A Drosophila for Computational Dialectics

Simon Wells
Division of Applied Computing
University of Dundee, swells@computing.dundee.ac.uk

Chris Reed
Division of Applied Computing
University of Dundee, chris@computing.dundee.ac.uk

1 Introduction

Drosophila melanogaster is a fruit fly used in the biological sciences as a model organism for research [2]. This model organism is used as a base line against which to test new theories. Herbert Simon [17], and later John McCarthy [12] refer metaphorically to the game of Chess as a "*drosophila*" for AI. In a similar vein McCarthy also proposes the missionaries and cannibals problem as a drosophila for problems in logical AI [11] . The suggestion is that certain classes of problems can be used to quantify progress in the field of AI overall and to demonstrate individual theories within the field. Our research uses the concept of a drosophila to tackle the problems associated with testing systems of computational dialectic. We present a framework for specifying and implementing dialectitc systems and our drosophila, a scenario named GC_0, which is used to test some aspects of the implemented system. The implementation enables formal dialectic systems to be rapidly set up and example dialogues to be produced. The aim is to use this process to generate a body of empirical data that can be used to investigate the properties of dialectic systems. The results of such an investigation can in turn be used to inform the research, construction and implementation of computational dialectic systems. This fulfills a need in the field of agent communication for a means to comparatively evaluate systems of computational dialectic.

2 Scenario: GC_0

Our approach is to specify a scenario in the domain of graph colouring problems [4] that provides a knowledge domain within which a MAS can be situated. In one graph colouring problem it is asked whether each vertex in a connected graph may be coloured using *n* colours such that no neighbouring vertices share the same colour. This is applied to a MAS context by assigning each agent a colour state and giving each agent the goal of resolving all conflicts with its neighbours. Conflicts occur when neighbouring agents are the same colour. The assumption is that agents may engage in argumentative dialogue in order to resolve conflicts and that the currency of the reasoning process are the agents colours and relationships. The core scenario is named GC_0 in relation to the graph colouring domain in which it operates. The subscript indicates the version of the scenario which is being applied. GC_0 sets out the properties of the domain which should be implemented in order to test computational dialectic. These include the number of colour states available to the agents, the minimum requirements for agent knowledge, the circumstances in which conflicts arise and are resolved, agent goals and actions that agents can perform within the MAS. In GC_0 an agent would initially have knowledge of only its own colour state and that it had relationships with a set of other agents. In order to increase its knowledge of the colour states of other agents or the relationships that those agents possess, an agent must engage in information seeking dialogues. Where these information seeking dialogues lead to the discovery of a conflict, the agent attempts to resolve that conflict through the use of argumentative dialogue. The use of dialogue, regulated by computational dialectic systems is the only means that an agent possesses in GC_0 to find out about or effect an alteration of its environment. It is foreseen that GC_0 will provide a baseline for computational dialectic testing and that enhanced and extended scenarios can be applied in order to gain a deeper understanding of particular problems.

3 Implementation: Sweetwater

A MAS named Sweetwater, has been constructed which incorporates GC_0. Sweetwater is built atop the Jackdaw agent framework using the Jackdaw University Development Environment (JUDE) [3] to provide basic multi-agent functionality.

The Jackdaw agent framework is a lightweight, flexible, industrial-strength agent platform that uses a modular approach to agent development. This enables domain specific functionality of the GC_0 scenario to be encapsulated into a module which is dynamically loaded into a Jackdaw agent. JUDE provides an environment for easily developing individual modules in Java.

