Assignment 3: Spacecraft Tourism Problem (STP)

Read the source code of the relevant demo report at http://user.it.uu.se/~pierref/courses/COCF/demoReport before tackling any assignment: it contains problem-independent instructions on what to hand in, as well as useful problem-independent indications on how to proceed. It is strongly recommended to read the Submission Instructions and Grading Rules at the end of this document before attempting to address its tasks. It is also strongly recommended to prepare and attend the help sessions, as huge time savings may ensue.

It is still the year 2137. You are consultants and a client organises space travel for tourist groups. The client company has a fleet of spacecrafts of various amounts of seats, and it knows, for each tourist group, the group size and the language all group members can speak. Assuming there are as many spacecrafts of each size as needed, the company needs to decide for each group which spacecraft to use, such that:

- there are enough seats in each spacecraft to accommodate all the groups that travel in it;
- no group is spread over several spacecrafts;
- at most two languages are spoken aboard each spacecraft; and
- the total number of unused seats during all the spaceflights is minimal.

There is no limit to the number of groups in a spacecraft, except as dictated by the seat and language constraints. There is no limit to the number of used spacecrafts of any particular size. The only objective function is the total number of unused seats: the spacecrafts the unused seats are located in and the number of used spacecrafts are irrelevant.

Here is a small feasible STP instance. There are two spacecraft sizes: 6 and 8 seats. There are four groups, named A, B, C, and D, which all speak a different language and have the following respective sizes: 2, 3, 3, and 5 persons. An optimal solution assigns groups A and D to a spacecraft of size 8, and groups B and C to a spacecraft of size 6: there is only one unused seat, in the 8-seater, and there is no solution with no unused seats. Assigning also groups B and C to a spacecraft of size 8 is sub-optimal as the number of unused seats goes up to three. Assigning groups A, B, and C to a spacecraft of size 8 and group D to a spacecraft of size 6 also yields one unused seat but is incorrect as three languages are spoken aboard the 8-seater.

The company currently has one instance of the problem, with 20 spacecraft sizes and 111 tourist groups speaking a total of 88 languages. As the company managers do not know
the power of combinatorial optimisation, they think that you might not be able to solve this instance to optimality in reasonable time. Hence they suggest you also try your approach by only taking into account the first $n$ tourist groups, with $n$ ranging from 2 to 111, thus generating 110 instances. A skeleton MiniZinc model with preprocessing to compute all the parameters based on the value of $n$, which is called nGroups in the model, and a file for the large instance are at [http://user.it.uu.se/~pierref/courses/COCP/assignments/assignment3/assignment3.zip](http://user.it.uu.se/~pierref/courses/COCP/assignments/assignment3/assignment3.zip). Perform the following sequence of tasks:

A. Write and evaluate a MiniZinc model STP.mzn in order to solve the STP. As this problem contains symmetries, you must introduce symmetry-breaking constraints.

**Hint:** Think about problem viewpoints, useful constraint predicates, pre-computation towards using element, as well as implied constraints. Recall that you may use a search annotation towards accelerating CP and LCG backends: first state a suitable search strategy in plain English and argue for it, and then formulate or approximate it as a MiniZinc search annotation.

**Hint:** A straightforward (and acceptable) way to model this problem is to have a size variable for each spacecraft. However, note that it is possible to model the problem without modelling the size of each spacecraft explicitly.

B. Briefly describe a real-world situation where (a variation) of the STP can occur.

C. (Optional) In case your model in Task A is very good, you may also evaluate it on the much harder instance STP-data-hard.mzn by modifying the include line in STP.mzn. Remember to restore the include line when submitting.

For the evaluation, start from $n = 2$ and report the results for the chosen backends for all the considered technologies. Increase $n$ by steps of 1 if your model does not perform well under all chosen backends, by steps of 10 if your model can solve all instances under at least one backend, and otherwise by an appropriate step size between 1 and 10. **Use any time-out of maximum 20 CPU minutes per instance in order to avoid too long solving times.** Note that the difficulty of instances increases with $n$, but with some exceptions, hence you can stop the experiments when all chosen backends time out for five consecutive instances. For your convenience, here are the optimal objective values for some small values of $n$:

<table>
<thead>
<tr>
<th>$n$</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
<th>9</th>
<th>10</th>
</tr>
</thead>
<tbody>
<tr>
<td>unused seats</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

This problem can be solved to optimality very efficiently even for the largest given instance.

