The Application of Deductive Synthesis Techniques to the Rapid Assembly and Re-Assembly of Grid Applications

Case for Support

Alan Bundy Alan Smaill

September 16, 2002

1 Previous Track Record

The applicants are members of the Mathematical Reasoning Group (MRG), which is part of the Centre for Intelligent Systems and their Applications within the Division of Informatics at the University of Edinburgh. The Division of Informatics was one of only six computing departments in UK to have obtained a 5* ranking in the 2001 Research Assessment Exercise. It returned the highest number of research active staff and was the only 5*A department. It contains world-class research groups in the areas of theoretical computer science, artificial intelligence and cognitive science.

1.1 The Mathematical Reasoning Group

Since the mid 1970s, the MRG has been engaged on the computational analysis, development and application of mathematical reasoning processes and their interactions. Its work is characterised by its unique blend of computational theory with artificial intelligence (http://dream.dai.ed.ac.uk/). The processes studied by MRG have included theorem proving via proof methods, proof patching, analogy, symmetry, abstraction, diagrams and reflection; the learning of new proof methods, the formalisation of informally stated problems, the formation of concepts and conjectures and the interaction of automated systems with human users. The applications of MRG's work have been to: proof by mathematical induction and co-induction; analysis, including non-standard analysis; mechanics problems; the building of ecological models; the synthesis, verification, transformation and editing of both hardware and software, including logic, functional and iterative programs and process algebras; the configuration of hardware; game playing; and cognitive modelling. Since the mid 1980s, it has produced over 300 publications, trained 17 research fellows and 41 PhD students, two of whom have won the BCS/CPHC Distinguished Dissertations Award. It has been supported by over 40 research grants, including an EPSRC rolling funding grant (1982-2002), and it has just been awarded an EPSRC platform grant.

1.2 Prof. Alan Bundy

Prof. Bundy has been active in automated mathematical reasoning research since 1971 and has become a world authority. His international reputation is witnessed by his being made a founding fellow of both of the two international AI societies: AAAI and ECCAI, in addition to the UK society, AISB, and serving terms as Chair of both IJCAI Inc and CADE Inc. He is also a Fellow of the Royal Society of Edinburgh and of the British Computer Society. He won the SPL Insight Award in 1986, was an SERC Senior Fellow (1987-92), a member of the Hewlett-Packard Research Board (1989-91), Head of the Division of Informatics at Edinburgh (1998-2001), a member of the ITEC Foresight Panel (1994-96), a member of the Computer Science RAE panel (1999-2001) and is the founding Convener of UKCRC (2000-date). He is the author of over 140 publications.

1.3 Dr. Alan Smaill

Dr. Smaill has worked in the area of Automated Reasoning since 1986. He holds a D.Phil. from the University of Oxford in Mathematical Logic, and is currently a lecturer in the Division of Informatics in the University of Edinburgh. His research work centres around reasoning in higher-order and constructive logics, with applications in program construction and automated software engineering, and he has published widely in this area. He has been a grant-holder on an EPSRC rolling grant, an ESPRIT project on Logical Frameworks and is currently a PI on EPSRC grant on the mechanisation of first-order temporal logic via embedding into a higher-order representation.

1.4 Mr. Bin Yang

Mr. Bin Yang is currently an MSc student in Advanced Computer Science at the University of Manchester, Computer Science Department, where he has specialised in Artificial Intelligence and its applications. His MSc project is expected to lead to a research publication. His interest in automated mathematical reasoning was encouraged by Prof. Peter Aczel at Manchester. Prior to this, he obtained a B.Eng degree from Shanghai Institute of Electric Power, winning the Best Graduates Award for 2000 (awarded to only 10 of 500 graduates). During each of his four years at Shanghai (1996-2000) he won the Annual Assessment Prize and in 1998 he won the Leifeng Award for outstanding social activity as Class President and Secretary of the University Associates Committee.

2 Description of Proposed Research and its Context

This project will explore the application of deductive synthesis techniques to the construction and maintenance of eScience systems.

2.1 Background

The recent EPSRC report "Computer Science Challenges to Emerge from eScience" identifies the "rapid customised assembly of services" and "autonomic computing" as two of the major challenges that must be solved if the dream of the Grid is to become a reality. We propose an exploratory investigation to discover whether techniques of deductive synthesis can address these two challenges.

2.1.1 eScience and the Grid

We take eScience to be the application of high-performance computing and high-bandwidth communications to the automatic processing and interpretation of the massive data sets arising from data-intensive scientific experiments, such as the Large Hadron Collider, the Human Genome Project and Earth-monitoring satellites. Many sciences are becoming data-intensive, so progress in eScience is an essential ingredient in further scientific advance.

