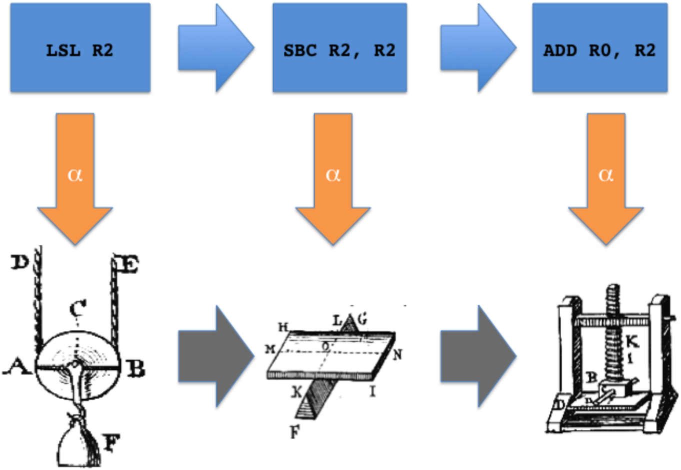
# Transfer Function Synthesis without Quantifier Elimination

Jörg Brauer and Andy King

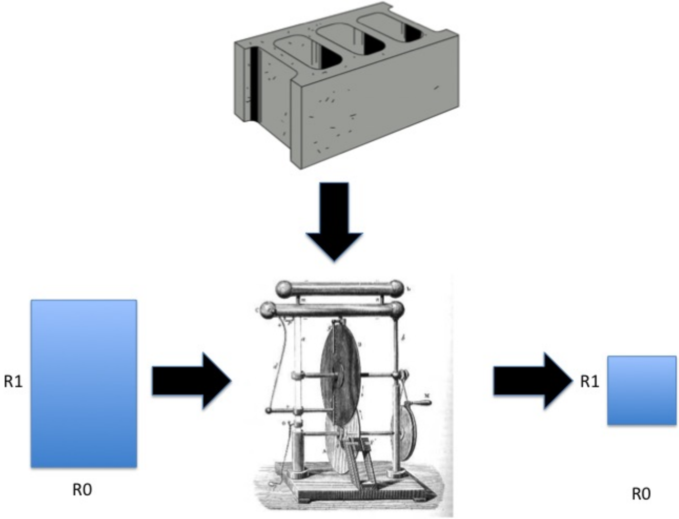
RWTH Aachen University

Portcullis Computer Security

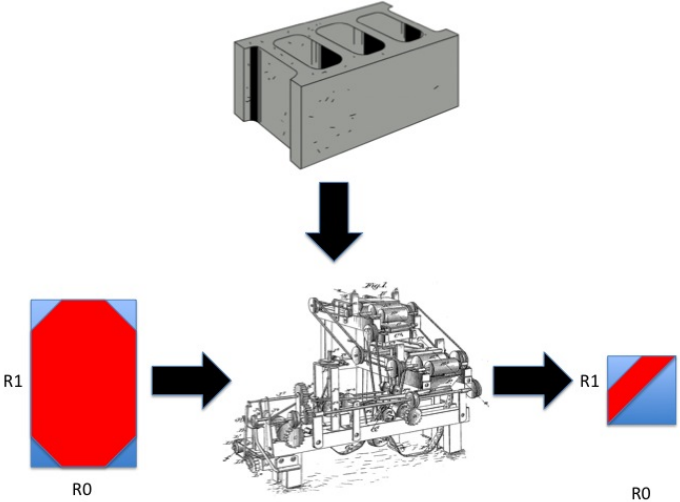

# One instruction at a time abstraction by transfer function lookup



# Transfer function synthesis with $\exists_x$ and $\forall_x$ [SAS'10]



# Transfer function synthesis without $\exists_x$ and $\forall_x$



# Feasible mode combinations

# Transfer functions as systems of guarded updates

- Consider the following:

  $1:$ `ADD R0, R1`   $2:$ `MOV R2, R0`   $3:$ `EOR R2, R1`   $4:$ `LSL R2`
  $5:$ `SBC R2, R2`   $6:$ `ADD R0, R2`   $7:$ `EOR R0, R2`