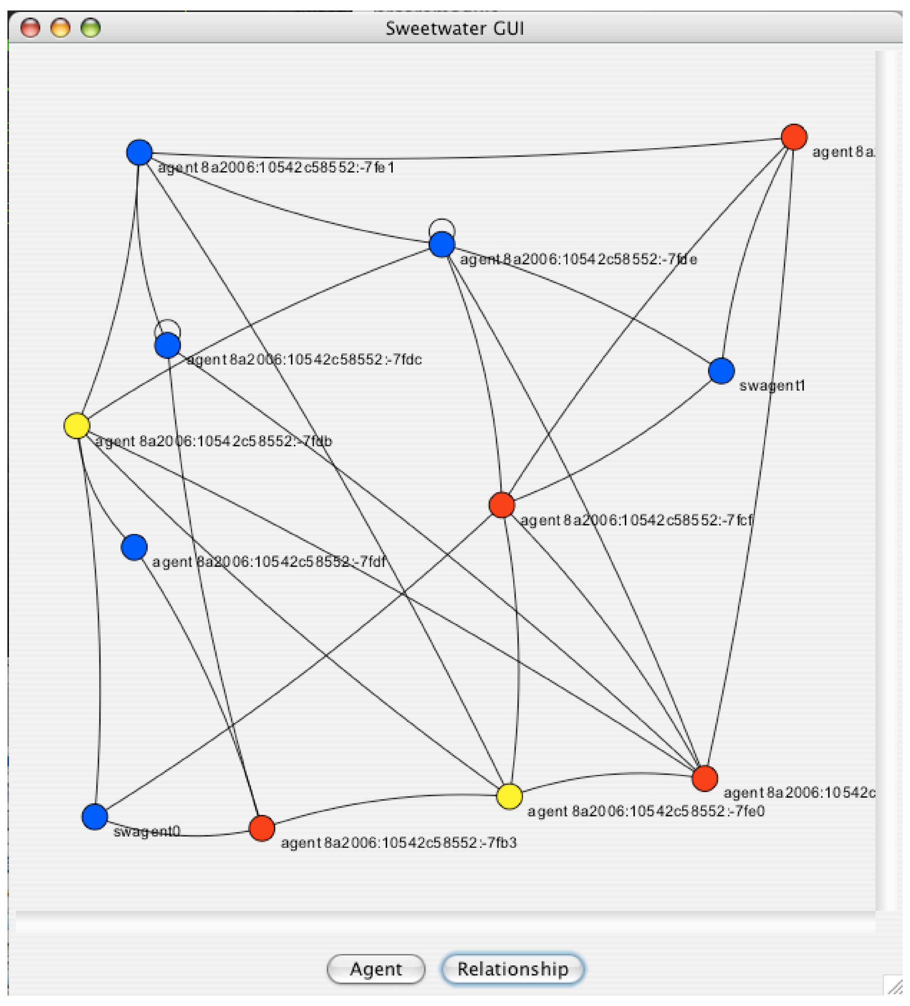

Figure 1: Sweetwater Visualisation GUI

A modular approach was taken to Sweetwater agent development which implements computational dialectic and the GC_0 scenario functionality as distinct components, a dialectic manager, a knowledge manager and a reasoning manager. The dialectic manager enables a computational dialectic system to be loaded at runtime from an XML specification file using a unified specification format [21]. This enables a wide range of computational dialectic systems, including H [7], DC [9], PPD [20] and DL3 [6] , to be implemented simply by specifying their rules in the required format. The knowledge manager incorporates a store to maintain information about an agents relationships and the colour states of other known agents. An agents knowledge is maintained through dialogue with its neighbours. As a result an agents knowledge can be limited and uncertain. Knowledge data is represented in an XML file which provides a simple, structured and easily extended means to store and access agent knowledge data. Arguments are constructed from concepts that are retrieved from the knowledge store, a process which is guided by the application of argument templates. Argument templates can be characterised as semi-instantiated argument schemes [18]. Argument schemes have been used in argument analysis and classification [14], and argument generation [16]. Argument templates are stored in XML files and specify the relations between knowledge concepts which can stand as premises and conclusions in an argument, in effect setting out the pattern for stereotypical arguments in the knowledge domain. The aim is to provide a means to access the knowledge store guided by argument theoretic concepts as is required by many systems of computational dialectic. The reasoning manager utilses an iterative heuristic process to controls the selection of moves, regulated by the dialectic manager, and content, regulated by the knowledge manager, to enable the agent to fully instantiate the set of relevant legal utterances at any given juncture with the goal of resolving any conflicts through argumentative dialogue. When a conflict is identified the agents attempt to persuade each other to change colour state by uttering requests which can be supported by arguments or attacked as required. By taking this modular approach scenarios other than GC_0 can be written and loaded into the MAS at runtime. It is straightforward to construct new components incorporating enhanced functionality which are loaded into the Sweetwater agents to enable extended scenarios to be explored and specific aspects of dialectic to be examined. A key consideration in this implementation was to enable a wide range of parameters to be set at run-time, both to increase the overall flexibility of the framework and to enable a wide range of experimental variables to be specified in order to tailor experiments toward particular aspects of dialogue. This was achieved through the use of XML files to specify the MAS structure, agent knowledge, argument templates and computational dialectic system.

