VU Programm- und Systemverifikation
Homework exercise: Propositional Logic / SAT

(10 points)

May 13, 2015

Consider the package dependency graph shown in Figure 1. Nodes depict software packages; a solid arrow from package P_i to package P_j requires P_j to be installed, if P_i is installed; a dashed arrow from package P_i to package P_j prohibits P_i to be installed, if P_j is installed. We model information about installed packages using 15 boolean variables $x_1, \ldots, x_{15}$: for $i : 1 \leq i \leq 15$, if $x_i = \mathit{true}$, then package i is installed.

Encode the constraints of the graph in Figure 1 as a CNF. Create the file `graph.cnf` and write down the CNF there in the DIMACS format.[1] Use exactly 15 variables: for a package P_i, use the boolean variable with the number i, for $1 \leq i \leq 15$.

You can check the correctness of your encoding with the following tests:

1. For each solid arrow $i \to j$ in the graph, if the user adds the clauses `"i 0"` and `"j 0"` to `graph.cnf` (with the number of clauses fixed accordingly) and runs the solver on `graph.cnf`, the solver returns "SAT".

2. For each pair (i, j) that does not have a solid arrow $i \to j$ in the graph, if the user adds the clauses `"i 0"` and `"-j 0"` to `graph.cnf` (with the number of clauses fixed accordingly) and runs the solver on `graph.cnf`, the solver returns "SAT".

3. If the user adds the clauses `"1 0"` and `"2 0"` to `graph.cnf` (with the number of clauses fixed accordingly) and runs the solver on `graph.cnf`, the solver returns "SAT".

4. If the user adds the clauses `"1 0"` and `"3 0"` to `graph.cnf` (with the number of clauses fixed accordingly) and runs the solver on `graph.cnf`, the solver returns "UNSAT".

Upload the file `graph.cnf` to TUWEL by May 27, 2014. Make sure that the file contains your name and ID (as a comment). Note that `graph.cnf` **must** contain only the CNF describing Figure 1, not including the changes specified in points (1)–(4).

[1] The DIMACS format: `http://www.satcompetition.org/2009/format-benchmarks2009.html`

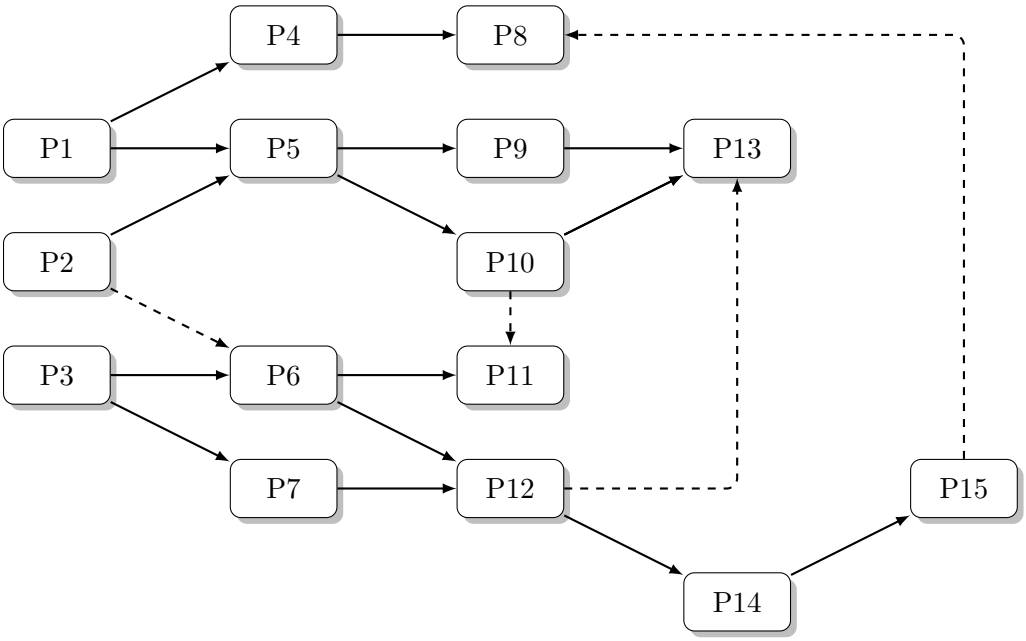

Figure 1: The package dependency graph: nodes depict software packages; a solid arrow from package P_i to package P_j requires P_j to be installed, if P_i is installed; a dashed arrow from package P_i to package P_j prohibits P_i to be installed, if P_j is installed.