Mathematical Structures for Rational Discourse

Desmond Fearnley-Sander
Department of Mathematics
University of Tasmania
dfs@hilbert.maths.utas.edu.au

Abstract

This paper reports the implementation of a system for the creation of *rational automata*: automata that communicate with one another, display curiosity, learn and are creative. The author calls the implemented automata *narrow minds*. The formal mathematical structure of a rational automaton is defined, a sketch is given of how they work and how they interact, and a sample dialogue between narrow minds is presented. Equation processing plays a central role. It is argued that basic everyday thinking and basic mathematical thinking, though very different in some ways, can be implemented using the same mechanisms.

The narrow minds system takes a step toward the productive social interaction of rational and, in particular, mathematically capable autonomous computer entities. Its implementation rests upon two fundamental computational paradigms: equational programming and object-oriented programming, the one for implementation of simple intelligence, the other for implementation of simple social interaction. The system user can control the level of intelligence of the narrow minds that are created, how they interact in dialogue, how they speak, and, to some extent, how they think. Parallelism is intrinsic to the system, as it is to human communities.

Some extensions are outlined, and the relationship is considered of the work reported here to automatic theorem-proving.

The paper begins by defining a class of automata that I call *rational automata* or, informally, *narrow minds*. Narrow minds are capable of dialogue with one another. They display human-like characteristics. They learn and

forget. They may have different personalities. They may co-operate or compete. In the second part of the paper the implementation of a Narrow Minds system is discussed and a sample dialogue generated by an implemented system is presented. Implementation involves two paradigms of computation: object oriented programming and equational programming. In the final part of the paper we discuss some extensions, the relationship of these ideas to automated theorem proving and briefly speculate on potential applications.

1 Positive Rational Automata

Positive rational automata are particular kinds of automata for which the inputs and outputs are equations. Some preliminary definitions are needed.

Our notion of automaton is a standard one. An *automaton* $\mathcal{A}$ is a 4-tuple $\langle Q, X, s, o\rangle$ where

- Q is the set of *states*,
- X is the set of *messages*,
- $s : Q \times X \to Q$ is the *next state function*, and
- $o : Q \to X$ is the *next output function*.

If at time n the automaton is in state $q[n]$ and registers an input $u[n] \in X$, then at time $n+1$ it will move to state $q[n+1] := s[q[n], u[n]]$ and compose the output $o[q[n+1]] \in X$.

An *equational language* L is a set of equations over some algebraic signature Σ; each *equation* is an ordered triple $\alpha =_i \beta$, where α and β are terms of the signature. A *consequence relation* over L is a relation $\models: 2^L \to L$ from sets of equations to equations that satisfies

- if $\psi \in \Gamma$ then $\Gamma \models \psi$;
- if $\Gamma \models \psi$ then $\Gamma \cup \Delta \models \psi$;
- if $\Gamma \models \theta$ and $\Delta \cup \{\theta\} \models \psi$ then $\Gamma \cup \Delta \models \psi$.

A *positive rational automaton* $\langle L, \models, Q, X, s, o\rangle$ with consequence relation $\models$ over an equational language L is an automaton $\langle Q, X, s, o\rangle$ for which

- the elements of Q are sets of equations from L;
- the elements of X are equations from L;
- $s[\Gamma, \phi]$ is a set of consequences under $\models$ of $\Gamma \cup \{\phi\}$.

2 Dialogues

A *dialogue* is a pair $\langle \mathcal{G}, \nu \rangle$ where $\mathcal{G} = \{1, 2, \ldots\}$ is a set of positive rational automata (all with the same language and state space, but with different consequence relations $\models_i$ and different information processing functions s_i and o_i) called the *participants* in the dialogue, and $\nu : \mathbb{N} \to \mathcal{G}$ is a sequence of participants. At each instant $n \in \mathbb{N}$ the output of a participant $\nu[n]$, called *the speaker*, is registered by each participant as its next input. The outputs of the successive speakers are called the *utterances* of the dialogue. Of all the outputs available from the participants at time n only the one that is uttered by the speaker is registered by the participants.