- Implements `R0':= isign(R0+R1,R1)` where `isign` assigns `abs(R0+R1)` to `R0` if `R1` $\geq 0$ and `-abs(R0+R1)` otherwise
- Need to extract cases:
  - Cases which are there by design: `R1` $\geq 0$
  - Cases which are implementation artefacts: when `abs` is applied to $-2^{31}$ then the result is $2^{31}$ subject to overflow which is $-2^{31}$

# Modes of `ADD R0, R1, LSL R2` and `ADD R0, R2`

- Let $\mu$ (mu) denote a Boolean encoding of `ADD R0, R1` over bit-vectors $\{\vec{r0}, \vec{r1}, \ldots\}$ obtained through SSA and

$$
\begin{aligned}
\mu_O &= \mu \wedge \neg \vec{r0}[31] \wedge \neg \vec{r1}[31] \wedge \vec{r0}'[31] \\
\mu_U &= \mu \wedge \vec{r0}[31] \wedge \vec{r1}[31] \wedge \neg \vec{r0}'[31] \\
\mu_E &= \mu \wedge (\vec{r0}[31] \vee \vec{r1}[31] \vee \neg \vec{r0}'[31]) \wedge (\neg \vec{r0}[31] \vee \neg \vec{r1}[31] \vee \vec{r0}'[31])
\end{aligned}
$$

- Let $\nu_O$ and $\nu_E$ (nu) express the overflow and exact modes of `LSL R2`.
- In an analogous way to the first `ADD`, let $\eta_O$, $\eta_U$ and $\eta_E$ express the semantics of `ADD R0, R2`.

# Composing modes for whole block

- Using these encodings that satisfy a single mode, we can compose a formula for a fixed mode-combination.
- The combination of $\mu_U$, $\nu_E$ and $\eta_E$ is infeasible
- The above block constitutes $3 \cdot 2 \cdot 3 = 18$ combinations of modes, but only five of which are satisfiable
- We derive a guard and update only for the feasible mode-combinations

# Synthesising guards

# Deriving guards with dichotomic search

- Consider the case where (1) underflows, (4) overflows and (6) is exact, with the corresponding formula denoted $\pi$
- To derive an octagonal guard for $\pi$, consider the problem of computing least $d$ such that $-\langle\!\langle \vec{r0} \rangle\!\rangle - \langle\!\langle \vec{r1} \rangle\!\rangle \leq d$
- Let $\kappa$ be a formula encodes $\langle\!\langle \vec{d} \rangle\!\rangle = -\langle\!\langle \vec{r0} \rangle\!\rangle - \langle\!\langle \vec{r1} \rangle\!\rangle$ where $\vec{d}$ is signed and $\kappa$ is extended to 34 bits to prevent wraps

# Maximising $-2^{33} \leq d < 2^{33}$ bit-by-bit

- Then check:

$$\psi^1 = \pi \wedge \kappa \wedge \neg \vec{d}[33]$$

- Satisfiability of $\psi^1$ shows $0 \leq d < 2^{33}$
- Then check:

$$\psi^2 = \pi \wedge \kappa \wedge \neg \vec{d}[33] \wedge \vec{d}[32]$$

- Satisfiability of $\psi^2$ shows $2^{32} \leq d < 2^{33}$
- Then check:

$$\psi^3 = \pi \wedge \kappa \wedge \neg \vec{d}[33] \wedge \vec{d}[32] \wedge \vec{d}[31]$$

- Unsatisfiability of $\psi^3$ shows $2^{32} \leq d < 2^{32} + 2^{31}$
- Continuing in this way we infer $2^{32} \leq d < 2^{32} + 1$.

