Algorithm Engineering **Summer 2026**

Dr. Phillip Keldenich
Dr. Dominik Krupke

Submission due: 16.07.2026 16:45

Homework 6

Work in teams of two. Homework is due every two weeks.

Submission format: Host your solution in a private Git repository on one of the following platforms and add Rouven as a collaborator: IBR GitLab (`kniep`), TU GitLab (`r.kniep`), GitHub (`rkniep`), or Codeberg (`rkniep`). If you want to use a different platform, please ask first.

Before the deadline, send the link to your submission commit by e-mail to `kniep@ibr.cs.tu-bs.de`. A single PDF with your written answers is sufficient for this sheet.

This sheet is an **exam-preparation sheet**. Every task below is drawn verbatim from the exam question catalogue, so working through them now is the most direct preparation you can do. Answer in **your own words** and hand in written answers (a single PDF is enough). Keep answers concise but complete: a good answer is one you could reproduce under exam conditions.

Not every question will appear in the exam, and the exam may ask slight variants (reworded for precision, or — for the modeling and CDCL tasks — with a different problem or input). The point of this sheet is that you have prepared the underlying ideas, not memorized these specific instances.

Exercise 1 (CPU Architecture & Performance Engineering): (3+2+3+3 points)

a) Describe the terms 'array of structures' and 'structure of arrays' in your own words and the benefits and drawbacks of each.

b) When solving hard problems, what simple approach to parallelization can often be surprisingly (and embarrassingly) competitive with complex, parallelized solvers?

c) State Amdahl's Law and explain what it implies for where you should spend optimization effort.

d) Profiling tools fall into two big families. Name both, describe each mechanism, and give the main trade-off.

Exercise 2 (Data Structures & Optimization Fundamentals): (4+3 points)

a) State the theoretical amortized complexity of Union-Find (`find`/`unite`) with union-by-rank + path compression, and the complexity to assume in practice. Justify.

b) Why can heuristics that find good solutions be a valuable addition to branch & bound-based algorithms?

Exercise 3 (Declarative Modeling): **(7 points)**

Read the problem statement, then apply the five-component recipe — for each component extract the relevant phrase(s) and write the corresponding formal object.

> *A delivery company must assign every parcel from a given batch to exactly one of its available vans. Each parcel has a known weight, and each van has a fixed weight capacity that the total weight of the parcels loaded into it must not exceed. Loading a parcel into a particular van incurs a handling cost that depends on the parcel–van pair. The company wants to load every parcel while spending as little as possible on handling.*

Exercise 4 (LP & MIP): **(4+4 points)**

a) What are the benefits of the barrier interior point algorithm over simplex? What are the benefits of the simplex algorithm? Why can both be a valuable part of a MIP solver and even both be used in solving the same MIP?

b) State the general Big-M construction for gating a linear constraint by a binary. Then explain the main practical drawback of the Big-M method and the mitigations.

Exercise 5 (Graph & Network Algorithms and Routing): **(3+4 points)**

a) Time-dependent shortest paths (edge travel time depends on departure time) can still be solved with Dijkstra. Which condition must hold, and what does it mean intuitively?

b) Vehicle Routing (VRP): give an informal description of the core problem, then name at least three constraints/objective variants that occur in practice (restrict to those in the lecture).

Exercise 6 (SAT & CP-SAT): **(7+3 points)**

Notation: we write 1 *instead of* x_1 *and* $\overline{1}$ *instead of* $\overline{x_1}$*, and* 123 *instead of* $1 \vee 2 \vee 3$*.*

a) Perform CDCL on the following formula until you have run into two conflicts or found the formula to be satisfied or unsatisfiable. When making a decision, prioritize variables with lower indices over those with higher indices, and set the variable to false in your decision. To make intermediate steps readable, fill out a new trail after each conflict.

$$1\overline{2},\ 123,\ \overline{1}\,\overline{2}3,\ 12\overline{3}4,\ \overline{3}4,\ \overline{1}\,\overline{2}4,\ \overline{1}2$$

b) Why is branching by the VSIDS (SAT decision) heuristic alone not sufficient to reach a complete solution in CP-SAT? What mechanism closes the gap?