ICFP Contest 2010

How 1000 people produced 3746 relative termination problems in 72 hours

Bertram Felgenhauer, University of Innsbruck
Johannes Waldmann, HTWK Leipzig

AJWS 2010, 2010-08-06

about the ICFP contest

- ICFP conference
- annual contest, with prizes sponsored by ACM
- 2010 contest organized by team lead by Johannes Waldmann
- contest format
  - 72 hours (June 18, 12:00 – June 21, 12:00 GMT)
  - programming task
  - participation online, international
  - teams allowed
  - no fixed programming language
  - lightning division (first 24 hours)

people

- Web server programming and maintenance:
  Daniel Borkmann, Tobias Kalbitz, Christopher Schädlich, Michael Schmeißer
- Web design, Brute force solver:
  Johannes Erber
- Log file evaluation:
  Christian Reichmann
- Contest task design, semantics server programming:
  Bertram Felgenhauer, Johannes Waldmann
- external testers:
  Alexander Kiel (Univ. Leipzig), Georg Martius (Univ. Göttingen), Henning Thielemann (Univ. Halle)
- external advisor:
  Robby Findler (Northwestern Univ.)

infrastructure

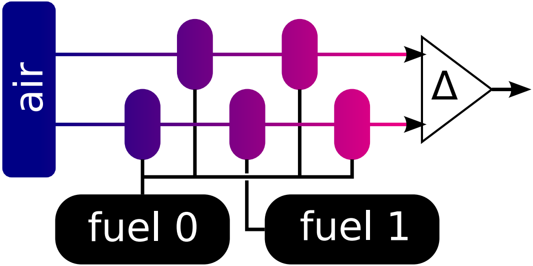
- contest server with
  - presentation (Java, Spring, Roo)
  - database (Postgres, Hibernate)
  - semantics (Haskell, XmlRpc)
- static information (task description)
- live information (blog)
- a bumpy ride. . .

contest task

- core problem: matrix interpretations for relative termination in SRS
- packaging:
  - SRS = car
  - matrix interpretation = fuel
  - *International Cars & Fuels Production*
- obfuscation:
  - ternary encoding
  - circuits
- teams collected points for submitted fuels and cars, accumulated over time

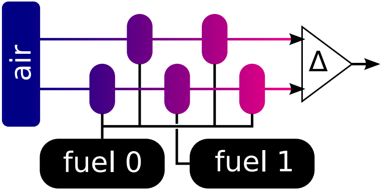
car engines

- chambers = rules



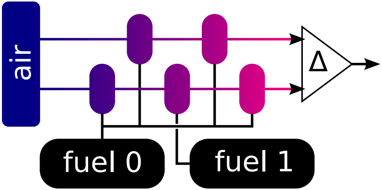
- upper, lower pipe = lhs, rhs of rewriting rule
- section = symbol

fuels



- air = vector of non-negative integers, first component positive
- fuel component = square matrix of non-negative integers, top left entry positive
- difference engine does component-wise comparison of upper and lower pipe's outputs:
  - $u_i \geq l_i$, for $1 \leq i \leq n$
  - for main chambers, $u_1 > l_1$.
- matrix interpretation for SRS

example



corresponds to the SRS

$$00 \to 010$$

which allows the interpretation

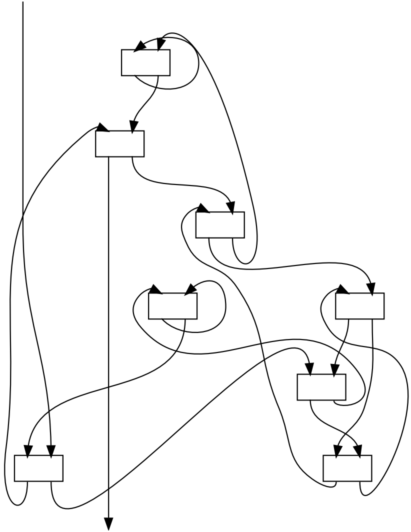
$$[\![0]\!] = \begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix}, [\![1]\!] = \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix}$$

$$[\![00]\!] - [\![010]\!] = \begin{pmatrix} 2 & 1 \\ 1 & 1 \end{pmatrix} - \begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix}$$

