SMT solver for theory of all-different

Milan Banković Filip Marić
{milan,filip}@matf.bg.ac.rs

Department of Computer Science
Faculty of Mathematics
University of Belgrade

Workshop on Formal and Automated Theorem Proving and Applications.
Belgrade 2009.

Outline

Outline

Definition of all-different constraint

Definition

Given a set of variables $x_1, x_2, \ldots, x_n$, where each variable x_i takes values from its corresponding finite domain $D(x_i)$, then *alldiff* $(x_1, x_2, \ldots, x_n)$ means that every two different variables must take different values $(i \neq j \Rightarrow x_i \neq x_j)$.

Applications

Broad variety of all-different based problems can be reduced to the *SAT* problem, using the *SMT* approach:

- Puzzle solving (Sudoku, Latin Square, Eight Queens).
- Scheduling and timetabling.

Example of all-different based problem

Example (Latin square 5×5)

	3	4		
3	4	5		
4	5			
5				

Notes

- Each cell should be filled with a value from 1 to 5.
- Each row and each column is constrained by all-different constraint.
- Some values are imposed.

Example of all-different based problem

Example (Latin square 5 $\times$ 5)

1	2	3	4	5
2	3	4	5	1
3	4	5	1	2
4	5	1	2	3
5	1	2	3	4

Notes

- Each cell should be filled with a value from 1 to 5.
- Each row and each column is constrained by all-different constraint.
- Some values are imposed.

Outline

Representation of problem

$$
\begin{array}{cr}
(x_{11} = 1 \vee x_{11} = 2 \vee \ldots \vee x_{11} = 5) & \wedge \\
(x_{12} = 1 \vee x_{12} = 2 \vee \ldots \vee x_{12} = 5) & \wedge \\
\ldots & \wedge \\
(x_{55} = 1 \vee x_{55} = 2 \vee \ldots \vee x_{55} = 5) & \wedge \\
alldiff(x_{11}, x_{12}, x_{13}, x_{14}, x_{15}) & \wedge \\
alldiff(x_{21}, x_{22}, x_{23}, x_{24}, x_{25}) & \wedge \\
\ldots & \wedge \\
alldiff(x_{15}, x_{25}, x_{35}, x_{45}, x_{55}) & \wedge \\
x_{51} = 5 & \wedge \\
x_{41} = 4 & \wedge \\
\ldots & \wedge \\
x_{23} = 4 &
\end{array}
$$

Notes

- For each cell one variable is introduced.

Representation of problem

$$
\begin{array}{lr}
(x_{11} = 1 \lor x_{11} = 2 \lor \ldots \lor x_{11} = 5) & \land \\
(x_{12} = 1 \lor x_{12} = 2 \lor \ldots \lor x_{12} = 5) & \land \\
\ldots & \land \\
(x_{55} = 1 \lor x_{55} = 2 \lor \ldots \lor x_{55} = 5) & \land \\
alldiff(x_{11}, x_{12}, x_{13}, x_{14}, x_{15}) & \land \\
alldiff(x_{21}, x_{22}, x_{23}, x_{24}, x_{25}) & \land \\
\ldots & \land \\
alldiff(x_{15}, x_{25}, x_{35}, x_{45}, x_{55}) & \land \\
x_{51} = 5 & \land \\
x_{41} = 4 & \land \\
\ldots & \land \\
x_{23} = 4 &
\end{array}
$$

Notes

- For each cell one variable is introduced.
- For each variable one clause defines its domain.

Representation of problem

$$
\begin{array}{rr}
(x_{11} = 1 \lor x_{11} = 2 \lor \dots \lor x_{11} = 5) & \land \\
(x_{12} = 1 \lor x_{12} = 2 \lor \dots \lor x_{12} = 5) & \land \\
\dots & \land \\
(x_{55} = 1 \lor x_{55} = 2 \lor \dots \lor x_{55} = 5) & \land \\
\mathit{alldiff}(x_{11}, x_{12}, x_{13}, x_{14}, x_{15}) & \land \\
\mathit{alldiff}(x_{21}, x_{22}, x_{23}, x_{24}, x_{25}) & \land \\
\dots & \land \\
\mathit{alldiff}(x_{15}, x_{25}, x_{35}, x_{45}, x_{55}) & \land \\
x_{51} = 5 & \land \\
x_{41} = 4 & \land \\
\dots & \land \\
x_{23} = 4 &
\end{array}
$$

Notes

- For each cell one variable is introduced.
- For each variable one clause defines its domain.
- For each row and each column – unit clause with all-different.