We take the Grid to be a globally distributed, hetereogenous collection of data-sources and data-processors, which can be combined in a wide variety of ways to provide customised knowledge retrieval and processing for scientists. We will call these customised systems, *Grid applications*, and their atomic components, *Grid services.*

The EPSRC report "Computer Science Challenges to Emerge from eScience" identifies many fundamental computing research problems that require solution if the full potential of eScience is to be realised. It also recognises that nearly all areas of computing research have the potential to contribute to these solutions. In particular, it identifies the problems of rapidly assembling a dependable, high quality Grid application from the available Grid services. It also identifies the related problem of automatic recovery from the failure of a Grid application, which it entitles *autonomic computing.* In a globally distributed system with a large number of heterogeneous components, such failures will occur frequently, whether due to the failure of a component Grid service or of the communications between them. In order to provide a dependable service, automatic recovery is required, *e.g.* by re-routing the communications or replacing a failed component by an equivalent one.

There are already several systems that support (re-)assembly of Grid and web services, for instance, the ICENI system [Furmento, 2001]. ICENI provides a common intermediate meta-data language, CXML, for describing Grid services and their performance. This language facilitates modularity by supporting the separation of component interfaces from their implementations. ICENI includes an Application Mapper, which uses performance modelling to find the optimal run-time combination of services. A visual interface assists users to assemble and reassemble applications from Grid services. Our proposal would build on systems like ICENI by providing a theoretical account based on deductive synthesis and, hence, increasing both dependability and automation.

DAML+OIL (http://www.daml.org/2001/03/reference.html) has been recommended by the Bio-Ontologies Consortium as the language of choice for the exchange of information objects in the life sciences (http://www.cs.man.ac.uk/ stevensr/boc/DAMLPlusOIL.htm) and the Gene Ontology Consortium Global Open Biological Ontologies initiative (GOBO) has specified it as one of only two approved implementation languages for its ontologies (http://www.geneontology.org/doc/gobo.html). DAML+OIL is not, by itself, sufficiently rich to describe programs, but it will provide a platform on which we can build a logic that is sufficient for deductive synthesis, *e.g.* either by extending the DAML+OIL or by translation into a richer logic. This head start makes Bioinformatics an attractive domain in which to initiate our investigations.

The work of Sheila McIlraith and collaborators at Stanford is relevant to the proposed research (see *e.g.* [McIlraith *et al*, 2001, McIlraith & Son, 2002]). This work associates a declarative description of a web service with that service via a markup language that is a dialect of DAML. These specifications describe generic services that need to be customised for particular applications; this customisation involves satisfying user constraints associated with the services. More precisely, planning techniques are used to construct the customised services that can achieve the user's goal, in this case using automated first-order reasoning based on the situation calculus.

2.1.2 Deductive Synthesis

Automated reasoning techniques can be applied to the synthesis of IT systems. For instance, in Type Theory a technique called *proofs as programs* extracts a synthesised program from a constructive proof that its specifications can be met. Roughly speaking, an automated reasoning system is required to prove a conjecture of the form:

$$\forall inputs.\exists output.\ spec(inputs, output) \tag{1}$$

where *spec* is the specification of a relationship between the *inputs* and the *output*. For instance, if the desired program was a data sorting algorithm then $spec(input, output)$ might be:

$$permutation(input, output) \wedge sorted(output)$$

This technique is equally applicable to hardware and to hybrid IT systems. *Our hypothesis is that deductive synthesis is applicable to Grid application assembly and re-assembly, provided that Grid services and applications can be specified in a suitable logic.*

Note that conjecture (1) asserts the existence of an *output* for each combination of *inputs*. Provided that the logic is constructive, *i.e.* excludes pure existence theorems, then the proof of (1) will implicitly define the construction of this *output* from the *inputs*. Type Theory makes this construction explicit by associating a partial program with each formula of the logic and program combination operations with each inference step, e.g. $prog : spec(inputs, output)$, where *prog* is the program associated with $spec(inputs, output)$. Thus the desired program *prog* is constructed as a side-effect of the proof process and is guaranteed to meet the original specification, *i.e.*

$$\forall inputs.\ spec(inputs, prog(inputs))$$

Proving synthesis conjectures, such as (1), is a challenging task for automated reasoning. Most Type Theory based synthesis systems provide facilities for human users to guide the search for a proof to augment the automatic techniques. The MRG has pioneered techniques for automating synthesis proofs [Kraan *et al*, 1996, Armando *et al*, 1998, Lacey *et al*, 2000]. Using proof planning, we have extended the state of the art on what can be automatically synthesised.