**Submission Instructions**

All task answers, with imported MiniZinc models or (relevant parts of) Gecode programs, must be in a single report in PDF format; all other formats are rejected. Furthermore:

- Identify the team members and state the team number(s) inside the report.
- Address each task of each problem, using the numbering and the ordering in which they appear in this assignment statement.
- Take the instructions of the relevant demo report at [http://user.it.uu.se/~pierref/courses/COCP/demoReport](http://user.it.uu.se/~pierref/courses/COCP/demoReport) as a strict guideline for the structure and content of a model description, model evaluation, and task answer in the report, as well as an indication of the expected quality of the report.
You must use the script explained in the current MiniZinc cheatsheet at http://user.it.uu.se/~gusbj192/courses/M4CO; it conducts the experiments and generates a result table that can be automatically imported (rather than manually copied) into a LaTeX report: each time you change the model, it suffices to re-run that script and re-compile your report, without any tedious number copying!

- Write clear task answers, source code, and comments.
- Justify all task answers, except where explicitly not required.
- State any assumptions you make that are not in this document.
- Thoroughly proofread, spellcheck, and grammar-check your report.
- Upload all MiniZinc models or Gecode programs.
- Match exactly the uppercase, lowercase, and layout conventions of any filenames and I/O texts imposed by the tasks, as we will process your models automatically.
- Write a paragraph, which will not be graded, describing your experience with this assignment: Which aspects were too difficult or too easy? Which aspects were interesting or boring? This will help us improve the course in the coming years.
- Remember that when submitting you implicitly certify (a) that your report and all its uploaded attachments were produced solely by your team, except where explicitly stated otherwise and clearly referenced, (b) that each teammate can individually explain any part starting from the moment of submitting your report, and (c) that your report and attachments are not freely accessible on a public repository.

Unless the team spans both course 1DL441 and course 1DL451, only one of the teammates submits the solution files (one PDF report with answers to all the tasks, and all model or program files), without folder structure and without compression, via the Student Portal, whose clock may differ from yours, by the given hard deadline.

**Grading Rules**

If all tasks have been seriously attempted, and all requested models exist in files with the imposed names and the comments exemplified in the relevant demo report, and your models produce correct outputs for some instances in reasonable time, then you score at least 1 point (read on), otherwise your final score is 0 points. Furthermore:

- If your models meet the minimum requirements of the first four rows or all five rows in Table 1 below and your task answers are mostly correct, then you get a final score of 4 or 5 points, depending also on the quality of your model comments; you are not invited to the grading session.
- If your models meet the minimum requirements of at most the first three rows in Table 1 or your task answers have many errors, then you get an initial score of 1 or 2 points, depending also on the quality of your model comments; you are invited to the grading session, where you can try and increase your initial score by 1 point into your final score.

If the assistant figures out a minor fix that is needed to make a model run as per our instructions above, then, instead of giving 0 points, the assistant may deduct 1 point at her/his discretion.
Table 1: In each row, a solution to most instances up to the value of $n$ (which is called nGroups in the skeleton model) in the ‘optimality’ column must be found and proven optimal by at least one backend within 300 CPU seconds, and a solution to most instances up to the value of $n$ in the ‘feasibility’ column must be found by at least one backend within 300 CPU seconds: the reference platform is version 2.3.1 of MiniZinc on any Linux computer of the IT department.

<table>
<thead>
<tr>
<th>score</th>
<th>optimality</th>
<th>feasibility</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.5</td>
<td>10</td>
<td>20</td>
</tr>
<tr>
<td>2.5</td>
<td>20</td>
<td>40</td>
</tr>
<tr>
<td>3.5</td>
<td>30</td>
<td>60</td>
</tr>
<tr>
<td>4.5</td>
<td>60</td>
<td>80</td>
</tr>
<tr>
<td>5</td>
<td>80</td>
<td>111</td>
</tr>
</tbody>
</table>