Representation of problem

$$
\begin{aligned}
(x_{11} = 1 \vee x_{11} = 2 \vee \ldots \vee x_{11} = 5) \quad & \wedge \\
(x_{12} = 1 \vee x_{12} = 2 \vee \ldots \vee x_{12} = 5) \quad & \wedge \\
\ldots \quad & \wedge \\
(x_{55} = 1 \vee x_{55} = 2 \vee \ldots \vee x_{55} = 5) \quad & \wedge \\
alldiff(x_{11}, x_{12}, x_{13}, x_{14}, x_{15}) \quad & \wedge \\
alldiff(x_{21}, x_{22}, x_{23}, x_{24}, x_{25}) \quad & \wedge \\
\ldots \quad & \wedge \\
alldiff(x_{15}, x_{25}, x_{35}, x_{45}, x_{55}) \quad & \wedge \\
x_{51} = 5 \quad & \wedge \\
x_{41} = 4 \quad & \wedge \\
\ldots \quad & \wedge \\
x_{23} = 4 \quad &
\end{aligned}
$$

Notes

- For each cell one variable is introduced.
- For each variable one clause defines its domain.
- For each row and each column – unit clause with all-different.
- For each imposed value – unit clause with equality.

Representation of solution

$$
\begin{array}{l}
x_{11} = 1 \wedge x_{12} = 2 \wedge \ldots \wedge x_{15} = 5 \quad \wedge \\
x_{21} = 2 \wedge x_{22} = 3 \wedge \ldots \wedge x_{25} = 1 \quad \wedge \\
\qquad\qquad \ldots \qquad\qquad\qquad \wedge \\
x_{51} = 5 \wedge \ldots \wedge x_{52} = 1 \wedge x_{55} = 4
\end{array}
$$

Notes

- Solution of the problem can be represented as a model of the corresponding formula.

Representation of solution

$$x_{11} = 1 \wedge x_{12} = 2 \wedge \ldots \wedge x_{15} = 5 \quad \wedge$$
$$x_{21} = 2 \wedge x_{22} = 3 \wedge \ldots \wedge x_{25} = 1 \quad \wedge$$
$$\ldots \quad \wedge$$
$$x_{51} = 5 \wedge \ldots \wedge x_{52} = 1 \wedge x_{55} = 4$$

Notes

- Solution of the problem can be represented as a model of the corresponding formula.
- Model of the formula is an assignment of values to variables which satisfies all the formula's clauses.

Outline

About *SMT* solvers

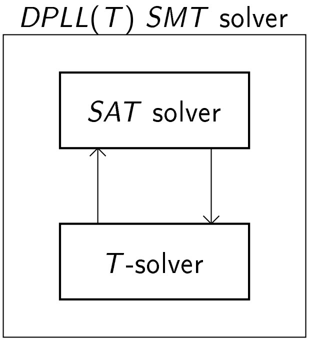
What is an *SMT* solver?

SMT solver checks for satisfiability of quantifier-free first-order formula with respect to some background theory.

"Lazy" approach

- Each atom in the formula is considered as a propositional symbol.
- *SAT* solver checks for propositional model of the formula.
- Propositional model is checked for satisfiability in the theory.