Note that the above deductive synthesis techniques can be used both to synthesise an IT system from scratch or to re-synthesise a system, *e.g.* with a revised specification or theory. In the latter case, the synthesis proof of the original system may be used to guide the search of the new system: if the specification or theory is only slightly changed then large parts of the new proof may be identical or very similar to the old one. The MRG has also worked on the use of analogy to guide proof search [Owen, 1990, Melis & Whittle, 1998]. Note that, even if the original synthesis proof had required human interaction, the re-synthesis process might be completely automated via the use of analogy, since many of the previously hard steps may now be simply transcribed to the new proof.

An example of a small theory change in a Grid problem would be the deletion of a Grid service or communication link. The automatic re-assembly of a previously successful Grid application can be achieved by deductive synthesis in a revised theory, guided by analogy with the original synthesis proof.

Isabelle is a generic theorem prover, built by Paulson at Cambridge [Paulson, 1986], which supports deductive synthesis. The MRG has considerable experience in the use of Isabelle, including a project to implement proof planning in Isabelle, GR/N37414.

2.1.3 Proof Planning and Analogy

A *proof method* is the computational representation of a common pattern of proof in a family of related proofs [Bundy, 1991]. A *proof critic* similarly represents a common technique for patching an initially failed proof attempt [Ireland & Bundy, 1996]. Proof planning is the use of AI plan formation technology to guide proof search by constraining that search to a set of proof methods and critics. Proof planning limits the combinatorial explosion of potential proof steps, which occurs if exhaustive search techniques are used.

The MRG invented the technique of proof planning, implemented it in the λ*Clam* proof planner [Richardson *et al*, 1998] and applied it particularly to the kind of inductive proof that arise in verification and synthesis of IT systems. It has extended the range of problems that can be solved without human intervention. In particular, the use of proof critics has automated the discovery of intermediate lemmas and generalisations [Ireland & Bundy, 1996] – so called, "eureka" steps, which were previously thought to require human intervention.

Analogy is used in automated reasoning to guide the proof search of a conjecture with the aid of an existing proof of a similar theorem. A flexible mapping is made between symbols of the old theorem and the new conjecture. This mapping is applied to the old proof to suggest steps of the new proof. Any missing steps are then filled-in and any necessary corrections are made to the suggested steps [Owen, 1990]. Analogy has been used in this way to re-verify previously verified programs [Reif & Stenzel, 1993] and the use of re-synthesis for system maintenance is also being explored [Schairer & Hutter, 2002].

2.2 Project Description

The aim of this project is:

> *to evaluate the applicability of deductive synthesis to the problems of rapid customised assembly and automated recovery from failure of Grid applications.*

2.2.1 Objectives

To realise this aim we will need to achieve the following objectives:

- Develop a logic suitable for representing the implementation of a Grid application in terms of its component Grid services.
- Extend this logic so that it is suitable both to specify Grid services and applications and to synthesise Grid applications from Grid services.
- Apply deductive synthesis techniques to the (re-)assembly of Grid applications from Grid services.
- Explore the automation of (re-)assembly using proof planning, analogy and such other proof search techniques as may prove applicable.
- Construct an interface that will assist users to assemble Grid applications.
- Evaluate the applicability of deductive synthesis to the (re-)assembly of Grid applications.

2.2.2 Workplan

Below we present the objectives of §2.2.1 as work-packages with deliverables and timescales.

1. **Formalisation of Grid Applications.** A formal language will be developed in which the structure of Grid applications can be represented as terms. This will be developed via and tested on various Grid applications drawn from NeSC experience.

   Deliverable: formal language.

   Timescale: 4 person months.

2. **Specification of Grid Applications and Services.** A formal language will be developed for the specification of Grid applications and their components. This will be developed via and tested on various Grid applications. This language and that of point 1 will be developed in parallel and integrated into a Type Theoretic language in which the expression $sys : spec$ will assert that sys is a system meeting specification $spec$.

   Deliverable: formal language.

   Timescale: 6 person months.

3. **(Re-)Assembly via Deductive Synthesis.** A sequent calculus inference system will be developed for the language in point 2 and implemented in the generic theorem prover Isabelle. It will be tested by interactive derivation and synthesis of various Grid applications from their specifications.

   Deliverable: interactive synthesis system, conference paper.

   Timescale: 6 person months.