The sequence u of utterances is called the *record* of the dialogue. It is determined by an initial utterance $u[0]$, the initial states $q_i[0]$ of the participants, their thinking processes, o_i and s_i, and the sequence of speakers ν:

$$u[1] = o_{\nu[1]}[s_{\nu[1]}[q_{\nu[1]}[0], u[0]]]$$

and, for $n > 1$,

$$u[n] = o_{\nu[n]}[s_{\nu[n]}[q_{\nu[n]}[n-1], u[n-1]]]$$

The course of the dialogue will be completely predictable if all this information is known; and it will be unpredictable to the extent that it is not known. We say that a description (or partial specification) of a dialogue is *non-deterministic* if it does not support, at every stage, exact prediction of the next utterance. To a listener, whose only information about the dialogue is the sequence of utterances so far, the dialogue is thoroughly non-deterministic in this sense. Nevertheless it is somewhat predictable. In a realistic dialogue participants will think in parallel, and the next speaker will be the first to be ready; this can be simulated in a serial implementation.

I will refer to the participants in a dialogue as *narrow minds.* In the current implemented system discussed in this paper, the underlying algebra signature is as simple as possible: the semigroup signature. Each term is a string of generators, called *words*; a word may include tags indicating, for example. who uttered it.

3 Implementation

Narrow Minds is an implemented system for the creation of rational automata and of dialogues between gatherings of rational automata. The implementation is of interest because it naturally involves two mainstays of

modern computing: *object oriented programming* and *equational programming*.

At the highest level of the system we have an object-oriented program of which the main *class* is the class of narrow minds. The user creates an *object* of this class by giving values for its many attributes; values that specify its initial state and the *methods* by which it is to construct its successive states and outputs. In a dialogue, each utterance is a *message* from the speaker to the other participants.

Narrow minds process equations. The engine that drives the Narrow Minds system is the Knuth-Bendix algorithm. (See [1] or [2].) It is the appropriate equational algorithm for a system in which terms are simple strings of words. For algebras of other signatures, other equational algorithms will be appropriate. For example, to extend the narrow minds system to encompass rational discussion of geometric structures, including inferences about such structures, the Buchberger algorithm may be used. (See [3], for example.)

It is not possible here to present a detailed discussion of the ways in which equational programming supports inference, but an example will suffice to show that it does. Suppose that the knowledge of a narrow mind ξ includes the equations:

$$\begin{aligned} \texttt{father Lisa} &=_i \texttt{Homer} \\ \texttt{wife father} &=_i \texttt{mother} \\ \texttt{wife Homer} &=_i \texttt{Marge} \end{aligned}$$

Then among the equational consequences that ξ may output is the equation

$$\texttt{mother Lisa} =_i \texttt{Marge}$$

Note that there are other equational consequences, such as

$$\texttt{Homer wife Homer} =_i \texttt{father Lisa Marge,}$$

that one would not want ξ to output. Similarly, from $\texttt{today} =_i \texttt{Monday}$ and $\texttt{today} =_i \texttt{Tuesday}$ it should not be inferred that $\texttt{Tuesday} =_i \texttt{Monday}$. A critical feature of our implemented narrow minds is that their consequence relations are stronger than equational consequence, so that unwanted utterances are not generated.

4 A Sample Dialogue

The following is the transcript of part of an interactive session with an implemented Narrow Minds system. User input is underlined. At the outset,

the user has created the dialogue `d123` with three participants, `p1`, `p2` and `p3`; in particular, the participants have been given various attributes such as their names ("Lisa", "Homer" and "Marge", respectively) and their initial knowledge states.

The utterances of the participants, printed here, are generated and spoken in real time. Utterances are rendered in ordinary English, which though not always grammatically perfect, is always comprehensible. One of the purposes of the system is to serve as a test-bed for the translation from the formal internal language to natural language.

