criterion, several BigTen schools do not have a fair schedule in 2002 [10].

There are also other fairness criteria. The BigTen conference has eleven teams and every team plays 16 games in a season, comprised of eight home games and eight road games, for a total of 88 games. Since each game day can schedule at most five games (one team will be idle, called a *bye*), a season consists of at least 18 game days. In 2002, the BigTen basketball regular reason consists of nine weeks from January 2 to March 2. If we pick two days from each week, one week day and one weekend, then we have exactly 18 game days. Since weekend home games draw larger audiences, and weekday road homes may hurt players' study, for a schedule to be fair, we may hope that each team plays exactly the same number of home games on weekends and on weekdays and the same number of road games on weekends and on weekdays. If we take a look at the actual BigTen regular season schedule [10], we can see that the schedule is far from fair. From Table 1, we can see that only Michigan and Purdue have an ideal schedule, playing the same number of games on both weekends and weekdays for both home and road games.

Because a BigTen regular season basketball schedule has a very tight constraint on the number of game days (it needs at least 18 game days, but is given only 18 game days), given various constraints, it is not a trivial matter to produce a fair schedule. The basketball schedule problem

*Partially supported by the National Science Foundation under Grant CCR-0098093.

Table 1: Statistics from the 2002 BigTen regular season basketball schedule (Nonconference games are not ignored; E = weekend games; D = weekday games)

	Home		Road			Home		Road			Home		Road	
Team	E	D	E	D	Team	E	D	E	D	Team	E	D	E	D
Illinois	3	5	4	4	Michigan St	4	4	4	4	Penn St	3	5	5	3
Indiana	3	5	5	3	Minnesota	5	3	4	4	Purdue	4	4	4	4
Iowa	5	3	4	4	Northwestern	3	5	4	4	Wisconsin	4	4	3	5
Michigan	6	2	2	6	Ohio State	4	4	5	3					

can be easily stated as a constraint satisfaction problem. Even if we have the most advanced computer technology at hand, if we are not careful on how to specify the problem properly, the solution can still be intractable. In this paper, we will show how to produce a fair schedule with the assistance of a propositional solver, when we formulate the problem smartly.

2 The Schedule Problem as Constraint Satisfaction

The BigTen conference basketball adopts a partial double round robin schedule where each team plays each other team at least once and at most twice. Because each team has ten opponents and plays only 16 games in a season, each team plays six teams twice (one home and one road game) and four teams once (two at home and two on road).

To formulate the schedule problem as a constraint satisfaction problem, let us denote the teams by 1 through 11, and number the game days 1 through 18, and define a set of 0/1 variables $p_{x,y,z}$, where $1 \leq x \leq 11, 1 \leq y \leq 11$, and $1 \leq z \leq 18$: $p_{x,y,z} = 1$ if and only if x plays a home game against y in game day z.

Various constraints can be expressed formally using the variables $p_{x,y,z}$.

1. Each team can play at most once in one game day: For $1 \leq x \leq 11$, $1 \leq z \leq 18$,

$$\sum_{y=1}^{11} (p_{x,y,z} + p_{y,x,z}) \leq 1.$$

2. Each team must play each other team at least once and no more than twice: For $1 \leq x, y \leq 11$, $x \neq y$,

$$\sum_{z=1}^{18} p_{x,x,z} = 0, \quad \sum_{z=1}^{18} p_{x,y,z} \leq 1, \quad \sum_{z=1}^{18} p_{y,x,z} \leq 1, \quad 1 \leq \sum_{z=1}^{18} (p_{x,y,z} + p_{y,x,z}) \leq 2;$$

3. Each team plays four home and four road games in weekends and in weekdays: For $1 \leq x \leq 11$,

$$\sum_{y=1}^{11}\sum_{i=1}^{9} p_{x,y,2i-1} = 4, \quad \sum_{y=1}^{11}\sum_{i=1}^{9} p_{x,y,2i} = 4, \quad \sum_{y=1}^{11}\sum_{i=1}^{9} p_{y,x,2i-1} = 4, \quad \sum_{y=1}^{11}\sum_{i=1}^{9} p_{y,x,2i} = 4;$$