# Just for the record

Repeating this tactic for all five feasible mode-combinations:

$$
\begin{array}{rcl}
g_{O^{(1)},O^{(4)},U^{(6)}} & = & 2^{31} \leq \langle\langle \vec{r0} \rangle\rangle + \langle\langle \vec{r1} \rangle\rangle \leq 2^{31} \quad \wedge \quad 0 \leq \langle\langle \vec{r1} \rangle\rangle \leq 2^{31}-1 \\
g_{E^{(1)},E^{(4)},E^{(6)}} & = & -2^{31} \leq \langle\langle \vec{r0} \rangle\rangle + \langle\langle \vec{r1} \rangle\rangle \leq 2^{31}-1 \\
g_{U^{(1)},O^{(4)},E^{(6)}} & = & \underline{-2^{32}} \leq \langle\langle \vec{r0} \rangle\rangle + \langle\langle \vec{r1} \rangle\rangle \leq -2^{31}-1 \\
g_{E^{(1)},O^{(4)},E^{(6)}} & = & 0 \leq \langle\langle \vec{r0} \rangle\rangle + \langle\langle \vec{r1} \rangle\rangle \leq 2^{31}-1 \quad \wedge -2^{31} \leq \langle\langle \vec{r1} \rangle\rangle \leq 1 \\
g_{O^{(1)},O^{(4)},E^{(6)}} & = & 2^{31}+1 \leq \langle\langle \vec{r0} \rangle\rangle + \langle\langle \vec{r1} \rangle\rangle \leq 2^{32}
\end{array}
$$

Redundant inequalities are omitted for clarity of presentation

# Synthesising updates

# Consider `ADD R0 R1; LSL R0` in exact modes

We want an update to map octagonal input constraints with symbolic constants to octagonal outputs with symbolic constants:

$$\left\{ \begin{array}{r} \langle\!\langle \vec{r0} \rangle\!\rangle \leq d_1 \\ \langle\!\langle \vec{r1} \rangle\!\rangle \leq d_2 \\ -\langle\!\langle \vec{r0} \rangle\!\rangle \leq d_3 \\ -\langle\!\langle \vec{r1} \rangle\!\rangle \leq d_4 \\ \langle\!\langle \vec{r0} \rangle\!\rangle + \langle\!\langle \vec{r1} \rangle\!\rangle \leq d_5 \\ -\langle\!\langle \vec{r0} \rangle\!\rangle - \langle\!\langle \vec{r1} \rangle\!\rangle \leq d_6 \\ -\langle\!\langle \vec{r0} \rangle\!\rangle + \langle\!\langle \vec{r1} \rangle\!\rangle \leq d_7 \\ \langle\!\langle \vec{r0} \rangle\!\rangle - \langle\!\langle \vec{r1} \rangle\!\rangle \leq d_8 \end{array} \right\} \rightsquigarrow \left\{ \begin{array}{r} \langle\!\langle \vec{r0}' \rangle\!\rangle \leq 2d_5 \\ \langle\!\langle \vec{r1}' \rangle\!\rangle \leq d_2 \\ -\langle\!\langle \vec{r0}' \rangle\!\rangle \leq 2d_6 \\ -\langle\!\langle \vec{r1}' \rangle\!\rangle \leq d_4 \\ \langle\!\langle \vec{r0}' \rangle\!\rangle + \langle\!\langle \vec{r1}' \rangle\!\rangle \leq 2d_5 + d_2 \\ -\langle\!\langle \vec{r0}' \rangle\!\rangle - \langle\!\langle \vec{r1}' \rangle\!\rangle \leq 2d_6 + d_4 \\ -\langle\!\langle \vec{r0}' \rangle\!\rangle + \langle\!\langle \vec{r1}' \rangle\!\rangle \leq 2d_6 + d_2 \\ \langle\!\langle \vec{r0}' \rangle\!\rangle - \langle\!\langle \vec{r1}' \rangle\!\rangle \leq 2d_5 + d_4 \end{array} \right\}$$

# Consider $\langle\langle \vec{r0}' \rangle\rangle \leq d_1'$ and the problem of discovering a relationship between $d_1'$ and $d_1, \ldots, d_8$

- Let $\vec{d_1}, \ldots, \vec{d_8}$ denote signed 34-bit vectors that represent the symbolic constants $d_1, \ldots, d_8$
- Let $\kappa$ denote a formula that holds iff the 8 inequalities $\langle\langle \vec{r0} \rangle\rangle \leq \langle\langle \vec{d_1} \rangle\rangle, \ldots, \langle\langle \vec{r0} \rangle\rangle - \langle\langle \vec{r1} \rangle\rangle \leq \langle\langle \vec{d_8} \rangle\rangle$ simultaneously hold
- Let $\pi$ denote a propositional encoding for `ADD R0 R1; LSL R0` operating in exact mode
- Let $\rho$ encode the *equality* $\langle\langle \vec{r0}' \rangle\rangle = \langle\langle \vec{d_1'} \rangle\rangle$ where $\vec{d_1'}$ is a signed bit-vector representing $d_1'$