4 Implementation Overview

Figure 2 gives an overview of the Sweetwater dialectic testing framework which was detailed in sections 2 and 3.

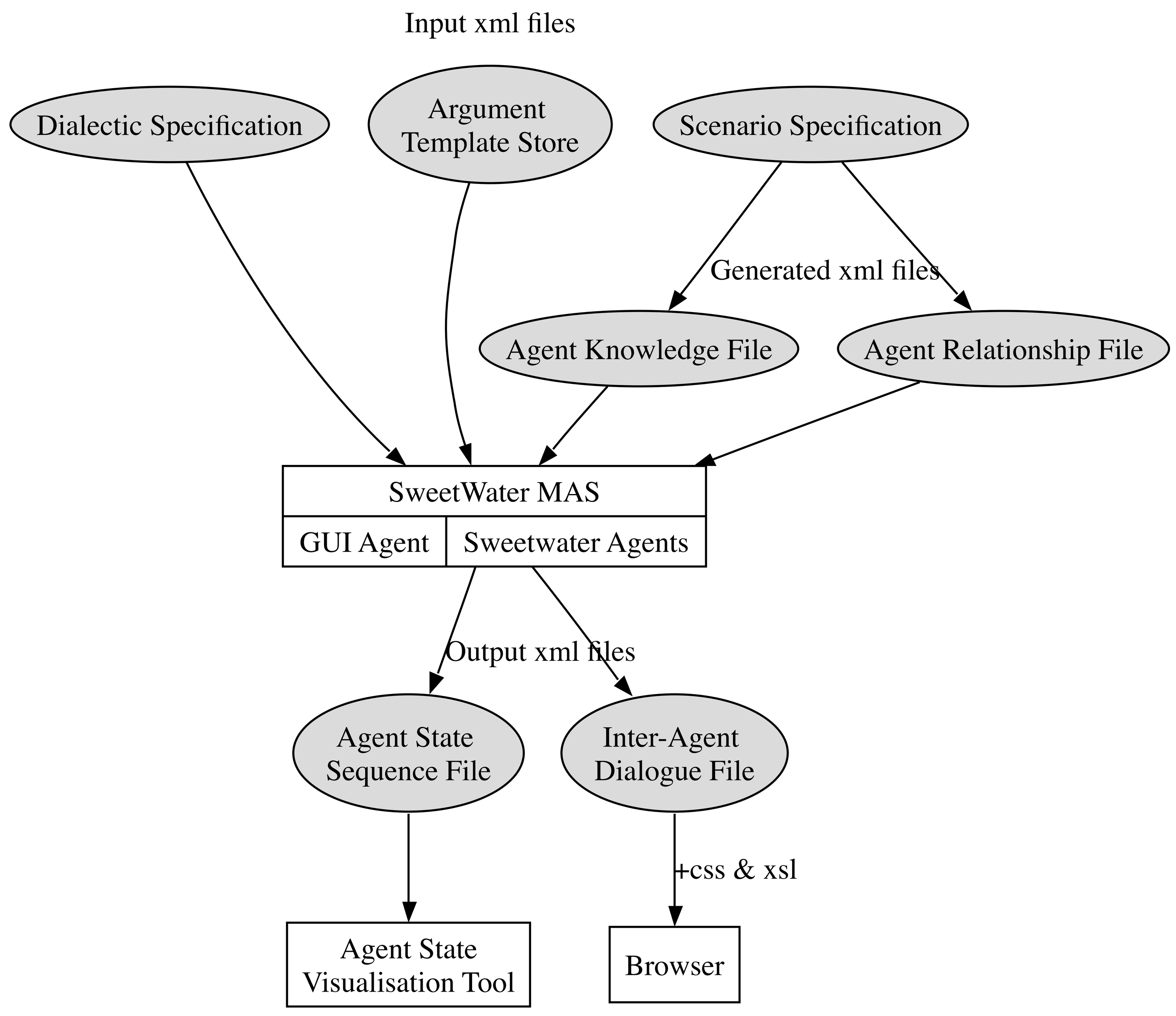

Figure 2: Overview of the Sweetwater Testing Framework

5 Evaluation Methodology

The drosophila builds upon the assumption that if the scenario and implementation remain constant, and the rules of the computational dialectic system are varied, then any differences in output that are measured will be attributable to the altered rules of the computational dialectic system. Utilising the common specification format of [21] enables existing formal dialectic systems to be recast into a common format and comparatively tested as well as enabling a systematic exploration of the space of possible systems to be made. To enable the difference between computational dialectic systems to be characterised, metrics are used. Two sets of metrics have been identified that can be applied to computational dialectic, *inspection metrics* which can be obtained through examination of the rules themselves, and *process metrics* which are obtained through application of the rules in dialogue. Sweetwater enables the measurement of both inspection and process metrics for a range of computational dialectic systems.

6 Summary

The drosophila is of benefit to researchers in AI and MAS because it enables the comparative testing of disparate systems of computational dialectic. Testing is required to support the wider adoption of computational dialectic as a standard means of structuring inter-agent communicative acts. GC_0, is simple, well-defined and easily extended to afford a focus on particular aspects of inter-agent argumentative communication. GC_0 and Sweetwater have a range of benefits for both theoretical and practical work. These include a means to test arbitrary dialectic systems using a unified knowledge base and a means to determine standard metrics by which dialectic systems can be measured, compared and characterised. An additional benefit is the generation of a corpus of example dialogues for each system of computational dialectic which can be used to inform future research.

