Register-Bounded Synthesis

Ayrat Khalimov
Universite libre de Bruxelles
Belgium

Orna Kupferman
Hebrew University
Israel

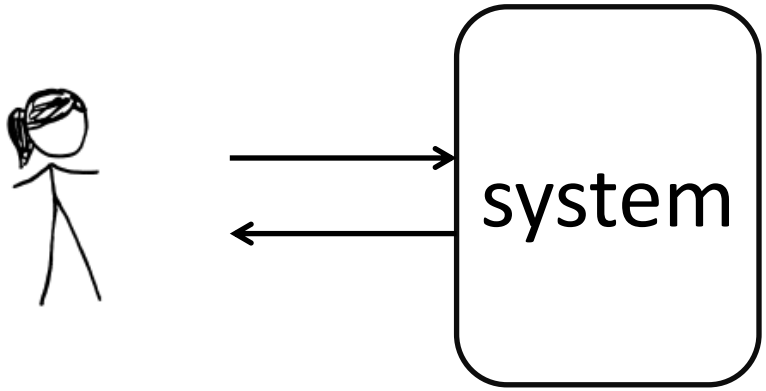
	DRA	NRA	URA	$\mathsf{NRA}_{\mathsf{tf}}$
Bounded Synthesis	ExpSpace (Thm. 13)	Undecidable ($k \geq 1$) (Thm. 3)	2ExpTime ([9] and Thm. 12)	2ExpTime (Thm. 16)
General Case	ExpSpace (Thm. 6)	Undecidable (Thm. 2)	Undecidable (Thm. 4)	Open

Why study *bounded* synthesis from *universal* automata?

why register-*bounded* synthesis?

- Not a limitation: designer usually knows the sensible bound on the number of registers
- Added benefit: small programs

why *universal* register automata?

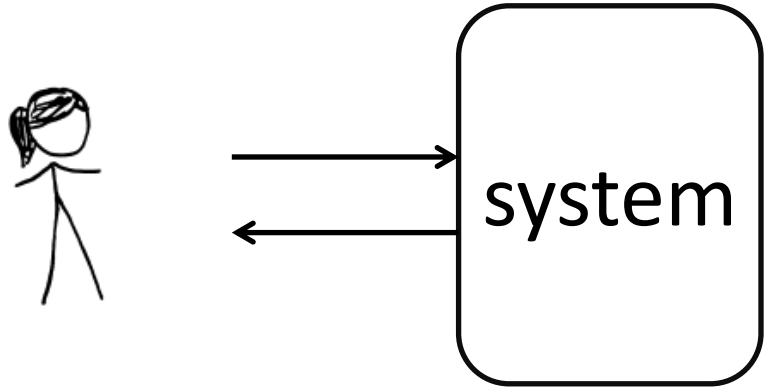


All computations

$(in_0, out_0)\ (in_1, out_1)\ (in_2, out_2)\ \ldots$

satisfy a given specification.

why *universal* register automata?



All computations

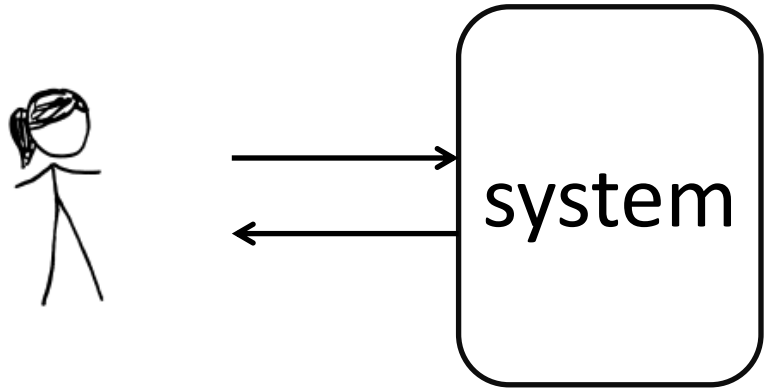
$(in_0, out_0)\ (in_1, out_1)\ (in_2, out_2)\ \ldots$

satisfy a given specification.

