Funding Proposal

Planet: A concurrent Prolog for .NET

Jonathan Cook and Stephen Gilmore

Laboratory for Foundations of Computer Science
The University of Edinburgh
July 2003

1 Introduction

This document is a proposal for funding to support further research into an existing extended Prolog implementation developed for the .NET platform in the Laboratory for Foundations of Computer Science at The University of Edinburgh.

We have developed a new Prolog implementation for .NET, called Planet [1], which is implemented as a set of C# classes. Planet operates by compiling Prolog source code into C# source code. New built-in predicates were added to enable the user to exploit the concurrency support in C# from the Prolog side. We now seek funding for one year of work on engineering extensions to the existing compiler infrastructure. This will include optimising the translated code by detecting common programming idioms; allowing the programmer to add mode annotations to their code and improving integration with .NET and the CLR.

Translating Prolog to C# source code provides one way of using Prolog within the .NET Framework. By translating to MSIL via C# we can exploit the C# compiler's ability to produce well optimised MSIL. It is also possible to exploit language features of C#, in particular its rich graphical, networking and other libraries, by building equivalent features in Prolog.

A technical introduction to Planet can be found in [2].

Examples of the use of Planet: As an example of the use of Planet we have implemented a system of disconnected collaborative agents allowing a number of users to asynchronously assert facts to a database, then to disconnect and alter their private copies of the database. On reconnection their private copies are synchronised with the central database by searching for conflicts between their copies and the central database.

Two other examples include an object-oriented browser application and a class hierarchy viewer, combining C# front-ends with Planet back-ends.

Relation to other logic programming language implementations: The .NET platform offers an unparallelled opportunity to build systems based on a number of interoperating languages. Logic programming is currently underrepresented on the .NET platform, with Mercury a notable exception. Prolog is the seminal logic programming language, and Planet is an efficient and reliable Prolog implementation for .NET useful in particular to programmers who wish to use legacy Prolog code which cannot be easily ported to Mercury. Planet supports many logic programming idioms which are not supported by Mercury.

A number of Prolog implementations are available under Windows. For example, Jinni 2002 is described as "a high performance Java and .NET based Object Oriented Prolog compiler". Planet has the advantage over Jinni 2002 that it depends only on C# and not Java. Visual Prolog is a traditional Prolog for Windows, and unlike Planet is not .NET or Windows XP friendly.

2 Intended Work

Work package 1: (Four months) The current compilation scheme leaves open the possibility of compiling a predicate, and all predicates deeper than it in the tree, into *more idiomatic and more efficient C#*. In particular tail recursive predicates which involve no cuts could be compiled into `while` loops. Making the generated code more idiomatic should ensure that we are working with the C# compiler's optimiser rather than against it.

Work package 2: (Four months) Related projects, such as Mercury, support *mode declarations*. A mode declaration provides extra information to the compiler regarding which arguments of a predicate are input arguments and which are output arguments. This information can be optional, only being used to provide more efficient compiled code when some mode declarations have been provided.

Work package 3: (Four months) Other engineering improvements to the Planet infrastructure would improve its ease-of-use and accessibility. These would include a *Visual Studio.NET plug-in for Planet* and other *attribute and metadata enhancements* of the generated C# code.

3 Funding request

We request twelve months of funding for Jon Cook to implement the above extensions to Planet. Computing equipment and infrastructure costs would be met by the Laboratory for Foundations of Computer Science. The level of stipend requested would be the entry point on the national AR1B scale of £18,265 per annum.

References

[1] Planet home page: `http://homepages.inf.ed.ac.uk/stg/research/PLANET/`. Compiler page: `http://www.lfcs.ed.ac.uk/psharp/`. (The Planet compiler is available for download from this page. Pre-release versions of Planet were distributed under the name P# and the current version of the Planet compiler still uses this name.)

[2] Planet: A concurrent Prolog for the .NET framework. Jon Cook. Submitted for publication in *Software Practice and Experience*, 2003.

Contact Details

Jonathan Cook	Stephen Gilmore
The University of Edinburgh	The University of Edinburgh
Edinburgh EH9 3JZ	Edinburgh EH9 3JZ
Scotland	Scotland
Phone: +44 (0)131 650 7354	Phone: +44 (0)131 650 5189
Email: `Jon.Cook@ed.ac.uk`	Email: `Stephen.Gilmore@ed.ac.uk`