# 07—Application of SAT Solving

CS 3234: Logic and Formal Systems

Martin Henz

September 30, 2010

Generated on Friday 1st October, 2010, 15:32

1. Propositional Logic: Application of SAT Solving

# The ACC 1997/98 Problem

- “ACC” stands for “Atlantic Coast Conference”, an American college basketball organization

# The ACC 1997/98 Problem

- “ACC” stands for “Atlantic Coast Conference”, an American college basketball organization
- 9 teams participate in tournament

# The ACC 1997/98 Problem

- “ACC” stands for “Atlantic Coast Conference”, an American college basketball organization
- 9 teams participate in tournament
- dense double round robin: there are $2 * 9$ dates

# The ACC 1997/98 Problem

- "ACC" stands for "Atlantic Coast Conference", an American college basketball organization
- 9 teams participate in tournament
- dense double round robin: there are $2 * 9$ dates
- at each date, each team plays either home, away or has a "bye"

# The ACC 1997/98 Problem

- “ACC” stands for “Atlantic Coast Conference”, an American college basketball organization
- 9 teams participate in tournament
- dense double round robin: there are $2 * 9$ dates
- at each date, each team plays either home, away or has a “bye”
- Each team must play each other team once at home and once away.

# The ACC 1997/98 Problem

- “ACC” stands for “Atlantic Coast Conference”, an American college basketball organization
- 9 teams participate in tournament
- dense double round robin: there are $2 * 9$ dates
- at each date, each team plays either home, away or has a “bye”
- Each team must play each other team once at home and once away.
- there should be at least 7 dates distance between first leg and return match.

# The ACC 1997/98 Problem

- "ACC" stands for "Atlantic Coast Conference", an American college basketball organization
- 9 teams participate in tournament
- dense double round robin: there are $2 * 9$ dates
- at each date, each team plays either home, away or has a "bye"
- Each team must play each other team once at home and once away.
- there should be at least 7 dates distance between first leg and return match.
- To achieve this, we assume a fixed mirroring between dates: (1,8), (2,9), (3,12), (4,13), (5,14), (6,15) (7,16), (10,17), (11,18)

# The ACC 1997/98 Problem (contd)

- No team can play away on both last dates

# The ACC 1997/98 Problem (contd)

- No team can play away on both last dates
- No team may have more than two away matches in a row.

# The ACC 1997/98 Problem (contd)

- No team can play away on both last dates
- No team may have more than two away matches in a row.
- No team may have more than two home matches in a row.

# The ACC 1997/98 Problem (contd)

- No team can play away on both last dates
- No team may have more than two away matches in a row.
- No team may have more than two home matches in a row.
- No team may have more than three away matches or byes in a row.

# The ACC 1997/98 Problem (contd)

- No team can play away on both last dates
- No team may have more than two away matches in a row.
- No team may have more than two home matches in a row.
- No team may have more than three away matches or byes in a row.
- No team may have more than four home matches or byes in a row.

# The ACC 1997/98 Problem (contd)

- Of the weekends, each team plays four at home, four away, and one bye.

# The ACC 1997/98 Problem (contd)

- Of the weekends, each team plays four at home, four away, and one bye.
- Each team must have home matches or byes at least on two of the first five weekends.

# The ACC 1997/98 Problem (contd)

- Of the weekends, each team plays four at home, four away, and one bye.
- Each team must have home matches or byes at least on two of the first five weekends.
- Every team except FSU has a traditional rival. The rival pairs are Clem-GT, Duke-UNC, UMD-UVA and NCSt-Wake. In the last date, every team except FSU plays against its rival, unless it plays against FSU or has a bye.

# The ACC 1997/98 Problem (contd)

- The following pairings must occur at least once in dates 11 to 18: Duke-GT, Duke-Wake, GT-UNC, UNC-Wake.

# The ACC 1997/98 Problem (contd)

- The following pairings must occur at least once in dates 11 to 18: Duke-GT, Duke-Wake, GT-UNC, UNC-Wake.
- No team plays in two consecutive dates away against Duke and UNC. No team plays in three consecutive dates against Duke UNC and Wake.

# The ACC 1997/98 Problem (contd)

- The following pairings must occur at least once in dates 11 to 18: Duke-GT, Duke-Wake, GT-UNC, UNC-Wake.
- No team plays in two consecutive dates away against Duke and UNC. No team plays in three consecutive dates against Duke UNC and Wake.
- UNC plays Duke in last date and date 11.

# The ACC 1997/98 Problem (contd)

- The following pairings must occur at least once in dates 11 to 18: Duke-GT, Duke-Wake, GT-UNC, UNC-Wake.
- No team plays in two consecutive dates away against Duke and UNC. No team plays in three consecutive dates against Duke UNC and Wake.
- UNC plays Duke in last date and date 11.
- UNC plays Clem in the second date.