Most specifications are derived from arbiter:

$$\forall d \in D\colon \mathbf{G}(req \wedge i = d \ \rightarrow\ \mathbf{X}\,\mathbf{F}(grant \wedge o = d))$$

why *universal* register automata?



All computations

$(in_0, out_0)\ (in_1, out_1)\ (in_2, out_2)\ \ldots$

satisfy a given specification.

Most specifications are derived from arbiter:

$\forall d \in D\colon \mathbf{G}(req \wedge i = d \ \rightarrow\ \mathbf{X}\,\mathbf{F}(grant \wedge o = d))$

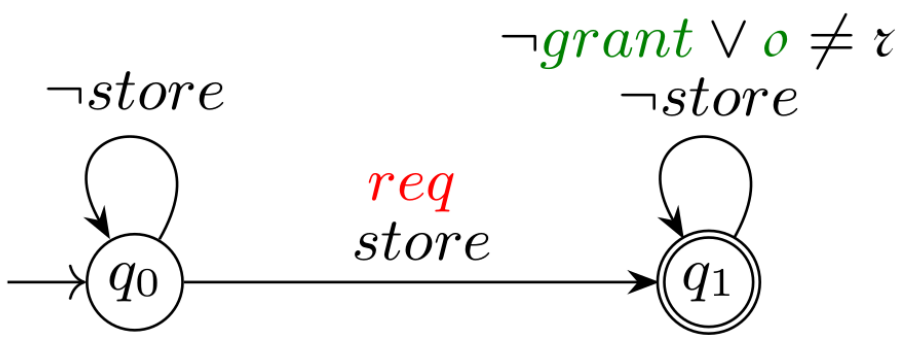
Universal register automata *can* express this.

Nondeterministic -- cannot.

universal register automaton

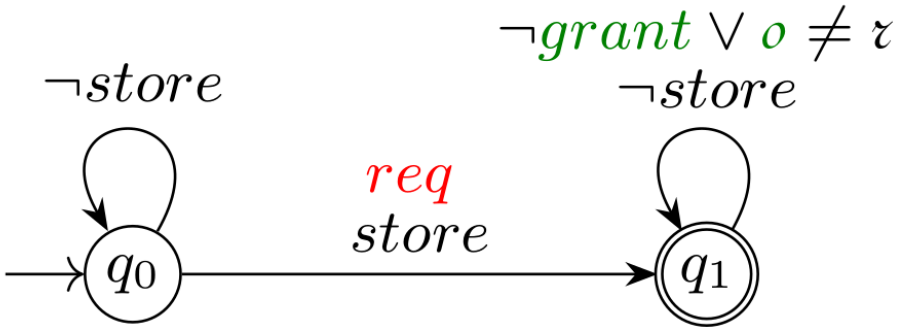
- Works on words in $(\Sigma \times D \times D)^\omega$
- Registers $R = \{r_1, \ldots, r_{k_A}\}$, initialized v_0
- Transition function