4. **Semi-Automation of Assembly.** Proof plans and critics will be developed for guiding the inference system described in point 3 and implemented in the proof planner $\lambda Clam$. It will be tested by semi-automated derivation of various Grid applications. The tests will include variants in which some components and/or their interconnections are unavailable and an alternative application must be synthesised to meet the same specification.

   Deliverable: semi-automatic synthesis system, conference paper.

   Timescale: 8 person months.

5. **Assembly Interface.** A GUI interface will be developed for the semi-automatic system described in point 4

   Deliverable: GUI interface, manual.

   Timescale: 6 person months.

6. **Automated Re-Assembly.** An analogical proof guidance system will be developed and tested on the problem of re-assembling a damaged Grid application. The synthesis proof of the original application will be used to guide the re-synthesis. Note that this work-package is an optional extension and will only be completed within the grant period if progress on the other work-packages is more rapid than anticipated.

   Deliverable: automatic synthesis system, conference paper.

   Timescale: 6 person months.

7. **Evaluation.** The system described in points 4 and 5 will be evaluated by a series of experiments in which a new set of Grid applications will be specified and synthesised. The results will be analysed to assess the applicability of deductive synthesis to rapid customisation of Grid applications.

   Deliverable: experimental results, journal paper.

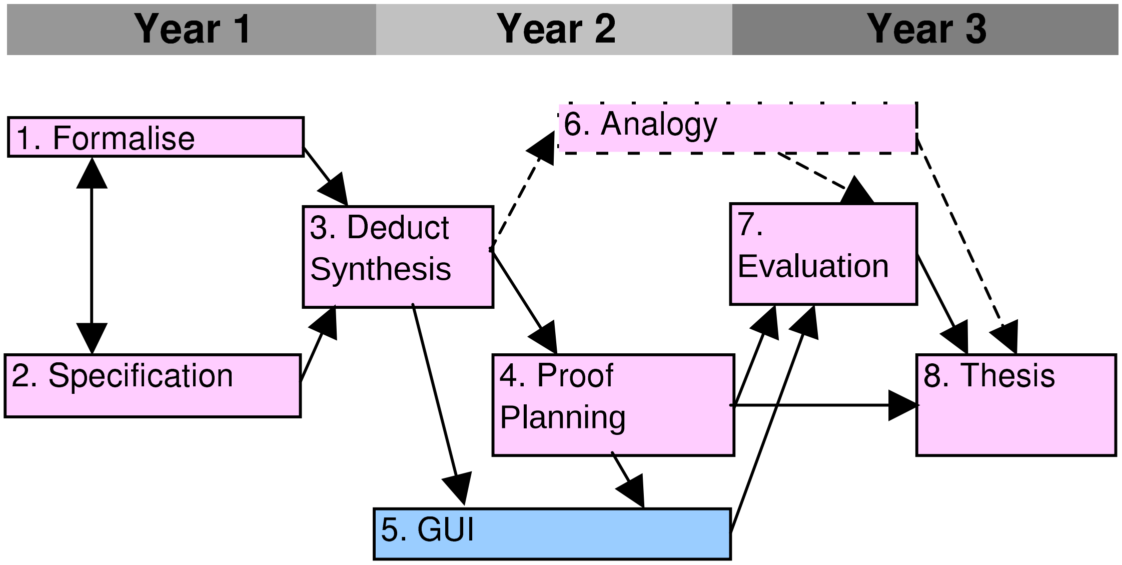
   Timescale: 6 person months.

8. **Dissertation Writing.** All the above work will be described in a PhD dissertation.

   Deliverable: PhD dissertation

   Timescale: 6 person months.

The scheduling of and dependencies between these work-packages is represented in figure 1.



Each box represents one of the work-packages from §2.2.2. Each arrow represents a dependency between two work-packages. The position of each work-package on the x-axis shows when it will be executed, its area shows its duration and its colour indicates who will execute it (student: pink, CO: blue, dotted: optional).

Figure 1: Diagrammatic Workplan

2.3 Relevance to Beneficiaries

This work will initially be of interest to our colleagues working on techniques for (re-)assembly of Grid applications. If it is successful then it will be of long-term benefit to eScientists by assisting them in the rapid customised assembly of Grid applications and enabling their Grid applications to automatically recover from failure of components and their interconnections.

2.4 Dissemination and Exploitation

Dissemination will be via conference and journal papers, presentations and the free availability of prototype systems. All deliverables will be available on the web as well as via traditional publication routes. We will work closely with the National eScience Centre in Edinburgh, Carole Goble's MyGrid and Ian Horrock's WonderWeb projects at the University of Manchester and John Darlington's ICENI project at Imperial College London, to ensure the relevance and dissemination of our work.