4. Each game day will hold four or five games, and the total number of games is 88:

$$\forall 1 \le z \le 18, 4 \le \sum_{x=1}^{11}\sum_{y=1}^{11} p_{x,y,z} \le 5; \quad \text{and} \quad \sum_{x=1}^{11}\sum_{y=1}^{11}\sum_{z=1}^{18} p_{x,y,z} = 88;$$

5. No three home games in consecutive days and no three road games in consecutive days: For $1 \le x \le 11$, $1 \le z \le 16$,

$$\sum_{y=1}^{11}\sum_{i=0}^{2} p_{x,y,z+i} < 3, \qquad \sum_{y=1}^{11}\sum_{i=0}^{2} p_{y,x,z+i} < 3.$$

Theoretically, each of the above constraints can be converted into a set of propositional clauses, treating $p_{x,y,z}$ as propositional variables[1]. However, it is very expensive to do so for Constraint 4. To simplify the conversion, we may enforce that only days 9 and 10 hold four games; the rest days hold five games each. So Constraint 4 can be replaced by Constraint $4'$:

$4'$. For $1 \le z \le 18$, $z \ne 9, 10$,

$$\sum_{x=1}^{11}\sum_{y=1}^{11} p_{x,y,z} = 5;$$

and for $9 \le z \le 10$,

$$\sum_{x=1}^{11}\sum_{y=1}^{11} p_{x,y,z} = 4.$$

We are able to obtain a set of over 10 million propositional clauses (the number of propositional variables is 2178) from Constraints 1–3, $4'$, and 5. However, Sato [12, 13] cannot produce any solution from this set of clauses after two days of running. It appears that this approach leads to a dead end.

3 A Three-Phases Approach

Checking the literature, we found that a similar work has been done by Nemhauser and Trick [6] for the 1997/98 Atlantic Coast Conference (ACC) basketball schedule. The ACC has 9 teams and plays a double round-robin tournament for the regular season basketball schedule. Each team plays 16 games (8 home games and 8 road games), just like the BigTen conference. Nemhauser and Trick used a three-phases approach, namely pattern generation, pattern set generation, and timetable generation, to search for a round-robin schedule. This approach was proposed by Cain [1], and used by de Werra [9] and Schruder [7], among others. In each phase, various constraints from the ACC organizers are used to narrow down the search space. They reported a "turn-around-time" of 24 hours using integer programming and explicit enumeration, which means it

[1]Given a set S of n 0/1 variables, to specify $\sum_{x\in S} x = m, 1 \le m < n$, we may use the following propositional clauses: For any $X \subseteq S$, $|X| = 1 + n$, create the clause $\vee_{x\in X}\overline{x}$, where $\overline{x}$ is the negation of x; for any $Y \subseteq S$, $|Y| = n - m + 1$, create the clause $\vee_{y\in Y} y$. The first type of clauses ensures that no more than n variables are true (1); the second type of clauses ensures that no more than $n - m$ variables are false (0). The number of clauses is thus $C_n^{m+1} + C_n^{n-m+1}$, or equivalently, $C_n^{m+1} + C_n^{m-1}$.

takes one day of computing time from specifying/modifying the constraints (using feedback from the ACC organizers) to proposing new solutions. This "turn-around-time" has been dramatically reduced to one minute by Henz [3], who used the finite-domain constraint programming [5, 8].

In the following, we will present the three-phases approach based on Henz' presentation [3].