$$Q \times \Sigma \times Tst_i \times Tst_o \to 2^{Q \times Asgn}$$

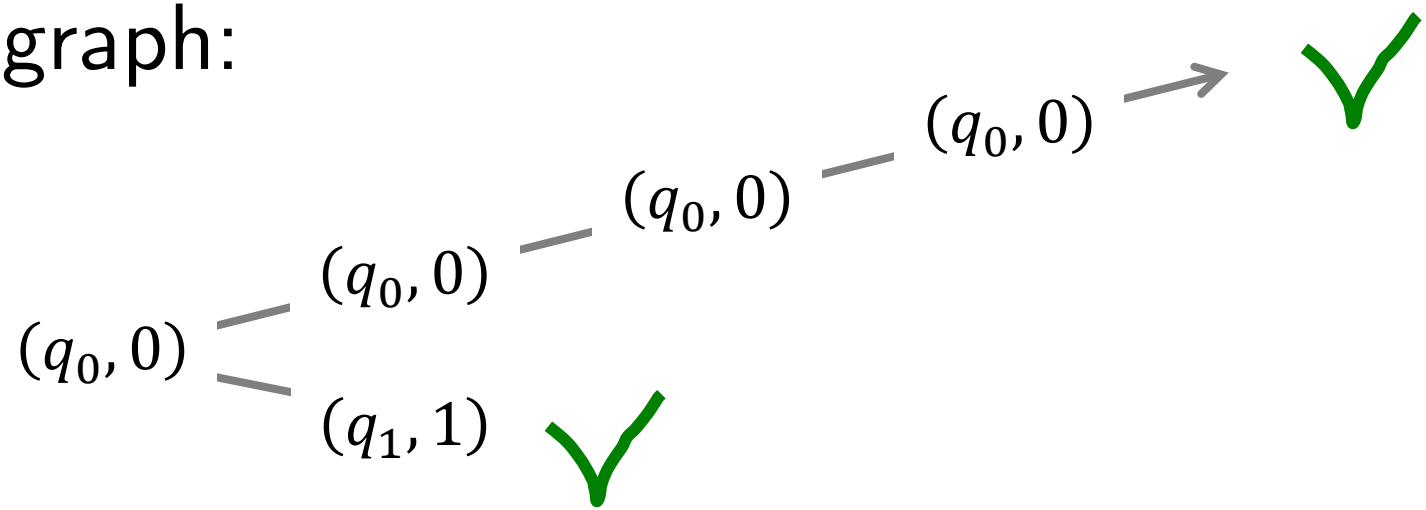


Arbiter specification (coBuchi)

universal register automaton



- Word: $\begin{pmatrix} req,1 \\ \neg grant,0 \end{pmatrix} \begin{pmatrix} \neg req,2 \\ grant,1 \end{pmatrix} \begin{pmatrix} \neg req,3 \\ \neg grant,1 \end{pmatrix} \ldots$
- Run-graph:

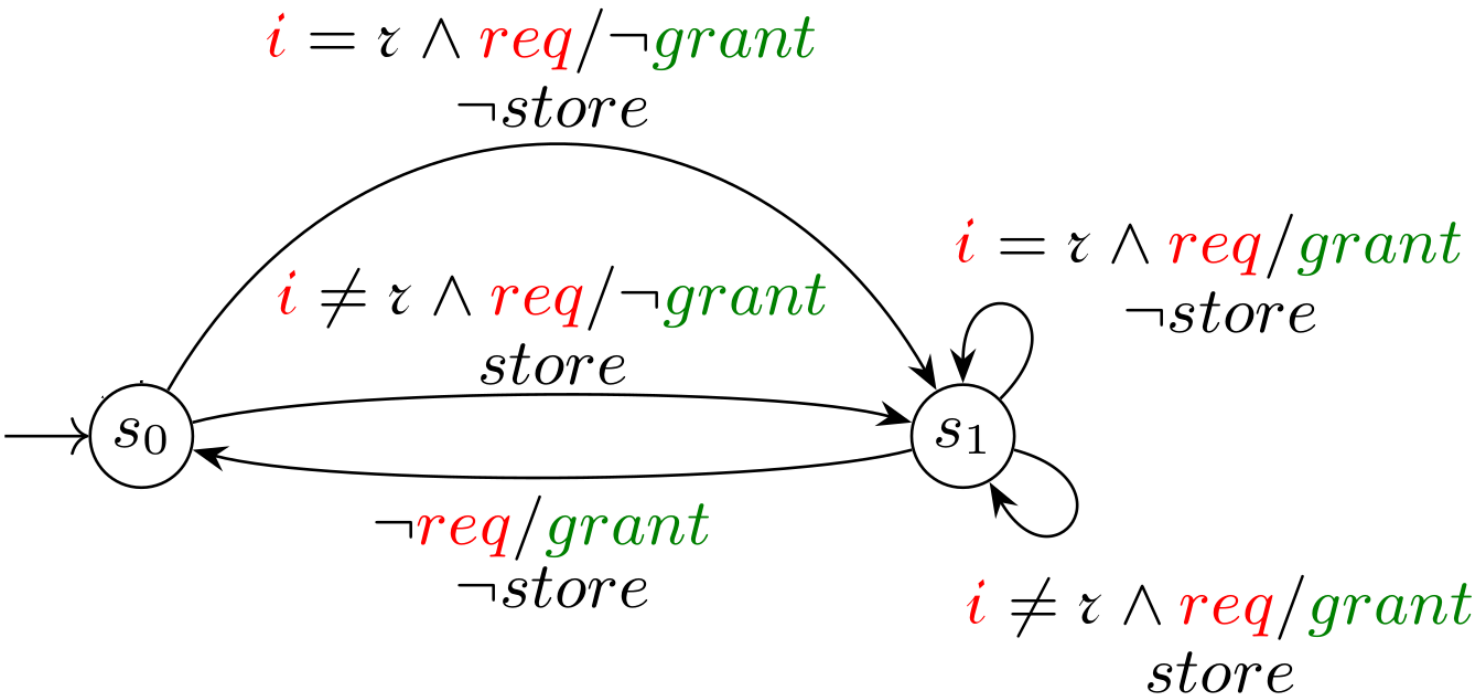


register transducer

- Reads a letter in $\Sigma_I \times D$
- Outputs a letter in $\Sigma_O \times D$
- Registers $R = \{r_1, \ldots, r_{k_s}\}$, initialized with v_0
- Transition function

$$S \times \Sigma_I \times Tst_i \to S \times \Sigma_O \times R \times Asgn$$

arbiter



- Input: $(req, 1)(\neg req, 2)(\neg req, 3) \ldots$
- Run: $(s_0, 0) \overset{\neg g, 0}{\Longrightarrow} (s_1, 1) \overset{g, 1}{\Rightarrow} (s_0, 1) \overset{\neg g, 1}{\Longrightarrow} (s_0, 1) \ldots$

bounded synthesis problem

Given:

- Σ_I, Σ_O
- universal register automaton A over $\Sigma_I \times \Sigma_O \times D \times D$
- the number k_s of system registers

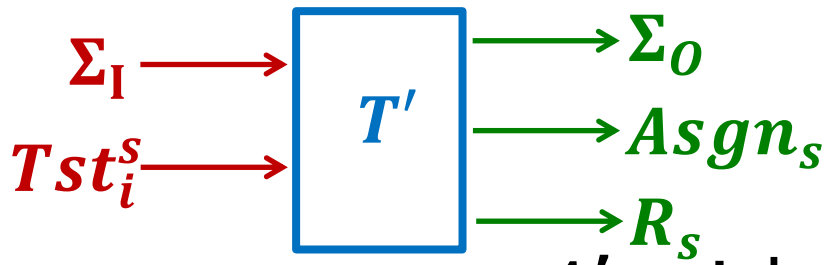
Return:

- k_s-register transducer T such that $T \models A$, or "unrealizable"

Bounded synthesis problem is solvable in EXP in $|Q|$ and k_s, and 2EXP in k_A.

abstraction A'

$$T\colon S \times \Sigma_I \times Tst_i^s \to S \times \Sigma_O \times R_s \times Asgn_s$$
$$T'\colon S \times \quad \Sigma_I' \quad \to S \times \quad \Sigma_O'$$



We construct register-*less* automaton A' with

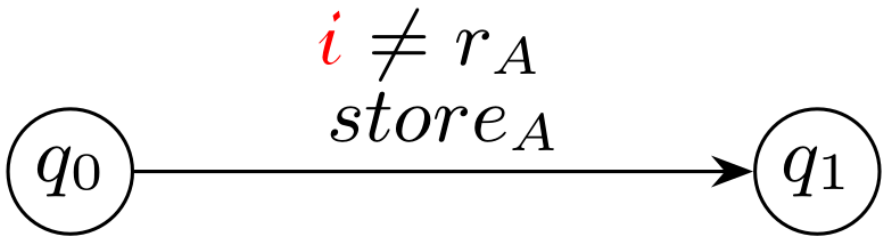
$$Q' \times (\Sigma_I' \times \Sigma_O') \to 2^{Q'}$$