DPLL-based *SMT* solver structure



Notes

- *SAT* solver:
  - incrementally builds partial model
  - checks for propositional satisfiability
  - backtracks if conflict happens
- *T*-solver (or *Theory* solver):
  - Detects conflicts with theory
  - Handles theory propagations
  - Explains conflicts and/or propagations

Matching theory and all-different

Implementation of all-different T-solver

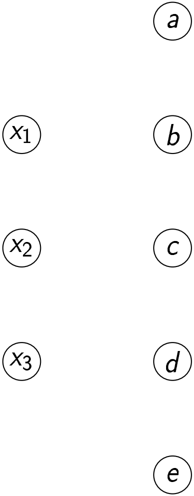
- T-solver implementation is based on matching theory in bipartite graphs.
- One bipartite graph is assigned to each all-different atom appearing in the formula.

Matching theory and all-different

Notes

- Each vertex at the left side corresponds to one variable.

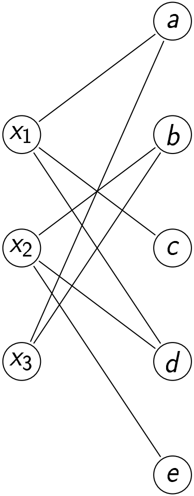
Matching theory and all-different



Notes

- Each vertex at the left side corresponds to one variable.
- Each vertex at the right side corresponds to one value.

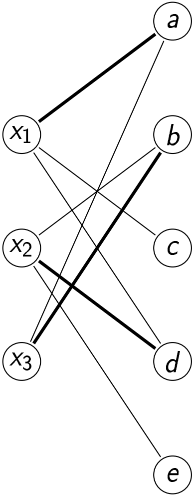
Matching theory and all-different



Notes

- Each vertex at the left side corresponds to one variable.
- Each vertex at the right side corresponds to one value.
- Each variable is connected to values from its domain.

Matching theory and all-different



Notes

- Each vertex at the left side corresponds to one variable.
- Each vertex at the right side corresponds to one value.
- Each variable is connected to values from its domain.
- Solution corresponds to matching that covers left side vertices.

Conflict detection

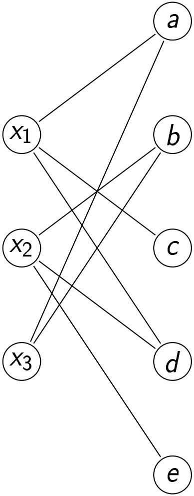
Definition

Given a partial model $\Delta = A_1, A_2, \ldots, A_n$, we say that model Δ is inconsistent with the theory T if $\Delta \vDash_T \bot$. We also say that Δ is in conflict with the theory T.

Conflict detection in all-different

- Optimal matching – matching with maximal cardinality.
- all-different is satisfiable if and only if optimal matching covers left side vertices.

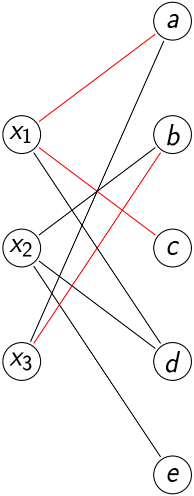
Conflict detection



Notes

- Partial model: $x_1 = d, x_3 \neq b$.

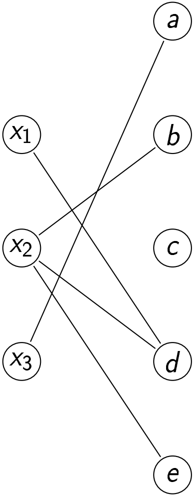
Conflict detection



Notes

- Partial model: $x_1 = d, x_3 \neq b$.
- Edges to remove: $x_1 = a, x_1 = c, x_3 = b$.

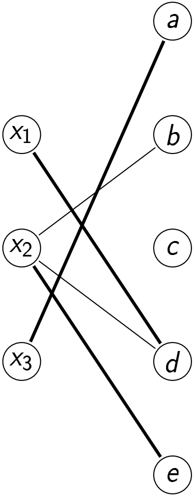
Conflict detection



Notes

- Partial model: $x_1 = d, x_3 \neq b$.
- Edges to remove: $x_1 = a, x_1 = c, x_3 = b$.

