Decision procedures with Automata

Alex J. Best

January 23, 2024

Motivation

Motivation: One example, if $a, b, c, d \in \mathbb{N} \cup \{\infty\}$, lets say we want to prove

$$\begin{aligned} & a + b \leq c \\ & 2a \leq d \\ \implies & 3a + b - 1 \leq 2c + d \end{aligned}$$

if $a, b, c, d \in \mathbb{N}$ there is a decision procedure for this (in Lean, `linarith`).

Motivation

Motivation: One example, if $a, b, c, d \in \mathbb{N} \cup \{\infty\}$, lets say we want to prove

$$\begin{aligned} & a + b \leq c \\ & 2a \leq d \\ \implies & 3a + b - 1 \leq 2c + d \end{aligned}$$

if $a, b, c, d \in \mathbb{N}$ there is a decision procedure for this (in Lean, `linarith`).

Do we need to implement a new / modified decision procedure for each new type of object we want to reason about? **Answer:** No, one general decision procedure can be used for many different types of objects. Comes from the notion of an *automatic structure*.

Motivation

Motivation: One example, if $a, b, c, d \in \mathbb{N} \cup \{\infty\}$, lets say we want to prove

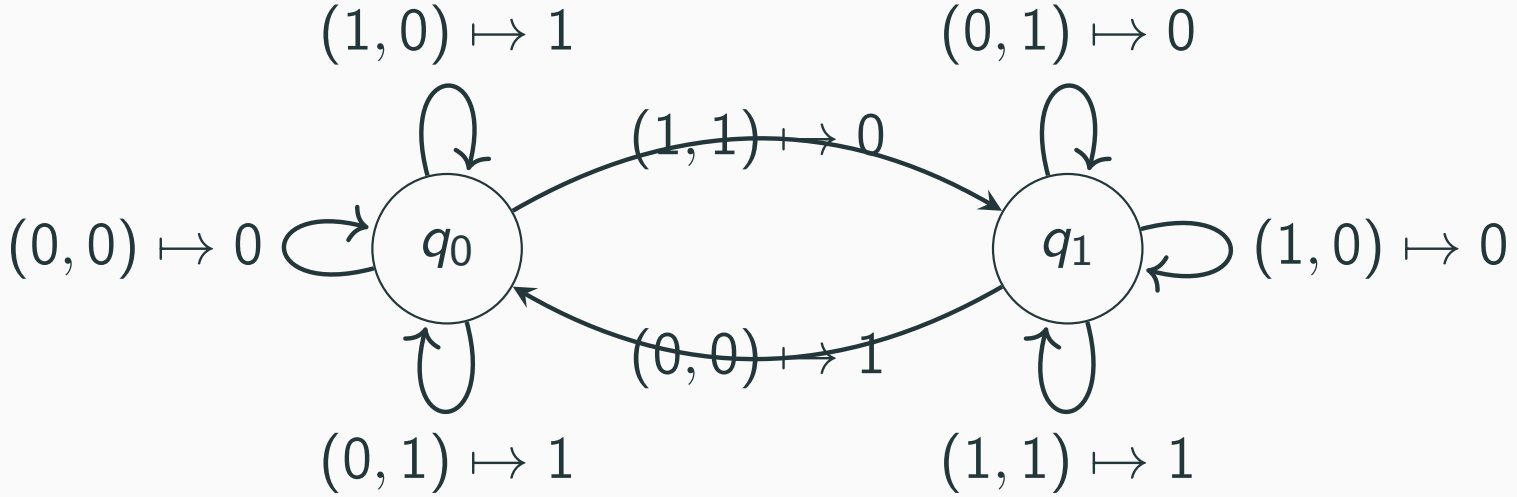
$$\begin{aligned} & a + b \leq c \\ & 2a \leq d \\ \implies & 3a + b - 1 \leq 2c + d \end{aligned}$$

if $a, b, c, d \in \mathbb{N}$ there is a decision procedure for this (in Lean, `linarith`).

Do we need to implement a new / modified decision procedure for each new type of object we want to reason about? **Answer:** No, one general decision procedure can be used for many different types of objects. Comes from the notion of an *automatic structure*. Sidebar: this sort of problem came up in a paper I'm working on verifying some mathematical algorithms.

Automatic structures

Consider adding a pair natural numbers bitwise



we can turn this into an automata that recognises whether a pair of naturals has a fixed sum. Questions about existence of solutions to equations can be reduced to questions about reachability in an automata. can decide any first order sentence about the natural numbers under addition using this technique.

$$\exists n, \forall m \geq n, \exists a\, b\, c, 6a + 9b + 20c = m$$

Automatic structures

This all generalizes to any mathematical object that can be represented by a regular language in this way.

With a sufficiently general implemention its easy to add a new element ∞ to the theory for example.

Many examples:

- Natural numbers
- Integers
- Real numbers (Büchi automata)
- Mixed integer linear programming
- Some specific groups
- Strings

Implementation?

Questions:

- How to represent automata in an efficient way in a proof assistant?
- The automata grow quickly in theory, in practice this seems to be less of an issue with good minimization procedures (cf. Walnut)
- Should one write tactics produce proofs, or prove once and for all that the algorithm is correct (reflection)
- How to make this convenient for an end user to use? How to make it easy to add new automatic structures? To give a translation from some objects with operations to automata?