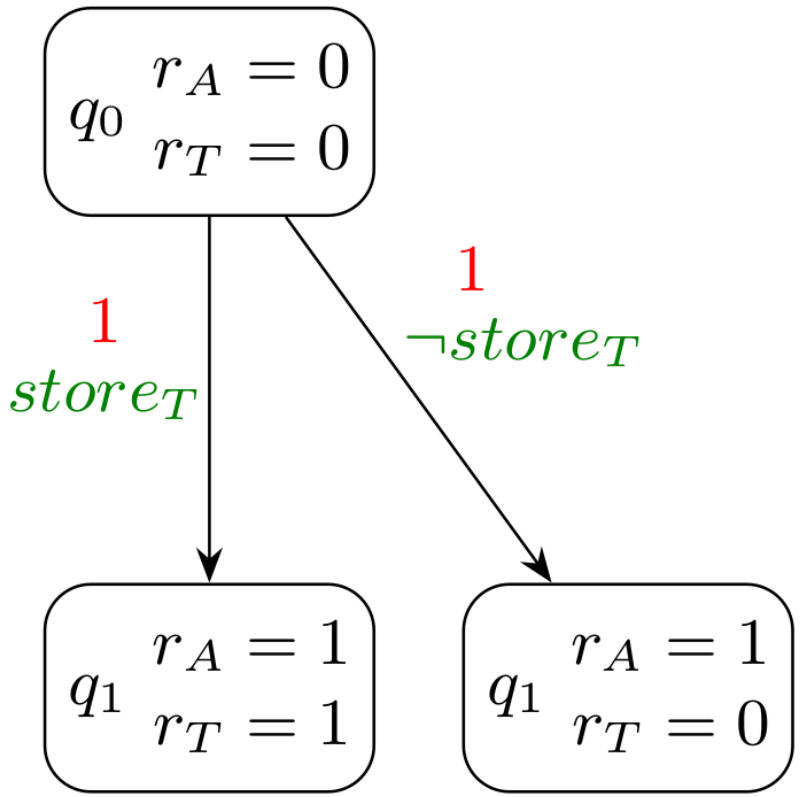
such that

$$T' \vDash A' \text{ iff } T \vDash A$$

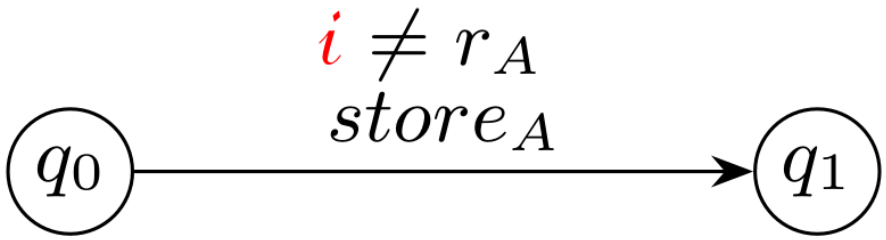
for every T or T'.