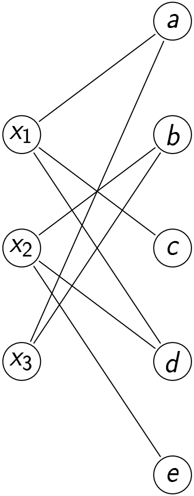
Conflict detection



Notes

- Partial model: $x_1 = d, x_3 \neq b$.
- Edges to remove: $x_1 = a, x_1 = c, x_3 = b$.
- Optimal matching covers set of left vertices, so all-different constraint is satisfiable.

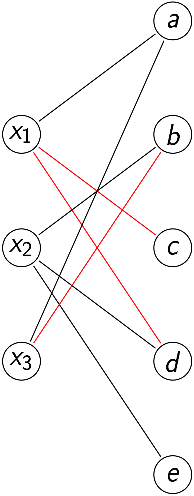
Conflict detection



Notes

- Partial model: $x_1 = a, x_3 \neq b$.

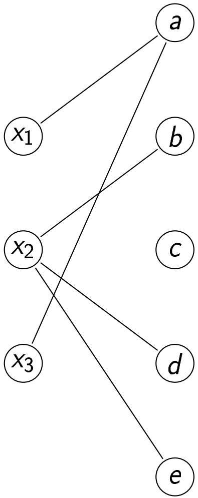
Conflict detection



Notes

- Partial model: $x_1 = a, x_3 \neq b$.
- Edges to remove: $x_1 = c, x_1 = d, x_3 = b$.

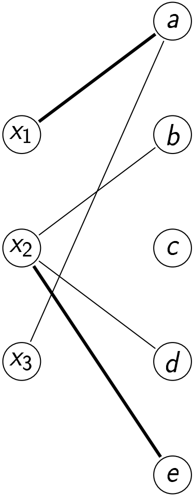
Conflict detection



Notes

- Partial model: $x_1 = a, x_3 \neq b$.
- Edges to remove: $x_1 = c, x_1 = d, x_3 = b$.

Conflict detection



Notes

- Partial model: $x_1 = a, x_3 \neq b$.
- Edges to remove: $x_1 = c, x_1 = d, x_3 = b$.
- Optimal matching doesn't cover set of left vertices, so all-different constraint is not satisfiable. Conflict is reported!

Conflict detection

Optimal matching construction

- Hopcroft and Karp's algorithm (1973.).
- The algorithm incrementally augments current matching until optimal matching is constructed.
- The algorithm executes in $\sqrt{k}$ graph traversals, where k is cardinality of constructed matching.

Theory propagations

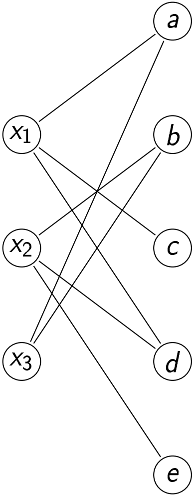
Definition

Given a partial model $\Delta = A_1, A_2, \ldots, A_n$, if $\Delta \vDash_T A$ holds for some atom $A \notin \Delta$, then atom A can be added to the partial model. This process is called theory propagation.

Theory propagations in all-different

- Vital edge – belongs to all optimal matchings (equality propagated).
- Inconsistent edge – doesn't belong to any optimal matching (disequality propagated).

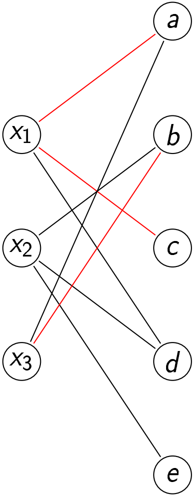
Theory propagations



Notes

- Partial model: $x_1 = d, x_3 \neq b$.

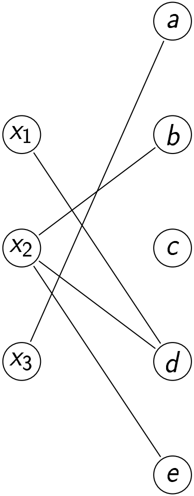
Theory propagations



Notes

- Partial model: $x_1 = d, x_3 \neq b$.
- Edges to remove: $x_1 = a, x_1 = c, x_3 = b$.

