Msc Projects 2004-2005

Dr. Marina De Vos
mdv@cs.bath.ac.uk

Project Proposals in Answer Set Programming

Project 1

Name: Marina De Vos

E-mail: mdv@cs.bath.ac.uk

Title: AN IDE for ASP

Description: ASP (Answer Set Programming) is an up and coming field in logic programming. Programs consist of a series of logical rules which are then put into a solver, which produces a list of 'possible world views' that are consistent with the information provided by the rules. It has a lot of potential for tasks such a knowledge representation, scheduling, planning, data mining, etc. At the moment a number of solvers exist but there are very few support tools to help programmers. This project would involve creating an integrated development environment for answer set programming. It is envisaged that this would be done by combining existing tools (most of which are command line based) with a suitable GUI. For example, integrating Smodels, lparse, noMoRe, DLT, DLV and IDEAS would give solvers, visualisation and debugging. The application would have to be able to be run under GNU/Linux and ideally under windows as well. As well as the practical issues there are a number of interesting research questions such as how to minimise cognitive load when working with large amounts of highly structured data and how to ensure consistence between what is modelled by each atom in the system.

Required Skills: Knowledge of GUI programming on GNU/Linux and (ideally) windows, familiarity with a variety of different approaches to development environments and support tools. Some understanding of HCI issues as relating to programming would be useful.

Project 2

Name: Marina De Vos

E-mail: mdv@cs.bath.ac.uk

Title: OCLP-Agents

Description: Logic Programming is a growing research field in artificial intelligence. With the introduction of answer set programming, logic programming has taken a new step in finding new application areas and efficient algorithms for solving represented problems. A good example is the use of answer set programming for the propulsion of the NASA space shuttles.

Ordered Choice Logic Programming (OCLP) is a recent logic programming formalism designed to model decision-making in an elegant and intuitive way such that the solutions of the represented problem can be obtained as the models of the logic program. The programs are equipped with two semantics: the skeptical and credulous answer set semantics.

OCLP can be used for the representation of an agents reasoning capabilities. A system of such agents is called a logic programming agents system (LPAS). The goal of this project is to implement a framework for such multi-agent systems.

Recently we have developed a program, called oct, that is capable of producing both semantics of the input program. This is done by transforming the program to a standard logic programming which can then be handled by an answer set solver. Unless you want to write your own code for it, you can use oct during your development.

Functional requirements: The program should:

- allow the user to specify the agents and the communication channels
- allow the user to specify the type semantics used by the agents
- support at least a reduced version of OCLP
- computes the requested semantics
- provide, if requested, a trace of the communication necessary to establish an agreement

Non-functional requirements: Documents should be written using latex.

Required Skills: A basic knowledge of mathematics. This includes basic set theory, partial ordered sets and the logical operators.

Indicative reading: For more information, have a look at the examples in research papers mentioned at "http::/www.cs.bath.ac.uk/∼mdv/publications.html". You can download the oct-software from "http::/www.cs.bath.ac.uk/∼mdv/oct/'

Project 3

Name: Marina De Vos

E-mail: mdv@cs.bath.ac.uk

Title: Finite State Autonoma Generation

Description:

When creating "intelligent systems" there are a variety of approaches that can be used. Finite State Autonoma are very fast and efficient to implement but are difficult and unintuitive to design and often difficult to prove. Logic programming gives a very simple and intuitive way of designing of creating sophisticated 'intelligent' systems but are often too slow to be used in time critical applications, such as games.

This project seeks to combine the advantages of both by generating finite state autonoma from logic programs. A logic program is developed that will plan one step of a solution, it is evaluated with respect to all possible states of the world and this used to build a planning graph with nodes corresponding to actions and links to changes in the state of the world, from the graph a finite state autonoma is then created.

The first step of the project would be to create tool to automate the process of creating autonoma from logic programs. Possible extensions are taking results on adding rules to logic programs and using them to update the finite state autonoma, creating a 'learning' system or to compare the performance of the tool against similar systems, such as qsmodels - a Quake 3 bot implemented using logic programming.

Required Skills: An interest in logic programming and artificial intelligence and general computer science.

Project 4

Name: Marina De Vos

E-mail: mdv@cs.bath.ac.uk

Title: Incremental Propositional OCLP

Description: Logic Programming is a growing research field in artificial intelligence. With the introduction of answer set programming, logic programming has taken a new step in finding new application areas and efficient algorithms for solving represented problems. A good example is the use of answer set programming for the propulsion of the NASA space shuttles.

Ordered Choice Logic Programming (OCLP) is a recent logic programming formalism designed to model decision-making in an elegant and intuitive way such that the solutions of the represented problem can be obtained as the models of the logic program. The programs are equipped with two semantics: the skeptical and credulous answer set semantics.

Recently we have developed a program, called IASAI, that is capable of producing of computing the answer sets of a program incrementally. Whenever the program changes, the solver will use the previous answer sets to compute the new ones.

The goal of this project is to extend IASAI to ordered choice logic programs.

Required Skills: A basic knowledge of mathematics. This includes basic set theory, partial ordered sets and the logical operators. An interest in developing and implementing novel algorithms.