2.5 Justification of Resources

- A project student will be required to implement the work-packages described in §2.2.2 except nos. 5 and 6 (unless time is available for no. 6). A suitable candidate, Mr. Bin Yang, is available to undertake this work.
- 20% of a computer officer is required to provide general support to the project and to execute work-package no. 5. A well-qualified candidate is available: Mr. Gordon Reid. Mr. Reid already has considerable experience with the development of GUIs, including relevant expertise, *e.g.* in TCL/TK.
- 10% of a secretary is required for general administrative support, including the organisation of travel, liaison with collaborators, dissemination of our publications, maintenance of our web pages, etc. An experienced candidate, Mrs. Carole Douglas, is available to undertake this work.
- A workstation is required for the project student. Consumables for the project staffs' use of group equipment are also required.
- Travel expenses are required to attend appropriate conferences and workshops, and to visit those doing related work elsewhere.

References

[Armando *et al*, 1998] Armando, A., Gallagher, J., Smaill, A. and Bundy, A. (1998). Automating the synthesis of decision procedures in a constructive metatheory. *Annals of Mathematics and Artificial Intelligence*, 22(3–4):259–279. also available as Research Paper no. 934, DAI, University of Edinburgh.

[Bundy, 1991] Bundy, Alan. (1991). A science of reasoning. In Lassez, J.-L. and Plotkin, G., (eds.), *Computational Logic: Essays in Honor of Alan Robinson*, pages 178–198. MIT Press. Also available from Edinburgh as DAI Research Paper 445.

[Furmento, 2001] (2001). *Optimisation of Component-based Applications within a Grid Environment.*

[Ireland & Bundy, 1996] Ireland, A. and Bundy, A. (1996). Productive use of failure in inductive proof. *Journal of Automated Reasoning*, 16(1–2):79–111. Also available from Edinburgh as DAI Research Paper No 716.

[Kraan *et al*, 1996] Kraan, I., Basin, D. and Bundy, A. (1996). Middle-out reasoning for synthesis and induction. *Journal of Automated Reasoning*, 16(1–2):113–145. Also available from Edinburgh as DAI Research Paper 729.

[Lacey *et al*, 2000] Lacey, D., Richardson, J. D. C. and Smaill, A. (2000). Logic program synthesis in a higher order setting. In Lloyd, J., Dahl, V., Furbach, U. and Stuckey, P.J., (eds.), *Computational Logic (CL2000, London, UK)*, volume 1861 of *Lecture Notes in Computer Science*, pages 87–100. Springer Verlag.

[McIlraith & Son, 2002] McIlraith, S. and Son, T. C. (2002). Adapting Golog for composition of semantic web services. In Fensel, D., McGuinness, D. and William, M.-A., (eds.), *Proceedings of the Eighth International Conference on Knowledge Representation and Reasoning (KR2002)*. Morgan Kaufmann.

[McIlraith *et al*, 2001] McIlraith, S., Son, T.C. and Zeng, H. (2001). Semantic web services. *IEEE Intelligent Systems*, 16(2):46–53. Special Issue on the Semantic Web.

[Melis & Whittle, 1998] Melis, E. and Whittle, J. (1998). Analogy in inductive theorem proving. *Journal of Automated Reasoning*, 22(2).

[Owen, 1990] Owen, S. (1990). *Analogy for Automated Reasoning*. Academic Press Ltd.

[Paulson, 1986] Paulson, L.C. (1986). Natural deduction as higher order resolution. *Journal of Logic Programming*, 3:237–258.

[Reif & Stenzel, 1993] Reif, Wolfgang and Stenzel, Kurt. (1993). Reuse of proofs in software verification. In Shyamasundar, R., (ed.), *Foundation of Software Technology and Theoretical Computer Science*, volume 761 of *Lecture Notes in Computer Science*. Springer.

[Richardson *et al*, 1998] Richardson, J. D. C, Smaill, A. and Green, I. (July 1998). System description: proof planning in higher-order logic with Lambda-Clam. In Kirchner, Claude and Kirchner, Hélène, (eds.), *15th International Conference on Automated Deduction*, volume 1421 of *Lecture Notes in Artificial Intelligence*, pages 129–133, Lindau, Germany.

[Schairer & Hutter, 2002] Schairer, Axel and Hutter, Dieter. (2002). Proof transformations for evolutionary formal software development. In *Proceedings of the 9th International Conference on Algebraic Methodology And Software Technology (AMAST 2002)*. Accepted.