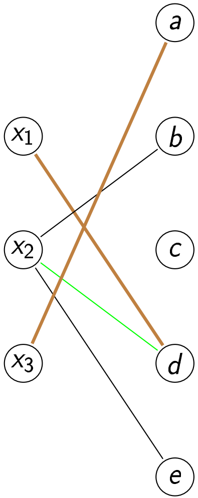
Theory propagations



Notes

- Partial model: $x_1 = d, x_3 \neq b$.
- Edges to remove: $x_1 = a, x_1 = c, x_3 = b$.

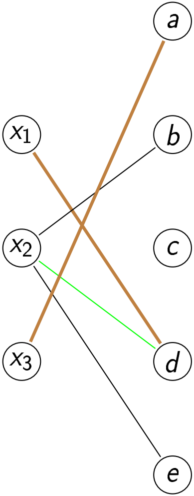
Theory propagations



Notes

- Partial model: $x_1 = d, x_3 \neq b$.
- Edges to remove: $x_1 = a, x_1 = c, x_3 = b$.
- Edges $x_3 = a, x_1 = d$ are vital. Edge $x_2 = d$ is inconsistent.

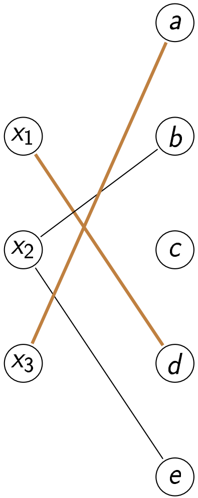
Theory propagations



Notes

- Partial model: $x_1 = d, x_3 \neq b$.
- Edges to remove: $x_1 = a, x_1 = c, x_3 = b$.
- Edges $x_3 = a, x_1 = d$ are vital. Edge $x_2 = d$ is inconsistent.
- Propagation of atoms: $x_3 = a, x_2 \neq d$.

Theory propagations



Notes

- Partial model: $x_1 = d, x_3 \neq b$.
- Edges to remove: $x_1 = a, x_1 = c, x_3 = b$.
- Edges $x_3 = a, x_1 = d$ are vital. Edge $x_2 = d$ is inconsistent.
- Propagation of atoms: $x_3 = a, x_2 \neq d$.
- Inconsistent edges can be removed from the graph.

Theory propagations

Finding vital and inconsistent edges

- Regin's filtering algorithm for all-different constraint (1994.).
- The algorithm executes in two graph traversals.

Theory propagations explaining

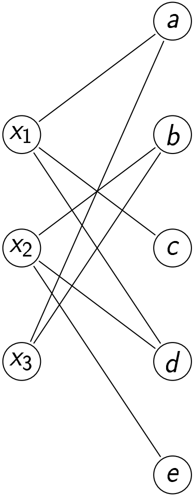
Definition

Given a partial model $\Delta = A_1, A_2, \ldots, A_k, \ldots, A_n$, if atom A_k is in model as a result of the theory propagation, then explanation of atom A_k is any subset Σ of $A_1, A_2, \ldots, A_{k-1}$ for which $\Sigma \vDash_T A_k$ holds.

Theory propagations explaining in all-different

Atom A (equality or disequality) is implied (in all-different theory) by set of atoms Σ if and only if the edge corresponding to the atom A is vital or inconsistent in the graph state corresponding to the model $\Delta = \Sigma$.

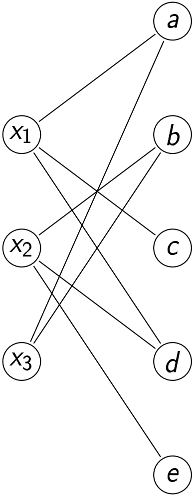
Theory propagations explaining



Notes

- Partial model: $x_1 = d, x_3 \neq b, x_2 \neq e, x_3 = a$.
- Atom $x_3 = a$ has been added to the model by the theory propagation.