Project 5

Name: Marina De Vos

E-mail: mdv@cs.bath.ac.uk

Title: Compute Time Estimator for ASP

Description: AnsProlog is a declarative logic programming system that is used at the University of Bath. Programs are made up of a series of rules, each of which expresses a simple statement about logical inference. There are no ways of estimating roughly how much compute time a program will need to produce answers - it can vary, largely independently of the size of the program. This projects aims at developing a rough heuristic for how long the program will take to compute. This can either be an absolute measure or relative to another program. In the case of a relative metric it may be useful to develop some common 'benchmarks' to check against. Implementing a tool that could assess this metric would be a useful extension.

Required Skills:A solid understanding of algorithms and what are 'easy' and 'hard' tasks for computers.

Project 6

Name: Marina De Vos

E-mail: mdv@cs.bath.ac.uk

Title: Precomputing branching strategies for Logic Programs

Description: AnsProlog, a declarative logic programming language used at the University of Bath works by computing what possible world views are consistent with the input program. The algorithms that compute these 'answer sets' all work in a similar fashion - they extend the set of information that is known as far as possible using inference rules, then they pick one of the unknown atoms and branch. One branch assume it to be true, the other assumes it to be false. Picking which atom to branch on makes a very significant difference to the performance of a system and currently a number of 'local' heuristics are used. These consider the short term effect of branching on each algorithm but do not consider the 'whole picture'.

This project aims to devise (and if possible implement) and algorithm that will take a logic program and produce an ordered list of which atoms it is best to branch on and in which order this should be done. This should help reduce the amount of compute time required to compute the answer sets of this program.

Required Skills: An understanding of graphs and basic logic.

Project 7

Name: Marina De Vos

E-mail: mdv@cs.bath.ac.uk

Title: An open implementation of DLT

Description: DLT is a system that allows a programmer to create templates for commonly used concepts in declarative programming languages. This saves having to re-implement common ideas such as sets, lists, etc. DLT is a closed source product that is only compatible with a very limited range of tools. This projects is to re-implement the functionality of DLT (matching the syntax if possible) such that it can be used with a range of tools and can be released under an open source licence. It is suggested that either perl or the C preprocessor is used to handle macro expansion.

Required Skills: An understanding of macros in C/C++ or some basic knowledge of perl and regular expressions.

More Applied Projects

Project 8

Name: Marina De Vos

E-mail: mdv@cs.bath.ac.uk

Title: Call Graphing in C

Description: Understanding large and often complex programs in a short space of time is a very common task. It is difficult not because of the lack of information - the source code contains everything that is needed, the key problem is accessibility. It is very difficult to get an overview of how a program works by simply reading sections of the source code. Tools such as LXR help as they make it easier to read the code but there is still room for an whole range of tools that summarise the information in the source code and display it in a useful format. This project aims to implement a prototype of such a tool. The first step is to create a graph from a series of in files containing C, with nodes representing functions and directed links representing the 'is called by' relation and to be able to output the graph in some pictorial form (a tool such as DaVinci is recommended for this). Extensions could be adding different types of arrow for 'may be called', 'definitely called', 'called repeatedly', etc. adding outline of control flow structures to the nodes so how the functions are called is easier to see, tracing which variables are passed between functions, handling callbacks and handling other languages.

Required Skills: An understanding of procedural programming and an eye for user interface design.

Project 9

Name: Marina De Vos

E-mail: mdv@cs.bath.ac.uk

Title: Customised Wiki

Description: One of my research projects needs a webpage that can be updated by all members of the consortium. The Wiki that is currently used does not support all required features. For this project I would a customised Wiki to be developed.

Required Skills: no specific skills required.

Project 10

Name: Marina De Vos

E-mail: mdv@cs.bath.ac.uk

Title: E-learning Site for @lis

Description: Bath is currently involved in a European Project on multi-agent systems (MAS). One of the core deliverables is a e-learning site for MAS. The current system allows only for multiple choice questions and has some other disadvantages. The aim of this project is create a new e-learning system that is suitable as an e-learning tool for MAS in a undergraduate and post-graduate context.

Required Skills: An interest in e-learning and the development of learning tools and webpages.

Project 11

Name: Marina De Vos

E-mail: mdv@cs.bath.ac.uk

Title: SSH Key Distribution

Description: ecSH or SSH is the most widely used technology for remote access and administration on Linux and UNIX systems. It uses public key cryptography to authenticate the remote server and symmetric key cryptography to protect against eavesdropping and modification. The one key weakness in the system is the lack of a widely used system to distribute the keys of SSH servers, allowing for 'man in the middle' style attacks. A number of solutions have been proposed but none have been implemented widely. This projects aims to create a simple series of scripts that allow a group of users, possibly in different administrative domains to publish and share keys and verification information - all of which are then available via an SSH session to a central server. Users can then use their own criteria for deciding whether they trust the key(s) they receive from the system (i.e. trusting that the person running the central server as sufficiently verified all users, checking GPG signatures supplied with each key, etc.).

Required Skills: Understanding of the basics of a UNIX system and cryptographic authentication.

General Areas of Interest

If you have a project proposal in one of the areas below, you can always send me email (mdv@cs.bath.ac.uk) or come to see me (1West 2.28).

- Logic Programming
- Automated Reasoning
- Artificial Intelligence
- Genetic Algorithms
- Software Frameworks
- Design Patterns