Research Specification: Exploring optimisation of constraint program solvers with sonified algorithms[1]

G. Han[2] and A. Judd[2]

Monash University,
Faculty of Information Technology,
Clayton, Australia

E-mail: ajud0001@monash.student.edu, ghan0004@monash.student.edu

Abstract. Constraint satisfaction problems model many real world applications but are generally computationally expensive to solve. This is due to their exponentially scaling domains and search space. This project investigates the usefulness of sonified algorithms in optimising constraint satisfaction problem solvers. This will be approached through the analysis of sonified execution traces of constraint program solving algorithms.

[1]Research specification produced for FIT2082: Computer Science Research Project
[2]Supervised by Guido Tack and Kevin Leo

Contents

1 Introduction

Constraint satisfaction problems reflect many real world scenarios. They are extremely important in both the optimization and satisfaction domain. The search space and, in turn the solving time of constraint optimisation problems increase exponentially with the size of the problem. As such, techniques for reducing overall search space is an active field of research.

This research project aims to explore possible optimisations to constraint problem solving. This problem will be approached through a novel idea, termed 'sonification' of algorithms. This approach is rooted in the parallels that can be drawn between mathematical substructure of algorithmics and the mathematics underlying music theory. For this project we will extend the work of A. Lallouet and J. Vautard, who developed *'CSPsinger'*, a tool which auralizes the execution of CSP solving algorithms. [1] , this project will sonify execution traces of constraint problem solving algorithms and then analyse them. This project seeks to evaluate the viability of sonification as an analysis tool, and discover possible optimisations to CSP solvers.

2 Background

2.1 Constraint Satisfaction Problems

Constraint satisfaction problems are mathematically defined questions which are solved by searching a set domain of possible values and finding solutions which satisfy imposed constraints. CSPs can model and express many real world problems and are considered an extremely useful framework for combinatorial optimisation and satisfaction. Examples of constraint satisfaction problems include both classical problems such as Map Colouring and N-Queens, and also practical problems such as timetable and employee scheduling.[2]

Generally, CSP are concerned with very computationally demanding problems, formally - the NP-complete domain of problems [3]. In addition, the time needed to solve CSPs increases exponentially with problem size. Hence, the main optimisation to CSP solving is reducing the overall search space, decreasing the time necessary to solve the problem.

CSPs have been described and written on extensively, in particular, with regards to optimisations or search space reduction heuristics. Kumar, 1992 [4] and P. Meseguer, 1989 [5] provide an overview of the the methods for solving CSPs.

2.2 Sonification

'Sonification' or 'auralization' are novel approaches to representing algorithms. While, there is no formal definition or theoretical framework to this paradigm, the general technique, and the approach we will be taking, is as follows. A tone is assigned to each variable based on its value and changes reflecting the behaviour of the algorithm. Essentially making music from the execution of the algorithm.

This was described by A. Lallouet and J. Vautard in 'Auralization of a Constraint Solver' [1], which forms the basis for this project. In this paper, A. Lallouet and J. Vautard describe the development of *CSP-singer*, - an open-source program which auralizes the execution trace of CSP solving algorithms. Additionally they provide and describe their approach to algorithm sonification. This is the approach our research will follow as we extend their software to support the CSP solver *GECODE*.

3 Motivation and Aims

The purpose of this project is to discover whether there is any merit in sonifying algorithms. In particular, investigating whether they convey any new information about the structure of CSP solving. We hope to apply optimisations to CSP solving with any information discovered through this approach.

3.1 Aims

The following are aims of this research project

- To understand the theoretical framework underlying CSPs
- To develop a tool which enables the modelling of behaviour of CSP solving algorithms through sonification.
- To analyse and compare sonified algorithms to reduce the search space of CSP solving.
- To address current deficiencies in CSP solvers based on information interpreted from sonified algorithms.
- To evaluate the merit of sonifying algorithms as a tool for analysis.

3.2 Research Question

Can algorithm sonification serve as a useful technique in runtime analysis?

More Specifically:

- Can meaningful conclusions be drawn about a CSP-solver's behaviour by listening to an auralised representation of its decision tree?
- Can the CSP-solver be optimised through the use of any significant interpretations made?

4 Methodology/Research Plan

The methodology for this research project is set out in five stages, – prepatory, pre-practical, practical, experimental and evaluation. These stages describe milestones and reflect the application and achievement of our aims as set out in 3.1. The preparatory stage enables us to achieve an understanding of the theoretical and technical background to this project. The pre-practical and practical stage encompass the development of the unified software tool. The experimental stage allows us to use the developed software, evaluate, and, analyse sonified algorithm behaviour. Finally, the evaluation stage consolidates our findings and assesses their usefulness. Specific details are outlined below.