You will see that `p1` learns a good deal in a short time — not only what she has heard but facts that can be inferred from what she has heard. You will also see, in the very last utterance that narrow minds may disagree; the disagreement as to Lisa's father depends upon the introduction, not discussed in this paper, of a feature that takes us beyond positive rational automata: falsity. Rational automata with this additional capability can disagree, argue and lie.

```
OUTPUT[KNOWLEDGE, p1]
```

```
I know that: The father of Lisa is Homer. The mother of Lisa
is Marge. Myself is Lisa. The kind of day is good.
```

```
RUN[d123,15]
```

```
MARGE:
Sue is my mother.
LISA:
My father is Homer.
HOMER:
I know that your father is me.
42 is my age.
LISA:
Marge is my mother.
HOMER:
What is my age?
LISA:
Homer, your age is 42.
Me is me.
MARGE:
The father of father is grandfather.
LISA:
The father of Homer is my grandfather.
```

```
HOMER:
I know that your grandfather is my father.
The sex of Marge is female.
LISA:
Me is myself.
MARGE:
Lisa, I know that you is my smart daughter.
My father is Bill.
LISA:
Your grandfather is the father of Bill.
MARGE:
I know that my grandfather is the father of Bill.
My age is 32.
LISA:
My father is Homer.
MARGE:
I don't believe that your father is Homer.
Your grandfather is the father of Flanders.
```

```
OUTPUT[KNOWLEDGE, p1]
```

```
I know that: The age of Homer is 42. The age of Marge is 32.
The father of father is grandfather. The father of Lisa is
Homer. The father of Marge is Bill. The sex of Marge is
female. The grandfather of Lisa is the father of Homer. The
grandfather of Marge is the father of Bill. The kind of day
is good. The mother of Lisa is Marge. The mother of Marge is
Sue. Me is Lisa.
```

5 Extensions

Currently the powers of the implemented Narrow Minds system are rather limited. There are many ways in which they can be extended. One is to enhance the expressiveness of the utterances of the participants, allowing the formation of more elaborate sentences by introducing conjunction of equations and exclusive disjunction, with rules that make the set of sentences a Boolean ring $R[\oplus, ., T, F]$. We call the resulting structures *equality algebras*; they are algebras equipped with a binary Boolean ring-valued operator $=_i$ which satisfies

- $(a =_i a) = T$; and

- for any operator f with codomain R

$$(a =_i b).f[a] = (a =_i b).f[b].$$

The properties of equality algebras and, in particular, their relationship with lattices of congruences over algebras are investigated in [4]. See also [5] and [6].

Rational automata enriched in this way support the generation of simple explicit theorem statements, such as the following:

$$(\texttt{father x} =_i \texttt{Homer}).(\texttt{wife Homer} =_i \texttt{Marge}) \Rightarrow (\texttt{mother x} =_i \texttt{Marge})$$

("If Homer is the father of a person and Marge is the wife of Homer then Marge is the mother of that person.") A narrow mind whose knowledge includes the equation $\texttt{wife father} =_i \texttt{mother}$ may prove such an assertion by using the Boolean ring reduction $\phi \Rightarrow \psi = \phi.\psi + \phi + T$ together with its basic equation processing.

A further extension is to internalize equality of sentences; then the map that takes each equation ϕ to the equation $\phi =_i T$ is a modal operator. See [7]. Exploration of these connections with propositional and modal logics is continuing.

6 Rationality versus Theorem Proving

Narrow minds display rational behaviour. In particular each utterance of a narrow mind is correct[1] for its current knowledge state: it is a theorem. So what is the relationship of this theory to automated theorem proving? We briefly consider this question.

The fundamental difference between main-stream theorem proving systems and the Narrow Minds system is that the one *recognizes* while the other *generates*. Automated theorem provers recognize theorems by providing proofs. In the main-stream logic-based approach to automatic theorem proving, proof discovery is essentially a search process: a path must be found from the hypotheses to the conclusion, each step of which is the application of an inference rule. Nothing can be concluded form failure to find a proof. I have argued elsewhere (in [8]) that the task of proving difficult theorems using independent inference rules (and axioms) is practically intractable for stand-alone theorem provers such as OTTER. This is not to say that such

[1] It is easy to see how to extend our notion of narrow mind to allow *duplicity* — the uttering of equations that are not correct — but we shall not consider that here.

provers are without value, merely that human input in various forms, such as provision of the right lemmas, plays an essential role.

