The Free Life Planner (DRAFT)

Andrew Dougherty
FRDCSA Project
adougher9@gmail.com

Abstract

The Free Life Planner aims to help any interested parties with logistical support for daily living, although it is targeted especially towards the disadvantaged (for instance, those experiencing poverty, illness, disability and/or homelessness). It helps to manage the affairs of daily living, such as activities of daily living (ADLs), executive skills, calendaring, recurrences, reminders, planning, scheduling and execution. It further consists of several integrated planning systems which handle finances, meals, transportation, short and long term life planning, and so forth. It is hoped that a free/libre application for this will reduce the strain on both social safety net systems and individuals using those systems, by improving the quality of planning. It has been demonstrated that poverty causes a marked decrease in executive functioning, and so a freely redistributable and modifiable program that addresses this could help to improve quality of life. It is accessible as an HTML5 web-application via cell-phone, computer or tablet.

Introduction

This paper documents progress within the Formalized Research Database; Cluster, Study and Apply (FRDCSA) project (**?**) towards the development of the flagship Free Life Planner (FLP) application.

A large pipeline for life rule acquisition has been created, which locates via a web search spider materials applicable to the life planner, and catalogs them in a digital library. Rules are mined, such as "Never get involved in a land war in Asia", and "Never go in against a Sicilian when death is on the line", and then translated using a custom tool semi-automatically into Prolog. Using extensions from the PrologMUD project, which contains a Free/Libre clone of Cyc, the FLP is firmly rooted in knowledge-based systems (KBS) and expert systems (ES) technologies. The system runs using a bidirectional interface bewteen Perl and Prolog, with Perl providing a CMS and vast libraries of helper functions, and Prolog handling many of the reasoning tasks. Everything persists using the FreeKBS2 system to a MySQL backend.

For an overview of the FRDCSA Project and the motivation for Free/Libre software development and conglomeration, see (Dougherty 2018).

Motivation for the Free Life Planner

The Free Life Planner extends the FRDCSA thesis into the space of life enhancement. This is desireable in and of itself as well as a means to promote the development of the AI, as it will require a lot of work and being organized and efficient is critical to the success of the endeavor.

The Free Life Planner aims to apply the vast corpus of existing AI technologies to the last-mile delivery of software services that promote social goods and enhance welfare.

The basic intuition is that it is possible to encode rules that in practice govern real life into formats that are executable and machine readable. Moreover, if this process of rule acquisition can be expedited, it would be possible to encode into the rulebase larger volumes of textual reference material such as from books and papers than a person could hope to read and/or retain. Then, when enaging in practical reasoning deliberation, potential actions could be proposed and vetted on the basis of these rules, in this case faster and more precisely than a person could.

Lately machine learning techniques like Deep Learning have been demonstrating superhuman performance in various tasks such as Go and DOTA2, demonstrating the feasibility of algorithmic approaches to complex problems and silencing critics of artificial intelligence. We do not take a view that machine learning approaches are in competition with rule-based systems, but rather in collaboration with them. The FRDCSA umbrella project is inherently multi-strategy and embraces all best-of-breed technologies. For instance, we have experiment with GPGPU-based planning systems.

What are Life Rules?

We take rules here to mean Prolog and PFC facts and rules. So a rule can be a statement of fact that is used to make inferences, such as for instance the nutrition content of a particular food. It can also mean IF-THEN rules, such as . Or it could mean forward chaining rules in PFC.

Here are some sample life rules:

```
done(pageRead(andrewDougherty,
 pageNo(1,publicationFn(morbiniPhdThesis))))
```

Overview of the Free Life Planner Project

FLP consists of several parts; including a responsive (mobile-friendly) web-based front end, an Amazon Alexa

voice interface, and an administrative backend which runs from Emacs. The front-end employs the Perl Catalyst Model/View/Controller (MVC) architecture, in particular, the ShinyCMS Content Management System (CMS). Backend capabilities are provided by a bidirectional Perl to Prolog interface based on Yaswi, a Perl module which achieves this. Furthermore, the UniLang InterLingua and the FreeKBS2 architecture implement a storage mechanism which persists to a MySQL database.