References

[1] L. Amgoud, S. Parsons, and N. Maudet. Arguments, dialogue, and negotiation. *(ECAI 2000)*, 2000.

[2] Michael Ashburner. *Drosophila*. Cold Spring Harbor Laboratory Press, 1989.

[3] Calico jack website. http://www.calicojack.co.uk, 2004.

[4] A. Cayley. Open problem. *Proceedings of the London Mathematical Society*, 9:148, 1878.

[5] F. Dignum, B. Dunin-Keplicz, and R. Verbrugge. Agent theory for team formation by dialogue. *(ATAL 2000)*, 2000.

[6] R. A. Girle. Commands in dialogue logic. *Practical Reasoning, Springer Lecture Notes in AI*, 1996.

[7] C. L. Hamblin. *Fallacies*. Methuen and Co. Ltd., 1970.

[8] E. C. W. Krabbe. Profiles of dialogue. In J. Gerbrandy, M. Marx, M. de Rijke, and Y. Venema, editors, *JFAK*. Amsterdam University Press, 1999.

[9] J. D. Mackenzie. Question begging in non-cumulative systems. *Journal Of Philosophical Logic*, 8:117–133, 1979.

[10] N Maudet and F. Evrard. A generic framework for dialogue game implementation. In *(FSPD 1998)*, 1998.

[11] J. McCarthy. Elaboration tolerance.

[12] J. McCarthy. Ai as sport. *Science*, 276(5318):1518–1519, 1997.

[13] S. Parsons and N. R. Jennings. Negotiation through argumentation. *(ICMAS'96)*, pages 267–274, 1996.

[14] C. Reed and G. Rowe. Araucaria. Technical report, University Of Dundee, 2001.

[15] C. Reed and D. Walton. Argumentation schemes in argument-as-process and argument-as-product. In *Proceedings of the Conference Celebrating Informal Logic @ 25*, 2003.

[16] C. Reed and D. Walton. Towards a formal and implemented model of argumentation schemes in agent communication. *(ARGMAS)*, 2004.

[17] H. Simon and J. Schaeffer. The game of chess, 1992.

[18] D. N. Walton. *Argumentation Schemes for Presumptive Reasoning*. Lawrence Erlbaum Associates, 1996.

[19] D. N. Walton. *Logical Dialogue-Games And Fallacies*. University Press Of America, 1984.

[20] D. N. Walton and E. C. W. Krabbe. *Commitment in Dialogue*. State University of New York Press, 1995.

[21] S. Wells and C. Reed. Formal dialectic specification. *(ARGMAS)*, 2004.

[22] J. Woods and D. N. Walton. Question-begging and cumulativeness in dialectical games. *Nous*, 16:585–605, 1982.