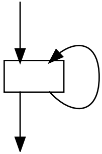
obfuscation: ternary encoding

- self-delimiting ternary code
- type-driven encoding
- lists:
  - $[\![\text{nil}]\!] = 0$, $[\![x : \text{nil}]\!] = 1[\![x]\!]$
  - $[\![x]\!] = 2[\![\text{len}(x) - 2]\!][\![x_1]\!] \ldots [\![x_{\text{len}(x)}]\!]$
- raw trit: $[\![t]\!] = t$
- natural numbers: $[\![n]\!] = [\![raw(n)]\!]$
  - $raw(0) = \text{nil}$
  - $raw(n) = (n-1) \bmod 3 : raw((n-1) \operatorname{div} 3)$
  - 0, 10, 11, 12, 2 0 00. . . 2 0 22, 2 10 000. . . 2 10 222, . . .
- tuples: $[\![(a, b)]\!] = [\![a]\!][\![b]\!]$
- types: car = $[([\mathbb{N}], \mathbb{N}, [\mathbb{N}])]$, fuel = $[[[\mathbb{N}]]]$
- contestants had to reverse engineer the encoding

obfuscation: circuits



obfuscation: circuits



- one input, one output
- each gate has left and right inputs and outputs
- every output is connected to exactly one input
- left: $(l - r) \bmod 3$, right: $(l \cdot r - 1) \bmod 3$
- backwards wires are delayed
- contestants had to deduce the gate semantics
- exercise: this gate is a base for ternary circuits with equal number of inputs and outputs, similar to NAND for boolean circuits

numbers

- 871 teams
- 214 teams figured out the circuit
- 146 teams submitted valid fuels
- 3,746 submitted cars
- 257,901 fuels (i.e. correct solutions)
- 350,344 bytes: max fuel (circuit description)
- 22,889 bytes: max car (ternary string)

matrix solvers

- linear programming for 1-dimensional matrices
- simulated annealing for higher dimensions
- similar randomized hill climbing approaches
- no SAT encoding?

problem generation

- fix random interpretation, then generate rules. prefer length increasing rules. (Carl Witty)
  $5305453 \succeq 5510450343$
  $5412501 \succ 3343403001$
- some systematic constructions
  $0121 \succ 1211012012,$
  $0121 \succ 1211012012012, \ldots$
- encoding of diophantine equations
- but also lots of trivial cars

diophantine equations

idea: use rules to enforce the matrix interpretation

$$[\![0]\!] = \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix}, [\![1]\!] = \begin{pmatrix} 1 & 1 \\ 0 & 0 \end{pmatrix}, [\![2]\!] = \begin{pmatrix} 1 & 0 \\ 0 & 2 \end{pmatrix},$$

$$[\![3]\!] = \begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix}, [\![4]\!] = \begin{pmatrix} a & 0 \\ 0 & 0 \end{pmatrix}, [\![5]\!] = \begin{pmatrix} b & 0 \\ 0 & 0 \end{pmatrix}.$$

- rules: $0 \approx 00$, $0 \approx 10$, $0 \approx 20$, $1 \approx 01$, $1 \approx 11$, $0 \approx 030$, $0 \approx 1220$, $1 \approx 121$, $4 \approx 40$, $5 \approx 50$, $1231 \approx 1331$, $12231 \approx 133331$, $122231 \approx 1333333331$, ...
- $44 \succ 4$ ($a > 1$), $55 \succ 5$ ($b > 1$), $1231 \succ 131$, $131 \succ 1$,
- then add an aditional rule encoding $a \cdot b = n$: $45 \approx 122323222322 \ldots 2232322323230$
- targeted at matrix interpretations
- weaknesses even in that context

ICFP problems in termcomp

problem submissions:

- full problems to SRS/Relative
  these are all terminating, with small certificates
- without relative rules to SRS/Complexity
  probably all linear.

Current termination/complexity provers perform poorly.

a complexity challenge

Conjecture

If an SRS has a compatible E_1/P_1 interpretation, then (relative) derivational complexity is linear.

Example

The following SRS with quadratic derivational complexity

$$ab \to ba$$

has no compatible E_1/P_1 interpretation: Assume

$$AB \succ BA.$$

Then $tr(AB) > tr(BA)$, but in fact $tr(AB) = tr(BA)$.

`http://icfpcontest.org/`