Equational theorem provers based on the pioneering work of Wu and Buchberger (see [9], [10], [11] and [3]) have come to prominence in recent years, mainly because of their extraordinary success in proving geometry theorems. They work quite differently, the proving process being generation rather than search. The Narrow Minds system belongs squarely in this camp. It adds new perspectives to the theorem-proving point of view. Narrow minds are not just passive respondents to requests for information: they ask for information, they communicate, they co-operate or compete, and they learn from one another. So far as I know the possibilities of using polynomial equations for intelligent discourse between automata about geometric objects have not yet been investigated.

It will be remarked, of course, that the theorems which are generated by the Narrow Minds system are trivial. Remember, though that the quadratic formula was at one time a great advance in the development of mathematics. Narrow Minds systems open the possibility of a mechanized development of mathematics which parallels its development by humans: development through the social interaction of individuals that learn and are creative.

7 Conclusion

The potential applications of narrow minds are many. To take one example, a simple adaptation of the current system, probably achievable quite easily, would be to create autonomous, intelligent questionnaires (or questioners!): rational automata that address multiple-choice questions to human subjects, interpret the responses as equations, and make use of the information that is inferred from the responses for the purpose of keying subsequent questions to the distinctive characteristics of the person who is being interviewed. Other possibilities, such as the making of applets with genuine autonomous intelligence, will suggest themselves to the reader.

A final disclaimer is necessary. It is not claimed that narrow minds simulate the processing that is performed by human brains. Indeed it seems that narrow minds differ in an essential way from human minds since they process phrases without having any external semantics for those phrases. What is claimed is merely that dialogues between narrow minds simulate dialogues between humans. It is conjectured that narrow minds can be engineered which will pass an *objective rationality test*: fragments of dialogue between them will be indistinguishable from fragments of dialogue in a play or in

actual human conversation. I call such a test *objective*, since it requires the judge to be a neutral observer, rather than, as in the Turing test [12], being an active participant in the dialogue.

Acknowledgements. Implementation of the Narrow Minds system that generated the sample transcript given above was done by Peter Purdon and the author; Peter was responsible for design and implementation of the Incremental Knuth-Bendix engine that powers the reasoning system. The author also thanks Tim Stokes, Peter Purdon and Damien Foster for many discussions concerning algebraic and logical aspects of equational reasoning.

References

[1] N. Dershowitz and J.-P. Jouannaud. Rewrite systems. In J. van Leeuwen, editor, *Formal Models and Semantics*, volume B of *Handbook of Theoretical Computer Science*, pages 243–320. Elsevier, 1990.

[2] J. W. Klop. Term rewriting systems. In S. Abramsky, D. M. Gabbay, and T. S. E. Maibaum, editors, *Background: Computational Structures*, volume 2 of *Handbook of Logic and Computer Science*, pages 243–320. Oxford, 1992.

[3] D. A. Cox, J. B. Little, and D. O'Shea. *Ideals, Varieties and Algorithms.* Springer-Verlag, 1992.

[4] D. Fearnley-Sander and T. Stokes. Equality algebras. *Bull. Aust. Math. Soc.*, 56:177–191, 1997.

[5] M. Bulmer, D. Fearnley-Sander, and T. Stokes. Toward a calculus of algorithms. *Bull. Aust. Math. Soc.*, 50:81–89, 1994.

[6] D. Fearnley-Sander. A logic of definitional reasoning. *Australian Computer Science Communications*, 18:81–89, 1996.

[7] D. Fearnley-Sander and T. Stokes. Internalization of equality in boolean algebras. *Algebra Universalis*, accepted for publication.

[8] D. Fearnley-Sander. Automated theorem proving and its prospects. *Psyche*, 2, 1996.

[9] W.-T. Wu. *Mechanical theorem proving in geometries: Basic principles.* Springer, 1994.

[10] S.-C. Chou. *Mechanical Geometry Theorem Proving*. Reidel, 1988.

[11] B. Buchberger. Gröbner bases: an algorithmic method in polynomial ideal theory. In N. K. Bose, editor, *Multi-dimensional Systems Theory*, chapter 6, pages 184–232. Reidel, 1985.

[12] A. Turing. Computing machinery and intelligence. *Mind*, 59:422–460, 1950.