# The ACC 1997/98 Problem (contd)

- The following pairings must occur at least once in dates 11 to 18: Duke-GT, Duke-Wake, GT-UNC, UNC-Wake.
- No team plays in two consecutive dates away against Duke and UNC. No team plays in three consecutive dates against Duke UNC and Wake.
- UNC plays Duke in last date and date 11.
- UNC plays Clem in the second date.
- Duke has bye in the first date 16.

# The ACC 1997/98 Problem (contd)

- Wake does not play home in date 17.

# The ACC 1997/98 Problem (contd)

- Wake does not play home in date 17.
- Wake has a bye in the first date.

# The ACC 1997/98 Problem (contd)

- Wake does not play home in date 17.
- Wake has a bye in the first date.
- Clem, Duke, UMD and Wake do not play away in the last date.

# The ACC 1997/98 Problem (contd)

- Wake does not play home in date 17.
- Wake has a bye in the first date.
- Clem, Duke, UMD and Wake do not play away in the last date.
- Clem, FSU, GT and Wake do not play away in the fist date.

# The ACC 1997/98 Problem (contd)

- Wake does not play home in date 17.
- Wake has a bye in the first date.
- Clem, Duke, UMD and Wake do not play away in the last date.
- Clem, FSU, GT and Wake do not play away in the fist date.
- Neither FSU nor NCSt have a bye in the last date.

# The ACC 1997/98 Problem (contd)

- Wake does not play home in date 17.
- Wake has a bye in the first date.
- Clem, Duke, UMD and Wake do not play away in the last date.
- Clem, FSU, GT and Wake do not play away in the fist date.
- Neither FSU nor NCSt have a bye in the last date.
- UNC does not have a bye in the first date.

# Background

- Trick and Nemhauser work on the problem from 1995 onwards

# Background

- Trick and Nemhauser work on the problem from 1995 onwards
- Trick and Nemhauser publish the problem and their approach in “Scheduling a Major Basketball Conference”, Operations Research, 46(1), 1998

# Background

- Trick and Nemhauser work on the problem from 1995 onwards
- Trick and Nemhauser publish the problem and their approach in “Scheduling a Major Basketball Conference”, Operations Research, 46(1), 1998
- From then onwards, Henz, Walser and Zhang use different techniques to solve the problem

# General Approach

- Three phases:

# General Approach

- Three phases:
  1. Generate all possible patterns such as “A H B A H H A H A A H B H A A H H A”

# General Approach

- Three phases:
  1. Generate all possible patterns such as "A H B A H H A H A A H B H A A H H A"
  2. Generate all feasible 9-element pattern sets that can be used to construct a schedule

# General Approach

- Three phases:
  1. Generate all possible patterns such as "A H B A H H A H A A H B H A A H H A"
  2. Generate all feasible 9-element pattern sets that can be used to construct a schedule
  3. Generate schedules from pattern sets

# General Approach

- Three phases:
  1. Generate all possible patterns such as "A H B A H H A H A A H B H A A H H A"
  2. Generate all feasible 9-element pattern sets that can be used to construct a schedule
  3. Generate schedules from pattern sets
- Output: all feasible solutions, from which the organizers can choose the most suitable one

# Solution Techniques

- Nemhauser and Trick use integer programming for all three steps, leading to a “turn-around time” of 24 hours

# Solution Techniques

- Nemhauser and Trick use integer programming for all three steps, leading to a “turn-around time” of 24 hours
- Henz uses constraint programming, turn-around time of less than 1 minute, publishes his approach in “Scheduling a Major Basketball Conference—Revisited”, Operations Research, 49(1), 2001

# Solution Techniques

- Nemhauser and Trick use integer programming for all three steps, leading to a “turn-around time” of 24 hours
- Henz uses constraint programming, turn-around time of less than 1 minute, publishes his approach in “Scheduling a Major Basketball Conference—Revisited”, Operations Research, 49(1), 2001
- Zhang Hantao uses SAT solving, turn-around time of 2 seconds, see “Generating College Conference Basketball Schedules using a SAT Solver”

# Solution Techniques

- Nemhauser and Trick use integer programming for all three steps, leading to a “turn-around time” of 24 hours
- Henz uses constraint programming, turn-around time of less than 1 minute, publishes his approach in “Scheduling a Major Basketball Conference—Revisited”, Operations Research, 49(1), 2001
- Zhang Hantao uses SAT solving, turn-around time of 2 seconds, see “Generating College Conference Basketball Schedules using a SAT Solver”
- Different approach: In 1998, J.P. Walser described a local-search based method for finding some (not all) solutions, without using 3 phases

# How to Encode ACC as a SAT Formula

- Consider Phase 3: Generation of schedule, assigning teams to opponents at every day of the tournament

# How to Encode ACC as a SAT Formula

- Consider Phase 3: Generation of schedule, assigning teams to opponents at every day of the tournament
- For teams $x$, $y$, day $z$, introduce atom $p_{x,y,z}$

# How to Encode ACC as a SAT Formula

- Consider Phase 3: Generation of schedule, assigning teams to opponents at every day of the tournament
- For teams $x, y$, day $z$, introduce atom $p_{x,y,z} = T$ iff team $x$ plays a home game against team $y$ in day $z$.

