CEUR-WS.org/Vol-2491/abstract30.pdf

An Interactive Knowledge Base Application for Group Assignment

Simon Vandevelde, Kylian Van Dessel, and Herman Crauwels

KU Leuven, De Nayer Campus, Dept. of Computer Science
{s.vandevelde, kylian.vandessel, herman.crauwels}@kuleuven.be

The aim of this work is to research how well the IDP system[1], developed at KU Leuven, functions in an interactive context. The IDP system is an implementation of the knowledge base paradigm (KBP), meaning that it stores its information in a knowledge base, to which it can apply inference methods to solve problems. An advantage of such a system is the reuse of the knowledge base. Using the same information, different inference methods can be used to achieve different goals. Recently, research has been done into the interactivity of the IDP system[2]. To further investigate the system's applicability in an interactive context, we developed an application for group assignment using IDP as backend. The main challenge was ensuring the scalability of this implementation.

The research is divided in two parts. In the first part, we focus on how to implement group assignment using the IDP system. In the second part, we develop an application around this implementation in the IDP system. At KU Leuven Campus De Nayer, each year around 150 students sign up. These students are assigned to groups based on their background and personal preferences, allowing them to feel more comfortable in their new environment. This process is currently done manually, and is not time efficient. Our implementation in the IDP system is subdivided in a coarse initial group assignment and a refined incremental assignment.

The initial group assignment forms groups based on residence and former high school, stimulating students to carpool to campus. Additionally, because students usually prefer to stay with their former classmates, most preferences will already be fulfilled. After starting with a naive implementation, we added improvements to increase its time-efficiency. Some of these improvements are: avoiding symmetry (group layouts that appear different, but are logically the same), calculating the distances between students on the fly instead of inputting a distance matrix, and choosing the optimal precision for the distance.

The incremental assignment recalculates an existing assignment to fulfill specific student preferences. In this way, the assignment can be updated incrementally each time new preferences are received. For each preference, the user decides on a priority, which is then used as a weight for the soft constraints in the optimization problem. For instance, the priority of two students wanting the same group could be lower than the priority of 2 students not wanting to be together.

Copyright © 2019 for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

The IDP implementation minimizes the number of violated preferences, weighted by their priorities.

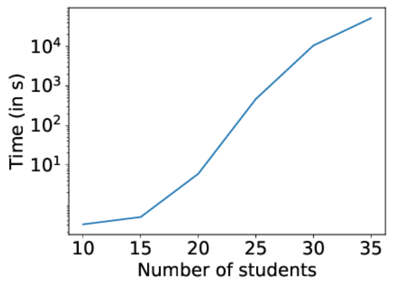

Fig. 1. Duration of initial assignment.

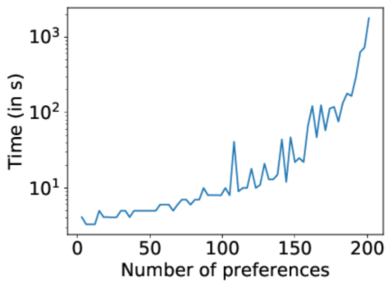

Fig. 2. Duration of incremental assignment.

For the second part, we need to be able to interface with the IDP system. The application uses PyQt5 to construct a Graphical User Interface (GUI) and the Pyidp3 API[3] to communicate with the IDP system. This way, the IDP system controls the intelligence of our application, whilst the GUI provides a user-friendly system. It is important to work interactively, so it is possible to change the system parameters for making a group layout. For example, the desired group size and the number of groups can be altered. For the incremental assignment, it is possible to add preferences, with a priority, to calculate multiple possible solutions and to show the differences between the current and a newly calculated layout.

As shown in fig 1, the initial assignment does not scale well. The implementation relies heavily on mathematical calculation-intensive tasks, by which it loses the advantage of the IDP system. However, this long calculation time can be overcome by limiting the amount of time we allow the IDP system to run. This way the result is not ideal, but it is satisfactory seeing as it is only a starting point to use in the incremental assignment. The incremental assignment on the other hand succeeds in recalculating a group layout with up to 100 preferences within 10 seconds, as shown in figure 2. We prove that in a realistic setting, i.e. considering a realistic amount of preferences, our system provides satisfactory results. The system will be used by the planner at the campus next year.

References

1. Bogaerts, B., Bruynooghe, M., Janssens, G., Denecker, M.: Predicate logic as a modelling language: The IDP system. arXiv.org (2018), http://search.proquest.com/docview/2071739773/
2. Van Hertum, P., Dasseville, I., Janssens, G., Denecker, M.: The KB paradigm and its application to interactive configuration. Theory and Practice of Logic Programming **17**(1), 91–117 (2017), http://search.proquest.com/docview/1853690223/
3. Vennekens, J.: Lowering the learning curve for declarative programming: A python API for the IDP system. vol. 10137, pp. 86–102. Springer Verlag (2017)