1. **Preparatory**
    - Explore the pre-existing algorithmic environment and past research

- Formulate research questions to underpin the research techniques applied
- Establish a practical scope which encompasses the research questions and yields achievable aims
- Determine the research aims as a logical progression to answer the research questions.

2. **Pre-practical**
    - Gain an understanding of the CSP-Singer program and the sonification techniques in practice
    - Analyse the GenTra4CP format – the input, and the MIDI format – the output of CSP-Singer
    - Build familiarity with MiniZinc and the CSP solver in use, and in turn the profiler used to map its decision tree
    - Map out the necessary steps to use the profiler and solver in conjunction on a variety of CSPs
    - Analyse the profiler's output SQLite Databases and draw comparisons between the event sequences it describes and the GenTra4CP input
    - Establish a method of playing CSP-Singer's output MIDI files, or otherwise a method of converting from MIDI to a more convenient format

3. **Practical**
    - Develop a program which parses the records in the SQLite Databases and produces GenTra4CP output compatible with CSP-Singer
    - Generate a variety of SQLite Databases, ideally:
        - Using varying CSP solver algorithms in MiniZinc
        - Using varying CSPs with a range of constraint values
        - Categorised by the problem, problem class, solver runtime, or other attribute

4. **Experimental**
    - Use the developed program to generate a large number of input files for CSP-Singer and map them to their MIDI outputs
    - Take the MIDI files and manually analyse them – listening for:
        - Musical pattern
        - Structure and tone
        - Frequency and rhythmic structure
        - Whether or not similar classes of problems exhibit similar musical structure
    - Attempt to relate anomalies identified to events the profiler has reported (as per the decision tree) - note these findings

5. **Evaluation**
    - Analyse the connections between the musical anomalies and solver's events

- Determine whether any of these connections are indicative of a substantial discovery by the solver
  - If so, determine whether the solver could terminate early, exclude meaningless potential solutions to the CSP, or otherwise modify its behaviour to approach optimisation

4.1 Timeline

Table 1 Proposed Timeline

Week 2	**Stage I**- Attend initial meeting, gain an understanding of the research context and high-level purposes, establish timeline, formulate methods of inquiry, research questions and aim.
Week 3	Familiarise self with technical tools to be used (MiniZinc), preparation of project specification, refine goals and research questions. Begin exploration of the algorithms and salient background work.
Week 4	**Stage II** - Analyse the in and output formats of each of the programs. Run practical tests to ensure they can be used in conjunction with eachother. Familiarise self with GE-Code, CSP Singer, MiniZinc.
Week 5	**Stage III** - Develop the middle-man program responsible for parsing the decision trees, test it on sample 8-queens databases
Week 6	Use MiniZinc and the profiler to generate output to modularise and further test the program, generate a database of SQLite files
Week 7	**Stage IV** - Conduct experiments sonifying different types and classes of CPS solving algorithms.
Week 8	Continue experimentation and analysis of algorithms. This could include tabulating sonified algorithms and examining the musical structure
Week 9	**Stage V** - Evaluate the meaning of the analysis conducted. Formulate a plan for optimising CSP solver if appropriate results are yield.
Week 10	Continue evaluation and optimisation. Begin preparing results for final report and presentation.
Week 11	Write up findings and create poster.
Week 12	Prepare for the final presentation

4.2 Proposed Outcomes

At the culmination of this project the CSP-Singer software will have been extended in order to accomodate input from MiniZinc and GECODE (when used in conjunction with a profiler). Due to the highly experimental nature of this research there is not a clear deliverable in terms of CSP solver optimsation. For instance: sonifying CSP algorithms may prove fruitless from an audible perspective. Listening to a sonified decision tree could aid in interpretability, but yield no clear path towards solver optimisation. Conversely, auralisation could produce an opaque cacophony. The main purpose of this research project is to ascertain whether or not sonification is a meaningful approach in understanding the behaviour of these algorithms. We expect that there could be future research into applying machine learning analysis to the output MIDI files, as they could yield a meaningful result even if human-ear analysis cannot, but this is outside the scope of this project.

References

[1] A. Lallouet, J. Vautard, *Auralization of a Constraint Solver*, ICMC'06, International Computer Music Conference, New Orleans, LA, USA, November 6-11, 2006.

[2] A.E. Eiben and Zs. Ruttkay, *Constraint Satisfaction Problems*, doi:10.1.1.49.2753

[3] A.K. Mackworth, *Consistency in networks of relations*, Articial Intelligence, 8, 99-118, 1977

[4] V. Kumar, *Algorithms for Constraint Satisfaction Problems: A Survey*, AI Magazine, 13, 1992

[5] P. Meseguer, *Constraint Satisfaction Problems: An Overview* , AI Communications, vol. 2, no. 1, pp. 3-17, 1989