3.1 Phase One: Pattern Generation

A pattern indicates the way in which a team can play home, away (road), and bye throughout the season. In the ACC 1997/98 case, for example, the following pattern is acceptable, where a home game is represented by $+$, a road game by $-$, and a bye by b.

days	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
pattern	$-$	$+$	b	$-$	$+$	$+$	$-$	$+$	$-$	$-$	$+$	b	$+$	$-$	$-$	$+$	$+$	$-$

A suitable constraint programming specification for pattern generation consists of 0/1 variables $h_j, r_j, b_j, 1 \leq j \leq 18$. A team that plays according to the pattern represented by these variables plays home (road, bye) at day j if and only if $h_j = 1$ ($r_j = 1, b_j = 1$). The following constraints are used by Nemhauser and Trick [6] and Henz [3] for the 1997/98 ACC basketball schedule.

1. **Exclusiveness.** For all game days j: $h_j + r_j + b_j = 1$.
2. **Mirror.** The game days are grouped into pairs (d_1 d_2), such that each team will play against the same team in days d_1 and d_2. Such a grouping is called a mirroring scheme. The following mirror scheme is used for the 1997/98 ACC basketball season:

   $$m = \{(1\ 8), (2\ 9), (3\ 12), (4\ 13), (5\ 14), (6\ 15), (7\ 16), (10\ 17), (11\ 18)\}.$$

   That is, for $(i\ j) \in m$: $h_i = r_j, r_i = h_j, b_i = b_j$.
3. **No two final road games.** That is, $r_{17} + r_{18} < 2$.
4. **Home/Road/Bye Pattern.** No team may play three home games in a row or three road games in a row, that is, for all $j \leq 16$: $h_j + h_{j+1} + h_{j+2} < 3$, $r_j + r_{j+1} + r_{j+2} < 3$.
   No team may play four days without a home game, or five days without a road game, that is, for $j \leq 15$: $h_j + h_{j+1} + h_{j+2} + h_{j+3} > 0$; for $j \leq 14$: $r_j + r_{j+1} + r_{j+2} + r_{j+3} + r_{j+4} > 0$.
   No team may play the first three or the last three days without a home game, that is, $h_1 + h_2 + h_3 > 0$, $h_{16} + h_{17} + h_{18} > 0$.
5. **Weekend Pattern.** Of the weekends, each team plays four at home, four away, and one bye, that is, $(\sum_{j \in \{2,4,\ldots,18\}} h_j) = 4$, $(\sum_{j \in \{2,4,\ldots,18\}} r_j) = 4$, $(\sum_{j \in \{2,4,\ldots,18\}} b_j) = 1$.
   The same is true for the weekdays games: $(\sum_{j \in \{1,3,\ldots,17\}} h_j) = 4$, $(\sum_{j \in \{1,3,\ldots,17\}} r_j) = 4$, $(\sum_{j \in \{1,3,\ldots,17\}} b_j) = 1$.
6. **First Weekends.** Each team cannot have more than three road games in the first five weekends, that is, $(\sum_{j \in \{2,4,6,8,10\}} r_j) \leq 3$.
7. **Idiosyncratic criteria.** (see [3] for explanation) $b_1 + r_{18} < 2$, $b_1 + h_{17} < 2$, $b_{16} + r_{18} < 2$.

The above seven constraints can be easily transformed into a set of 1499 propositional clauses by considering h_j, r_j, b_j as propositional variables[2]. It takes less than 0.01 seconds for Sato [12] to generate all the 38 patterns. In comparison, Henz' Friar Tuck [4] took 0.44 seconds on a similar machine.

3.2 Phase Two: Pattern Set Generation

After generating all patterns that meet the given constraints, in Phase 2, sets of 9 patterns are selected for the ACC in such a way each game day satisfies certain constraints. Suppose the 38 feasible patterns are represented by three 38×9 matrices, h, r, b of 0/1 values, where $h_{i,j} = 1$ ($r_{i,j} = 1, b_{i,j} = 1$) if and only if the pattern i fixes day j to a home game (road game, bye). For each pattern i, a 0/1 variable x_i indicates whether this pattern occurs in the resulting pattern set. The constraints are given below.

