Nurse rostering competition 2010: Geoffrey De Smet

Abstract

This paper has a brief description of the Drools Planner framework and the nurse rostering implementation used for the International Nurse Rostering competition 2010.

The Drools Planner framework

Drools Planner (formerly known as Drools Solver) is a meta-heuristic solver and it is being used in production today for all kinds of planning problems. It is business friendly open source software (under the apache license) and part of the JBoss Drools project. Drools Planner has been created and improved by Geoffrey De Smet since 2006. Because Drools Planner focuses on production use, there is an extensive reference manual, many examples, a regularly updated blog and high quality releases.

Drools Planner delegates the score calculation to the powerful Drools rule engine. Each hard and soft constraint is written as a score rule, allowing the developer to focus on the business logic instead of the iteration and hashing logic. The score rules are isolated from each other, making it easy to add, change or remove hard and soft constraints. Yet, the rule engine is very fast and scalable, because it executes all the rules in a single, optimized ReteOO network, with appropriate indexing, node sharing and many more optimizations. Because of the forward-chaining nature of the ReteOO network, the rule engine does automatic delta score calculation, without any need for domain specific code. Writing a score implementation is developer-friendly and highly customizable.

Drools Planner includes several meta-heuristic algorithms, such as hill-climbing, tabu search (solution tabu, property tabu, move tabu) and simulated annealing.

The Nurse Rostering Competition implementation

Every track starts with a deterministic initializer, before switching to a meta-heuristic. The sprint track uses property tabu search, the medium track uses simulated annealing with property tabu and the long track also uses property tabu search (because there was not enough time left to switch it to simulated annealing too). There a several move implementations, such as the employee change move, the employee switch move, the employee sequence switch move, etc.

The entire implementation is object orientated: for example, an Employee class instance has a Contract class instance, a Shift class instance has a ShiftType class instance, …

Future improvements

Future improvements for Drools Planner include better simulated annealing tweaking, solver phasing, constraint based statistics and multi-threading/cloud support.

Meanwhile, the Drools rule engine will become faster with ReteOO compilation, concurrent node evaluation and better forward-chaining accumulates.

Drools Planner links

The Drools Planner website:
http://www.jboss.org/drools/drools-planner.html

The Drools Planner blog:
http://blog.athico.com/search/label/planner

The Drools Planner (AKA Solver) manual:
http://www.jboss.org/drools/documentation.html
The latest Drools Planner manual:

https://hudson.jboss.org/hudson/job/drools/lastSuccessfulBuild/artifact/trunk/target/docs/drools-planner/html_single/index.html

The Drools user mailing list:
http://drools-java-rules-engine.46999.n3.nabble.com/Drools-User-f47000.html

The Drools source code (includes Drools Planner source code):
http://anonsvn.jboss.org/repos/labs/labs/jbossrules/trunk/

Geoffrey's twitter:
http://twitter.com/geoffreydesmet