# Step i: solving and maximisation

- Present $\kappa \wedge \pi \wedge \rho$ to a SAT solver and find a model:

$$m_1 = \left\{ \langle\!\langle \vec{d_1'} \rangle\!\rangle = 1, \langle\!\langle \vec{d_1} \rangle\!\rangle = 1, \langle\!\langle \vec{d_2} \rangle\!\rangle = 1, \ldots, \langle\!\langle \vec{d_7} \rangle\!\rangle = 1, \langle\!\langle \vec{d_8} \rangle\!\rangle = 1 \right\}$$

- $\langle\!\langle \vec{d_1'} \rangle\!\rangle = 1$ may not be maximum for $\langle\!\langle \vec{d_1} \rangle\!\rangle = 1, \ldots, \langle\!\langle \vec{d_8} \rangle\!\rangle = 1$
- Let $\zeta$ denote a formula that holds iff $\langle\!\langle \vec{d_1} \rangle\!\rangle = 1, \ldots, \langle\!\langle \vec{d_8} \rangle\!\rangle = 1$
- Apply dichotomic search to find the maximal value of $\langle\!\langle \vec{d_1'} \rangle\!\rangle$ subject to $\kappa \wedge \pi \wedge \rho \wedge \zeta$.
- This gives the model:

$$m_1' = \left\{ \langle\!\langle \vec{d_1'} \rangle\!\rangle = 2, \langle\!\langle \vec{d_1} \rangle\!\rangle = 1, \langle\!\langle \vec{d_2} \rangle\!\rangle = 1, \ldots, \langle\!\langle \vec{d_7} \rangle\!\rangle = 1, \langle\!\langle \vec{d_8} \rangle\!\rangle = 1 \right\}$$

# Triangularisation and resolving

- Suppose the matrix $\mathbf{M}_1$ is constructed from $m'_1$ by using the variable ordering $\langle d'_1, d_1, \ldots, d_8 \rangle$ on columns:

$$\mathbf{M}_1 = \left[\begin{array}{ccccccccc|c} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 2 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \end{array}\right]$$

- Let $\mu$ denote a formula that holds iff $\langle\langle \vec{d_8} \rangle\rangle \neq 1$ holds

# Step ii: solving and maximisation

- Present $\kappa \wedge \pi \wedge \rho \wedge \mu$ to a SAT solver and find a model:

$$m_2 = \left\{ \langle\langle \vec{d_1'} \rangle\rangle = 8, \langle\langle \vec{d_1} \rangle\rangle = 3, \langle\langle \vec{d_2} \rangle\rangle = 3, \ldots, \langle\langle \vec{d_7} \rangle\rangle = 2, \langle\langle \vec{d_8} \rangle\rangle = 0 \right\}$$

- $\langle\langle \vec{d_1'} \rangle\rangle = 8$ may not be maximum for $\langle\langle \vec{d_1} \rangle\rangle = 3, \ldots, \langle\langle \vec{d_8} \rangle\rangle = 0$
- Let $\zeta$ denote a formula that holds iff $\langle\langle \vec{d_1} \rangle\rangle = 3, \ldots, \langle\langle \vec{d_8} \rangle\rangle = 0$
- Apply dichotomic search to find the maximal value of $\langle\langle \vec{d_1'} \rangle\rangle$ subject to $\kappa \wedge \pi \wedge \rho \wedge \zeta$.
- This gives the model:

$$m_2' = \left\{ \langle\langle \vec{d_1'} \rangle\rangle = 10, \langle\langle \vec{d_1} \rangle\rangle = 3, \langle\langle \vec{d_2} \rangle\rangle = 3, \ldots, \langle\langle \vec{d_7} \rangle\rangle = 2, \langle\langle \vec{d_8} \rangle\rangle = 0 \right\}$$