1. **A set has 9 patterns.** $\sum_i x_i = 9$.
2. **Each day has four home games, four road games, and one bye.** For $1 \leq z \leq 18$: (a) $\sum_i h_{i,z} x_i = 4$, (b) $\sum_i r_{i,z} x_i = 4$, (c) $\sum_i b_{i,z} x_i = 1$.
3. **Two patterns in a set must be able to play a game.** For every pair i, i', if $h_{i,z} + r_{i',z} \neq 2$ for all z, then $x_i + x_{i'} < 1$.

While it is very expensive to convert Constraint 1 into a set of propositional clauses (ref. footnote 1), fortunately, Constraint 1 can be deduced from Constraint 2 (c), because $\sum_z \sum_i b_{i,z} x_i = 18$ so $\sum_i x_i = 9$. Thus, Constraint 1 can be safely discarded.

Constraints 2-3 produce a set of 569300 propositional clauses (over 38 variables). It takes 0.60 seconds for Sato to generate all the 17 pattern sets. Most clauses are come from Constraint 2 (a) and (b). If we leave Constraint 2 (a) and (b) out, then the number of propositional clauses is reduced to 140, and it takes only 0.02 seconds for Sato to complete the search (Sato found 3689 models and a post-check filters out the 17 pattern sets). In comparison, Henz' Friar Tuck [4] took 3.1 seconds for this task.

3.3 Phase Three: Timetable Generation

Suppose a pattern set is stored in a 9×18 matrix S: $S_{x,z} \in \{h, r, b\}$. In this phase, we need to assign 9 patterns to 9 teams to form a timetable. We use 0/1 variables $\pi_{x,j}$, $1 \leq x, j \leq 9$, to denote whether pattern j is assigned to team x. A timetable can be specified by the 0/1 variables $p_{x,y,z}$ introduced in Section 2: $p_{x,y,z} = 1$ if and only if x plays a home game against y in day z. Constraints on $\pi_{x,j}$ and $p_{x,y,z}$ can be easily specified:

1. π **is a permutation.** For all x, $\sum_j \pi_{x,j} = 1$; for all j, $\sum_x \pi_{x,j} = 1$.
2. **Each team can play at most once in one game day.** For $1 \leq x \leq 9$, $1 \leq z \leq 18$, $\sum_y (p_{x,y,z} + p_{y,x,z}) \leq 1$.

[2]The number of propositional varialbes can be reduced from 3×18 to 2×18 by eliminating variables b_j: $b_j = 1$ iff $h_j = 0$ and $r_j = 0$.

3. **Each team must play each other team at home and on road once.** For $1 \leq x, y \leq 9$, $x \neq y$, $\sum_z p_{x,x,z} = 0$, $\quad \sum_z p_{x,y,z} = 1$, $\quad \sum_z p_{y,x,z} = 1$.
4. **Mirror.** Let $m = \{(1\ 8), (2\ 9), (3\ 12), (4\ 13), (5\ 14), (6\ 15), (7\ 16), (10\ 17), (11\ 18)\}$. For $1 \leq x, y \leq 9$, for $(i\ j) \in m$, $p_{x,y,i} + p_{y,x,j} \neq 1$.
5. **Relationship between π and p.** For $1 \leq x, j \leq 9$, if $\pi_{x,j} = 1$, then for $1 \leq z \leq 18$, for $1 \leq y \leq 9$, if $S_{j,z} = h$, then $p_{y,x,z} = 0$; if $S_{j,z} = r$, then $p_{x,y,z} = 0$; if $S_{j,z} = b$, then $p_{x,y,z} = 0$ and $p_{y,x,z} = 0$.

All the above constraints can be easily converted into propositional clauses.

In [6] and [3], some ACC specific constraints are also presented. Suppose the teams in ACC are numbered as follows: Clemson (abbreviation Clem; team 1), Duke (Duke; 2), Florida State (FSU; 3), Georgia Tech (GT; 4), Maryland (UMD; 5), North Carolina (UNC; 6), North Carolina State(NCSt; 7), Virginia (UVA; 8), and Wake Forest (Wake; 9).