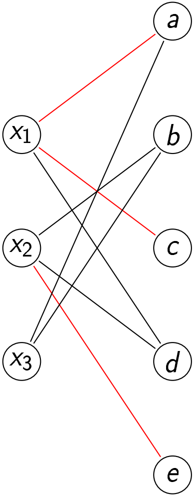
Theory propagations explaining



Notes

- Partial model: $x_1 = d, x_3 \neq b, x_2 \neq e, x_3 = a$.
- Atom $x_3 = a$ has been added to the model by the theory propagation.
- $x_1 = d, x_2 \neq e \vDash_T x_3 = a$, because...

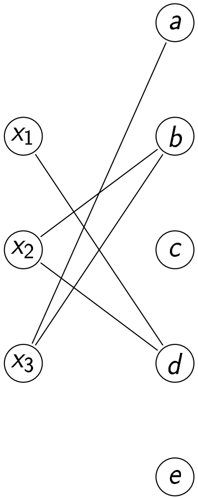
Theory propagations explaining



Notes

- Partial model: $x_1 = d, x_3 \neq b, x_2 \neq e, x_3 = a$.
- Atom $x_3 = a$ has been added to the model by the theory propagation.
- $x_1 = d, x_2 \neq e \vDash_T x_3 = a$, because... after removal of edges $x_1 = a, x_1 = c, x_2 = e$...

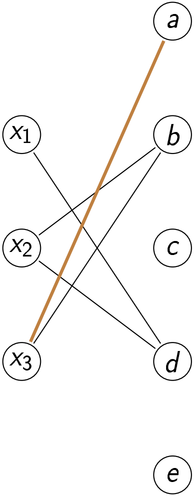
Theory propagations explaining



Notes

- Partial model: $x_1 = d, x_3 \neq b, x_2 \neq e, x_3 = a$.
- Atom $x_3 = a$ has been added to the model by the theory propagation.
- $x_1 = d, x_2 \neq e \vDash_T x_3 = a$, because... after removal of edges $x_1 = a, x_1 = c, x_2 = e$...

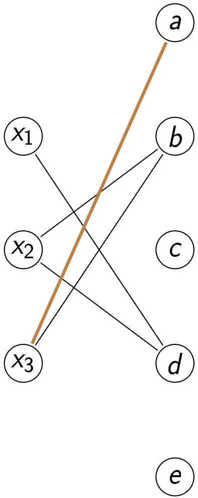
Theory propagations explaining



Notes

- Partial model: $x_1 = d, x_3 \neq b, x_2 \neq e, x_3 = a$.
- Atom $x_3 = a$ has been added to the model by the theory propagation.
- $x_1 = d, x_2 \neq e \vDash_T x_3 = a$, because... after removal of edges $x_1 = a, x_1 = c, x_2 = e$... edge $x_3 = a$ is vital.

Theory propagations explaining



Notes

- Partial model: $x_1 = d, x_3 \neq b, x_2 \neq e, x_3 = a$.
- Atom $x_3 = a$ has been added to the model by the theory propagation.
- $x_1 = d, x_2 \neq e \vDash_T x_3 = a$, because... after removal of edges $x_1 = a, x_1 = c, x_2 = e$... edge $x_3 = a$ is vital.
- Possible explanation: $x_1 = d, x_2 \neq e$.

Theory propagations explaining

Algorithm for finding minimal explanation

- The algorithm we propose can find explanation which is minimal in sense of inclusion.
- The algorithm is based on Regin's algorithm.
- Proposed algorithm is very efficient, because it executes in two graph traversals.

Outline

Future work

Future work

- Our all-different T-solver is planned to be integrated into *ArgoSMT*, which is generic *SMT* platform in early stage of developement.
- Possible application: timetabling (teaching timetable for Faculty of Mathematics).

Conclusions

Conclusions

- All-different constraint: broad application area.
- *SMT*-approach: reducing all-different based problems to *SAT*.
- Efficient decision procedures based on matching theory.
- Theory propagations explaining: new efficient algorithm is proposed.