This exam consists of **5** exercises. The available points for each item are written in the margin. In total there are 90 points. You need 45 points to pass.

1 Algorithms in Linear Arithmetic

Consider the following formula:

$$\varphi := \exists x.\exists y.\ 2x + 3y < 2 \wedge -x + 4y \geq 1$$

[10] (a) Remove the quantifier of y within φ using the method of Ferrante and Rackoff. You *do not* have to simplify the formula after the removal of the quantifier.

[10] (b) Start to solve φ using the simplex method: apply all initial steps and *one* iteration of the main loop. Use Bland's selection rule with the variable order $x < y < s < t$ where s and t are the introduced slack variables.
Is a second iteration of the main loop required? Just answer this question with a yes or no.

2 Understanding Linear Arithmetic

For some optimization problems it is required to consider *mixed* linear arithmetic problems, where the set of variables $\mathcal{V}$ is partitioned into $\mathcal{V}_\mathbb{Z} \uplus \mathcal{V}_\mathbb{Q}$, i.e., some variables are restricted to represent integers whereas the others are for rational numbers. For instance $\mathcal{V}_\mathbb{Z}$ might describe quantities which cannot be split, such as wheels, screws, etc., whereas $\mathcal{V}_\mathbb{Q}$ is used for amounts of fluids, for costs, etc.

A mixed solution of a quantifier-free formula φ is an assignment $v : \mathcal{V} \to \mathbb{Q}$ that satisfies φ and where additionally $v(x) \in \mathbb{Z}$ for all $x \in \mathcal{V}_\mathbb{Z}$.

Recall the Diophantine equation solver of Griggio. It converts any conjunction of linear equations into an equi-satisfiable solved form or reports unsatisfiability (both in $\mathbb{Z}$).

Here, a *solved form* is a list of equations where

1. each entry in the list is of the shape $x = e_x$ where e_x is a linear expression with integer coefficients; and
2. there are no cyclic dependencies: e_x in list $[\ldots, x = e_x, \ldots, y = e_y, \ldots]$ does neither contain x nor y.

Your task is to design an equation solver for the mixed setting.

[3] (a) Generalize the definition of solved form to the mixed setting, i.e., define the notion of a *mixed solved form*, such that equations in mixed solved form always have a mixed solution.

[12] (b) Design a mixed equation solver that takes as input a conjunction of linear equations. It either returns an equi-satisfiable mixed solved form or reports unsatisfiability (both in the mixed setting).
Of course, here you may invoke Griggio's solver as a sub-routine.

[10] (c) Apply your algorithm on the following equations where $x \in \mathcal{V}_\mathbb{Q}$ and $y, z \in \mathcal{V}_\mathbb{Z}$.

$$\begin{aligned} 2x + 3y &= 5 \\ 3x + 2y + 5z &= 4 \end{aligned}$$

Important: If you did not solve parts (a) and (b), then in part (c) you may assume that $x \in \mathcal{V}_\mathbb{Z}$ and should just apply Griggio's algorithm on the example constraints.

3 Algorithms for SMT

Consider a formula that combines difference logic over $\mathbb{Q}$ with EUF.

[3] (a) Is difference logic over $\mathbb{Q}$ convex?

[12] (b) Depending on the previous answer, choose the deterministic or the non-deterministic version of the Nelson–Oppen algorithm to determine satisfiability of the following formula.

$$f(f(x-5)) = f(a) \wedge y = f(z+2) \wedge x - z = 7 \wedge y \neq a$$

Provide intermediate steps of the Nelson–Oppen algorithm. Here, you should at least once illustrate the calculations of the congruence closure algorithm for EUF in detail, but you *do not* have to provide any details on how difference logic constraints are solved.

4 Encoding Problems

Assume that we are interested in satisfiability of quantifier-free formulas using non-linear arithmetic over $\mathbb{N}$. An example formula looks as follows:

$$xy^2 \geq 5 + x \wedge y(x+z) \geq x^2 + 8 \vee 3x \geq 2y + z$$

Although the problem is in general undecidable, we might consider to search for solutions in a finite domain, e.g., by restricting the range of each variable to $0, 1 \ldots, 2^b - 1$.

[15] Describe an encoding which transforms a formula φ as above into some other formula ψ (in some *decidable* logic), such that φ has a solution in $\mathbb{N} \cap \{0, 1, \ldots, 2^b - 1\}$ iff ψ has a solution. The transformation should be computable in polynomial time, so just substituting all possible values $\{0, 1, \ldots, 2^b - 1\}$ for each variable is not allowed.

After describing your approach, also apply it on the example formula for $b = 3$.

5 Multiple Choice

[15] There are five questions on the answer sheet.

Mark your answers by crossing the correct box, e.g., like this: ☑.

- Each correct answer is worth 3 points.
- Each wrong answer is worth 0 points.
- Giving no answer to a question is worth 1 point.