1. **Rival matches in final game**. Every team except FSU has a traditional rival. The rival pairs are Clem-GT, Duke-UNC, UMD-UVA, and NCSt-Wake. In the last day, every team except FSU plays against its rival, unless it plays against FSU or has a bye.
2. **Popular Matches in February**. The following pairings must occur at least once in days 11 to 18: Duke-GT, Duke-Wake, GT-UNC, UNC-Wake.
3. **Opponent Sequence Criteria**. No team plays in two consecutive days on road against Duke and UNC. No team plays in three consecutive days against Duke, UNC and Wake (independent of home/road).
4. **Idiosyncratic Criteria**. UNC plays Duke in the last day at Duke. UNC plays Clem in the second day. Duke has a bye in day 16. Wake does not play home in day 17 and has a bye in the first day. Clem, UMD and Wake do not play on road in the last day. Clem, FSU, GT and Wake do not play on road in the first day. Neither FSU nor NCSt have a bye in the last day.

All these constraints can be specified on variables $p_{x,y,z}$ and converted into propositional clauses. Using Sato, we could obtain 179 timetables in less than two seconds. In comparison, Henz' Friar Tuck [4] took 53.7 seconds. Because we used a different set of variables, we are not sure if the improvement is the result of new specification of constraints or is due to the efficiency of the SAT solver. Nonetheless, this experiment shows clearly that SAT solvers provide an alternative and efficient tool for scheduling problems.

4 Scheduling BigTen Conference Basketball

With the success of generating schedules for ACC, we applied the three-phases approach to the BigTen conference.

At first, we noticed that there is no mirroring scheme for the 18 game days in the BigTen case. So we dropped the mirror constraint and obtained 23786 patterns in Phase 1. In Phase 2, we searched for pattern sets with the following constraints.

1. **A set has 11 patterns.** $\sum_i x_i = 11$.

2. **Each day, except days 9-10, has five home games, five road games, and one bye.** For $1 \leq z \leq 18$, $z \neq 9, 10$: (a) $\sum_i h_{i,z}x_i = 5$, (b) $\sum_i r_{i,z}x_i = 5$, (c) $\sum_i b_{i,z}x_i = 1$.

3. **For days 9-10, it has four home games, four road games, and three byes.** For $9 \leq z \leq 10$: (a) $\sum_i h_{i,z}x_i = 4$, (b) $\sum_i r_{i,z}x_i = 4$, (c) $\sum_i b_{i,z}x_i = 3$.

4. **Two patterns in a set must be able to play a game.** For every pair i, i', if $h_{i,z}+r_{i',z} \neq 2$ for all z, then $x_i + x_{i'} < 1$.

Because of the large number of patterns, we failed to find any pattern set after an overnight run. From the number of patterns, we can see that a mirroring scheme can reduce the search space significantly. Since a mirroring scheme works only for a double round-robin tournament, we may first schedule a double round-robin tournament (of 22 days) for BigTen conference basketball, and then cut off the last four game days to make it a schedule of 18 game days. Of course, the last four games of any pattern in a double round-robin tournament of 11 teams cannot be arbitrary. To narrow down the search space, we impose the following constraints to the last four days of a pattern: There is at most one weekend home game, one weekday home game, one weekend road game, and one weekday road game in the last four days. That is,

$$r_{20} + r_{22} < 2, \quad h_{20} + h_{22} < 2, \quad r_{19} + r_{21} < 2, \quad h_{19} + h_{21} < 2.$$

If there is a bye in the last four days, then the pattern takes one of the following as its last four days.

(1)	b	+	+	−
(2)	+	b	−	+
(3)	−	+	b	−
(4)	+	−	−	b

Suppose we found a double round-robin tournament schedule of 11 teams from a pattern set and teams x, y, u, v are assigned the patterns having (1), (2), (3), and (4), respectively, as its last four days. After the last four days are cut from the schedule, we also need to remove the home game of u against x and the home game of v against y. The result is a balanced BigTen basketball regular season schedule.