# Merging [Karr'76]

- The model $m_2'$ is interpreted as a matrix:

$$\mathbf{M}_2 = \left[\begin{array}{ccccccccc|c} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 10 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 3 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 3 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 2 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{array}\right]$$

- The merge $\mathbf{M}_1 \sqcup \mathbf{M}_2$ is as follows:

$$\mathbf{M}_1 \sqcup \mathbf{M}_2 = \left[\begin{array}{ccccccccc|c} 1 & 0 & 0 & 0 & 0 & -2 & 0 & 0 & 0 & 0 \\ 0 & 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & -2 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 2 & 0 & 2 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 \end{array}\right]$$

# Step iii and iv: solving and maximisation

- Let $\mu$ now denote a formula that holds iff $\langle\langle \vec{d_7} \rangle\rangle + \langle\langle \vec{d_8} \rangle\rangle \neq 1$
- Presenting $\kappa \wedge \pi \wedge \rho \wedge \mu$ to a solver gives:

$$m_3 = \left\{ \langle\langle \vec{d'_1} \rangle\rangle = 22, \langle\langle \vec{d_1} \rangle\rangle = 8, \ldots, \langle\langle \vec{d_8} \rangle\rangle = 0 \right\}$$

- Maximising $\langle\langle \vec{d'_1} \rangle\rangle$ then gives:

$$m'_3 = \left\{ \langle\langle \vec{d'_1} \rangle\rangle = 26, \langle\langle \vec{d_1} \rangle\rangle = 8, \ldots, \langle\langle \vec{d_8} \rangle\rangle = 0 \right\}$$

- Form $\mathbf{M}_3$ and calculate another merge:

$$\mathbf{M}_1 \sqcup \mathbf{M}_2 \sqcup \mathbf{M}_3 = \left[\begin{array}{ccccccccc|c} 1 & 0 & 0 & 0 & 0 & -2 & 0 & 0 & 0 & 0 \\ 0 & 1 & -1 & 1 & -1 & 0 & 0 & 0 & 0 & 0 \end{array}\right]$$

- Repeating

$$\mathbf{M}_1 \sqcup \mathbf{M}_2 \sqcup \mathbf{M}_3 \sqcup \mathbf{M}_4 = \left[\begin{array}{ccccccccc|c} 1 & 0 & 0 & 0 & 0 & -2 & 0 & 0 & 0 & 0 \end{array}\right]$$

- Conclude $d'_1 = 2d_5$

# "With" versus "without" for intervals

| block | insts | bits | runtime | | | |
|---|---|---|---|---|---|---|
| | | | guards/SAT | affine/SAT | overall | SAS'10 |
| `inc` | 1 | 8 | 0.2s / 40 | 0.1s / 5 | 0.3s | 0.2s |
| | | 32 | 0.5s / 136 | 0.2s / 5 | 1.0s | 23.0s |
| `shifter` | 2 | 8 | 0.3s / 60 | 0.1s / 8 | 0.4s | 0.3s |
| | | 32 | 0.8s / 216 | 0.2s / 8 | 1.0s | $\infty$ |
| `swap` | 3 | 8 | — | 0.1s / 3 | 0.1s | 0.1s |
| | | 32 | — | 0.1s / 3 | 0.1s | 0.2s |
| `flipper` | 4 | 8 | 0.2s / 40 | 0.2s / 5 | 0.4s | 0.5s |
| | | 32 | 0.9s / 216 | 0.3s / 5 | 1.2s | $\infty$ |
| `abs` | 5 | 8 | 2.5s / 216 | 0.3s / 8 | 2.8s | 0.8s |
| | | 32 | 6.5s / 792 | 0.3s / 8 | 6.8s | $\infty$ |
| `isign` | 7 | 8 | 4.1s / 360 | 0.2s / 18 | 4.3s | 4.5s |
| | | 32 | 10.7s / 1320 | 0.4s / 18 | 11.1s | $\infty$ |
| `absolute` | 10 | 8 | 2.8s / 216 | 0.3s / 8 | 3.1s | 9.5s |
| | | 32 | 7.2s / 792 | 0.3s / 8 | 7.5s | $\infty$ |