# How to Encode ACC as a SAT Formula

- Consider Phase 3: Generation of schedule, assigning teams to opponents at every day of the tournament
- For teams $x, y$, day $z$, introduce atom $p_{x,y,z} = T$ iff team $x$ plays a home game against team $y$ in day $z$.
- Example of encoding constraints: "Each team must play each other team once at home and once away."

# How to Encode ACC as a SAT Formula

- Consider Phase 3: Generation of schedule, assigning teams to opponents at every day of the tournament
- For teams $x$, $y$, day $z$, introduce atom $p_{x,y,z} = T$ iff team $x$ plays a home game against team $y$ in day $z$.
- Example of encoding constraints: "Each team must play each other team once at home and once away."
- For every pair of distinct teams $s$ and $t$, we have:

$$
\begin{gathered}
(p_{s,t,1} \land \neg p_{s,t,2} \land \cdots \land \neg p_{s,t,18}) \lor \\
(\neg p_{s,t,1} \land p_{s,t,2} \land \neg p_{s,t,3} \land \cdots \land \neg p_{s,t,18}) \lor \\
\vdots \\
(\neg p_{s,t,1} \cdots \land \neg p_{s,t,17} \land p_{s,t,18})
\end{gathered}
$$

# How to Encode ACC as a SAT Formula

- Consider Phase 3: Generation of schedule, assigning teams to opponents at every day of the tournament
- For teams $x$, $y$, day $z$, introduce atom $p_{x,y,z} = T$ iff team $x$ plays a home game against team $y$ in day $z$.
- Example of encoding constraints: “Each team must play each other team once at home and once away.”
- For every pair of distinct teams $s$ and $t$, we have:

$$
\begin{gathered}
(p_{s,t,1} \wedge \neg p_{s,t,2} \wedge \cdots \wedge \neg p_{s,t,18}) \vee \\
(\neg p_{s,t,1} \wedge p_{s,t,2} \wedge \neg p_{s,t,3} \wedge \cdots \wedge \neg p_{s,t,18}) \vee \\
\vdots \\
(\neg p_{s,t,1} \cdots \wedge \neg p_{s,t,17} \wedge p_{s,t,18})
\end{gathered}
$$

- Convert formula into CNF, and use a complete SAT solver

# Some Statistics

- Zhang Hantao used the DPLL-based SAT solver SATO

# Some Statistics

- Zhang Hantao used the DPLL-based SAT solver SATO
- Phase 1: $18 \cdot 3 = 54$ propositional atoms, 1499 clauses, taking 0.01 seconds, resulting in 38 patterns

# Some Statistics

- Zhang Hantao used the DPLL-based SAT solver SATO
- Phase 1: $18 \cdot 3 = 54$ propositional atoms, 1499 clauses, taking 0.01 seconds, resulting in 38 patterns
- Phase 2: $38 \cdot 9 \cdot 3 = 1026$ propositional atoms, 569300 clauses, taking 0.60 seconds, resulting in 17 pattern sets

# Some Statistics

- Zhang Hantao used the DPLL-based SAT solver SATO
- Phase 1: $18 \cdot 3 = 54$ propositional atoms, 1499 clauses, taking 0.01 seconds, resulting in 38 patterns
- Phase 2: $38 \cdot 9 \cdot 3 = 1026$ propositional atoms, 569300 clauses, taking 0.60 seconds, resulting in 17 pattern sets
- Phase 3: $9 \cdot 9 + 9 \cdot 8 \cdot 18 = 1377$ propositional atoms, hundreds of thousands of clauses, taking less than 2 seconds, resulting in 179 solutions

# Conclusion

- For many discrete constraint satisfaction problems such as the ACC 1997/98 problem, an encoding in SAT and use of a state-of-the-art SAT solver provides an attractive solving technique.

# Conclusion

- For many discrete constraint satisfaction problems such as the ACC 1997/98 problem, an encoding in SAT and use of a state-of-the-art SAT solver provides an attractive solving technique.
- The approach takes advantage of the effort that the designers of SAT solvers such as SATO spent in order to optimize the solver.

# Conclusion

- For many discrete constraint satisfaction problems such as the ACC 1997/98 problem, an encoding in SAT and use of a state-of-the-art SAT solver provides an attractive solving technique.
- The approach takes advantage of the effort that the designers of SAT solvers such as SATO spent in order to optimize the solver.
- This works well, because the solver is independent of the application domain; it can be used without modification across application domains.