In Phase 1, using the same constraints given in Section 3.1 with a simple mirroring scheme, $m = \{(i\ i+11) \mid 1 \leq i \leq 10\}$, for Constraint 2, plus the above constraints for the last four games, Sato found 75 patterns in 0.02 seconds. In Phase 2, Sato found the first pattern set (of size 11) in 0.01 seconds and found all the 5499 pattern sets in 278 seconds. In Phase 3, Sato found a timetable from the first pattern set in 0.25 seconds (see Appendix A). Thousands of schedules can be found this way.

5 Conclusion

We have shown that current college basketball schedules are far from perfect and modern computer technology can help. We demonstrated that a SAT solver like Sato [12] can provide an efficient tool for modeling and solving sports tournament scheduling problems. We are able to generate all the 179 solutions to the ACC 1997/98 tournament scheduling problem presented by Nemhauser and Trick [6] in two seconds of CPU time. We created the first balanced BigTen conference basketball schedule where each team plays the same number of home/road games on both weekends and weekdays. We also showed how to use mirroring schemes in a partial double round-robin tournament like the BigTen conference basketball schedule. The short "turn-around-time" of our solution allows us to quickly pass through multiple cycles of problem refinement to satisfy all parties involved.

A fast SAT prover is just like fast computer hardware. How to specify a problem in propositional logic is like write software applications for the computer. We all know how the efficiency of software will affect the performance of problem solving. This is also true for problem solving using SAT solvers. This was shown to be the case in solving Latin square problems [11], and we showed here again in solving sports tournament scheduling problems.

References

[1] Cain, W.O., Jr.: The computer-assisted heuristic approach used to schedule the major league baseball clubs. In Ladany and Machol (eds.) *Optimal Strategies in Sports*, 5, Studies in Management Science and Systems, pp. 32-41. North-Holland Pub., 1977.

[2] J.M. Crawford and A.B. Baker. Experimental results on the application of satisfiability algorithms to scheduling problems. In Proceedings of the 12th AAAI, 1092–1097, 1994. `http://citeseer.nj.nec.com/crawford94experimental.html`

[3] Henz, M.: Scheduling a major college basketball conference – revisited. *Operation Research*, 49(1), 2001.

[4] Henz, M.: Friar Tuck 1.1: A constraint-based round robin planner. The software is available at `http://www.comp.nus.edu.sg/~henz/projects/FriarTuck`, 2002

[5] Marriott, K., Stuckey, P.: Programming with constraints. MIT Press, Cambridge, MA, 1998.

[6] Nemhauser, G., Trick, M.: Scheduling a major college basketball conference. *Operation Research*, 46(1), 1998.

[7] Schruder J.A.M.: Combinatorial aspects of construction of competition dutch professional football leagues. *Discrete Applied Mathematics*, 35, 301-312, 1992.

[8] Wallace, M.: Practical applications of constraint programming. Constraints, 1(1&2), 139-168.

[9] de Werra, D.: Some models of graphs for scheduling sports competitions. *Discrete Applied Mathematics*, 21, 47-65, 1988.

[10] Yahoo!Sports `http://sports.yahoo.com/ncaab/standings.html`, 2002.

[11] Zhang, H.: Specifying Latin squares in propositional logic, in R. Veroff (ed.): Automated Reasoning and Its Applications, Essays in honor of Larry Wos, Chapter 6, MIT Press, 1997.

[12] Zhang, H.: SATO: An efficient propositional prover, Proc. of International Conference on Automated Deduction (CADE-97). pp. 308–312, Lecture Notes in Artificial Intelligence 1104, Springer-Verlag, 1997.

[13] Zhang, H., Stickel, M.: Implementing the Davis-Putnam method, *J. of Automated Reasoning* 24: 277-296, 2000.

Acknowledgement

Many thanks to Sue Bruell for proofreading an earlier version of this paper.