System Architecture

PrologCYC Discipline

Psyclone is our project name for a prolog-based KBS system that mirrors that of PrologCYC, a Free/Libre Prolog reimplementation of CYC, part of the PrologMUD system [1]. Great advantages are conferred upon the FLP by making use of the PrologCYC discipline. This mainly consists of KBS technologies. For instance, it provides naming conventions, microtheories, predicates delineating various properties of other facts, rules and predicates.

Planners

Where possible, different domains are integrated into the input files, and where not possible, we engage in factored planning.

ADL Planner

Financial Planner

The financial planner [2] is based on a simplified planning domain (to be expanded upon in the future) which consists of a promiseToPayFor function.

The financial planner is tailored for use with OPTIC-CLP [3], specifically using both paid and unpaid predicates to track payment status in order to handle the lack of support for negative preconditions. A desireable property of the financial planner is the ability to present a statement of projected transactions. However calculating this could be complex when the simplified domain is extended to handle things like overdue payments. The VAL tool [4] provides the ability to list what the values of functions are after each step. However it is difficult to correlate the information given the way VAL handles temporal domains. So we convert from OPTIC-CLP's temporal plans to an abstracted sequential plan, and then obtain the values for the fluents using VAL's treatment of sequential plans.

We recently demonstrated a 3-month financial forecast, adding features which make it possible now to generate multiyear forecasts.

Given the speed at which plans are computed by the OPTIC-CLP planner, it is possible to engage in contingency planning. Since the promiseToPayFor functions are generated from specifications, it is possible to generate multiple planning problems with slightly different constraints, in order to engage in what-if scenario planning. For instance, individual payments can be moved backwards or forwards in time, eliminated or their amount changed, etc. At present this is done manually, but in the future we have a subsystem in development which makes common modifications, in order to develop contingency plans in order to be prepared for undesirable developments.

Whether a payment is late is not, when one has adequate funds, generally a large issue, however, when cash-flow is critical it can be arbitrarily destructive. This system helps to tell you how long your current funds will last you, what you can afford in the meantime while you wait for more funds, and so on. This kind of information can be very useful in tight financial situations, especially when decisions are under time pressure. While it is generally not that hard to run the numbers on one's own, having the process automated allows rapid recalculation, integration as a financial question-answering into other planners, and helps highlight accounting mistakes.

Meal Planner

The meal planner [5] is intimately connected with the inventory manager. Once inventory levels have been accurately assessed, it is possible to generate a meal plan. The meal planner makes use of the LPG-td 1.0 planner [6]. It is possible to set preferences for the amount of different nutrients, for example:

```
(at end
 (<= (intake saturated_fat grams ?agent)
  20.0))
```

Transportation Planner

Short and Long Term Life Planner

Calendaring

FLP comes with a range of calendaring features, including calendar entries and recurrences.

Acknowledgments

I am grateful especially to Meredith McGhan for pushing forward the development of the Free Life Planner, to Douglas Miles for countless assistance on the Prolog and PrologMUD dimension of the FLP, to Justin Coslor for inspiration and financial assistance, to Jess Balint for his financial assistance and software development assistance, and to my mother for her patience and support as I pursued the multi-year long development process.

References

Dougherty, A. J. 2011. Temporal planning and inferencing for personal task management with SPSE2.

Dougherty, A. J. 2018. The FRDCSA project.

[1] https://github.com/TeamSPoon/prologmud

[2] https://github.com/aindilis/financial-planner

[3] https://nms.kcl.ac.uk/planning/software/optic.html

[4] https://nms.kcl.ac.uk/planning/software/val.html

[5] https://frdcsa.org/ andrewdo/WebWiki/MealPlannerUpdate.html

[6] http://zeus.ing.unibs.it/lpg/

Gerevini, A.; Saetti, A.; and Serina, I. 2004. LPG-TD: a fully automated planner for PDDL2.2 domains. *14th Int. Conference on Automated Planning and Scheduling (ICAPS-04)*.

Howey, R., and Long, D. 2003. VAL's progress: The automatic validation tool for pddl2.1 used in the international planning competition journal. *in Proceedings of the ICAPS 2003 workshop on "The Competition: Impact, Organization, Evaluation, Benchmarks"*.