The 2002 BigTen Conference Basketball Schedule

The BigTen conference consists of the following universities: Illinois (Ill), Indiana (Ind), Iowa (Iow), Michigan (Mic), Michigan State (MiS), Minnesota (Min), Northwestern (Nor), Ohio State (Ohi), Penn State (Pen), Purdue (Pur), Wisconsin (Wis).

The 2002 season has 9 weeks, from January 2 to March 2. Two games are scheduled each week, one weekend and one weekday. The number in the parentheses following the week indicates whether it is a weekday (1) or a weekend (2).

```
Weeks |  Ill   Ind   Iow   Mic   MiS   Min   Nor   Ohi   Pen   Pur   Wis
------+-----------------------------------------------------------------
W1(1) |  Min  @Nor   Wis  @Pen   bye  @Ill   Ind  @Pur   Mic   Ohi  @Iow
W1(2) | @Wis   Pen  @Ohi   Pur  @Min   MiS   bye   Iow  @Ind  @Mic   Ill
W2(1) | @Pur   MiS   Nor  @Min  @Ind   Mic  @Iow   bye   Wis   Ill  @Pen
W2(2) |  Mic  @Iow   Ind  @Ill   Wis  @Pur   Ohi  @Nor   bye   Min  @MiS
W3(1) |  Iow   bye  @Ill   Nor   Pur  @Wis  @Mic   Pen  @Ohi  @MiS   Min
W3(2) |  bye  @Ohi  @Nor   Min  @Pen  @Mic   Iow   Ind   MiS  @Wis   Pur
W4(1) |  Wis  @Pen   MiS  @Ohi  @Iow   bye  @Pur   Mic   Ind   Nor  @Ill
W4(2) | @Ind   Ill  @Pur   bye   bye   Ohi   bye  @Min  @Wis   Iow   Pen
W5(1) | @Ohi   Pur   bye  @MiS   Mic   Pen   Wis   Ill  @Min  @Ind  @Nor
W5(2) |  Mis  @Min   Pen   Wis  @Ill   Ind  @Ohi   Nor  @Iow   bye  @Mic
W6(1) | @Mic   Iow  @Ind   Ill  @Nor   bye   MiS  @Wis   Pur  @Pen   Ohi
W6(2) |  Pur   bye   Min   Pen   Ohi  @Iow  @Wis  @MiS  @Mic  @Ill   Nor
W7(1) | @MiS   Wis  @Pen  @Pur   Ill  @Nor   Min   bye   Iow   Mic  @Ind
W7(2) |  bye  @Mic   Ohi   Ind  @Pur   Wis  @Pen  @Iow   Nor   MiS  @Min
W8(1) |  Pen   Ohi  @Wis   bye   Min  @MiS   Pur  @Ind  @Ill  @Nor   Iow
W8(2) | @Nor  @MiS   Mic  @Iow   Ind  @Pen   Ill   Pur   Min  @Ohi   bye
W9(1) |  Ind  @Ill   bye  @Wis  @Ohi   Nor   Min   MiS   bye   bye   Mic
W9(2) | @Min   Nor  @MiS   Ohi   Iow   Ill  @Ind  @Mic  @Pur   Pen   bye
```

Minimum number of games between repeating games: 2
Three or more consecutive home games (byes are ignored): 4
Three or more consecutive road games (byes are ignored): 5
Three consecutive weekends at home: 2
Three consecutive weekends on road: 2
Teams starting road/road (byes are ignored): 1
Teams ending road/road (byes are ignored): 0

A Balanced BigTen Conference Basketball Schedule

```
 Weeks |  Ill    Ind    Iow    Mic    MiS    Min    Nor    Ohi    Pen    Pur    Wis
 ------+------------------------------------------------------------------------------
 W1(1) | @Wis    Min    Ohi   @Nor    bye   @Ind    Mic   @Iow   @Pur    Pen    Ill
 W1(2) | @Min    bye    Pur    Wis    Nor    Ill   @MiS    Pen   @Ohi   @Iow   @Mic
 W2(1) |  Iow   @Wis   @Ill    bye   @Pur   @Ohi   @Pen    Min    Nor    MiS    Ind
 W2(2) | @MiS   @Pur   @Nor   @Pen    Ill    Wis    Iow    bye    Mic    Ind   @Min
 W3(1) |  bye    Ohi    Wis    Pur    Pen    Nor   @Min   @Ind   @MiS   @Mic   @Iow
 W3(2) |  Mic    MiS    bye   @Ill   @Ind   @Pur    bye   @Wis    bye    Min    Ohi
 W4(1) | @Ohi   @Pen   @MiS   @Min    Iow    Mic    Wis    Ill    Ind    bye   @Nor
 W4(2) |  Pen    Nor   @Mic    Iow   @Ohi    bye   @Ind    MiS   @Ill   @Wis    Pur
 W5(1) |  Pur   @Mic    Min    Ind    Wis   @Iow    Ohi   @Nor    bye   @Ill   @MiS
 W5(2) | @Nor   @Iow    Ind   @MiS    Mic   @Pen    Ill   @Pur    Min    Ohi    bye
 W6(1) | @Ind    Ill    bye   @Ohi   @Min    MiS   @Pur    Mic   @Wis    Nor    Pen
 W6(2) |  Wis   @Min   @Ohi    Nor    bye    Ind   @Mic    Iow    Pur   @Pen   @Ill
 W7(1) |  Min    bye   @Pur   @Wis   @Nor   @Ill    MiS   @Pen    Ohi    Iow    Mic
 W7(2) | @Iow    Wis    Ill    bye    Pur    Ohi    Pen   @Min   @Nor   @MiS   @Ind
 W8(1) |  MiS    Pur    Nor    Pen   @Ill    bye   @Iow    bye   @Mic   @Ind    bye
 W8(2) |  bye   @Ohi   @Wis   @Pur   @Pen   @Nor    Min    Ind    MiS    Mic    Iow
 W9(1) | @Mic   @MiS   @Pen    Ill    Ind    Pur    bye    Wis    Iow   @Min   @Ohi
 W9(2) |  Ohi    Pen    MiS    Min   @Iow   @Mic   @Wis   @Ill   @Ind    bye    Nor
```

Minimum number of games between repeating games: 10
Three or more consecutive home games: 0
Three or more consecutive road games: 0
Three consecutive weekends at home: 0
Three consecutive weekends on road: 0
Teams starting road/road (byes are ignored): 2
Teams ending road/road (byes are ignored): 0

The Pattern Set Used

```
 1:  +  b  -  -  +  +  -  +  -  -  +  -  b  +  +  -  -  +  -  +  +  -
 2:  +  +  -  -  +  +  -  -  +  +  b  -  -  +  +  -  -  +  +  -  -  b
 3:  -  +  b  -  +  -  -  +  +  -  -  +  -  b  +  -  +  +  -  -  +  +
 4:  b  +  -  +  +  -  +  -  +  +  -  b  -  +  -  -  +  -  +  -  -  +
 5:  -  +  -  +  +  -  +  b  -  -  +  +  -  +  -  -  +  -  b  +  +  -
 6:  +  -  -  +  -  b  +  -  +  +  -  -  +  +  -  +  b  -  +  -  -  +
 7:  -  +  +  b  -  -  +  +  -  -  +  +  -  -  b  +  +  -  -  +  +  -
 8:  -  -  +  +  -  -  +  -  b  +  -  +  +  -  -  +  +  -  +  b  -  +
 9:  +  -  +  +  -  +  b  -  -  +  +  -  +  -  -  +  -  b  +  +  -  -
10:  +  -  +  -  -  +  -  +  -  b  +  -  +  -  +  +  -  +  -  +  b  -
11:  -  -  +  -  b  +  -  +  +  -  -  +  +  -  +  b  -  +  -  -  +  +
```