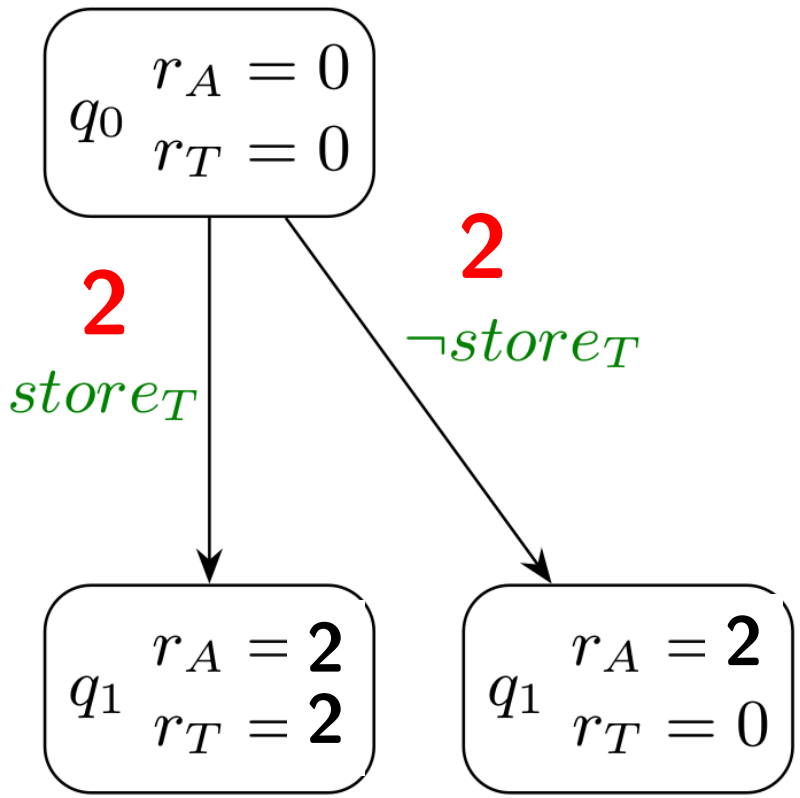
abstracting a single transition



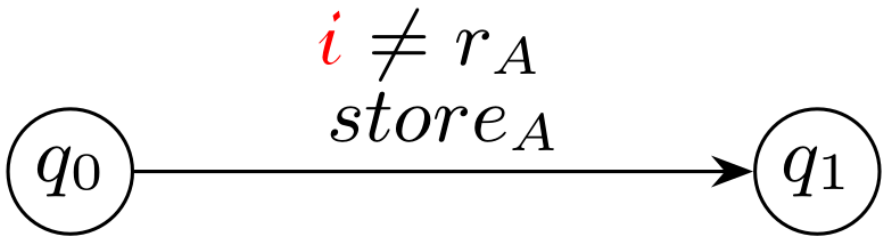


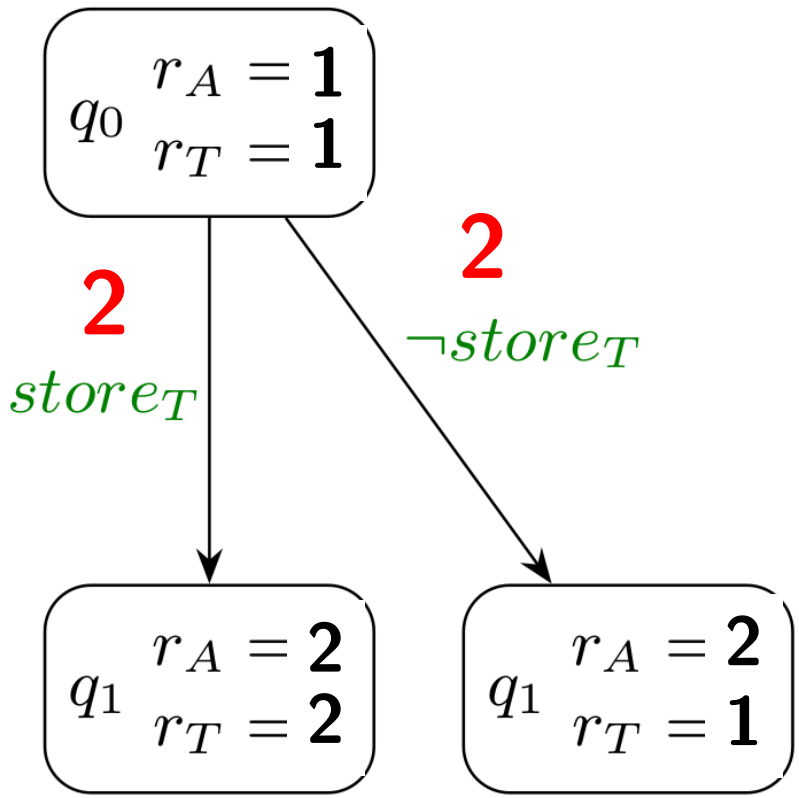
abstracting a single transition



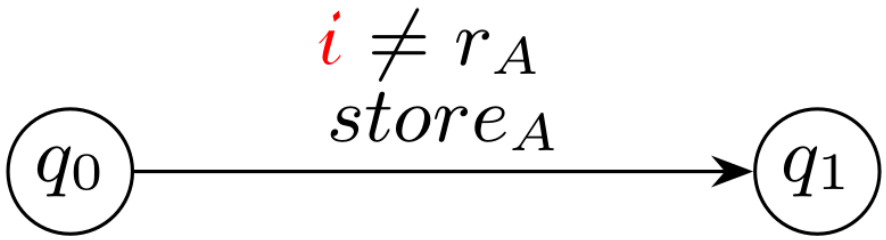


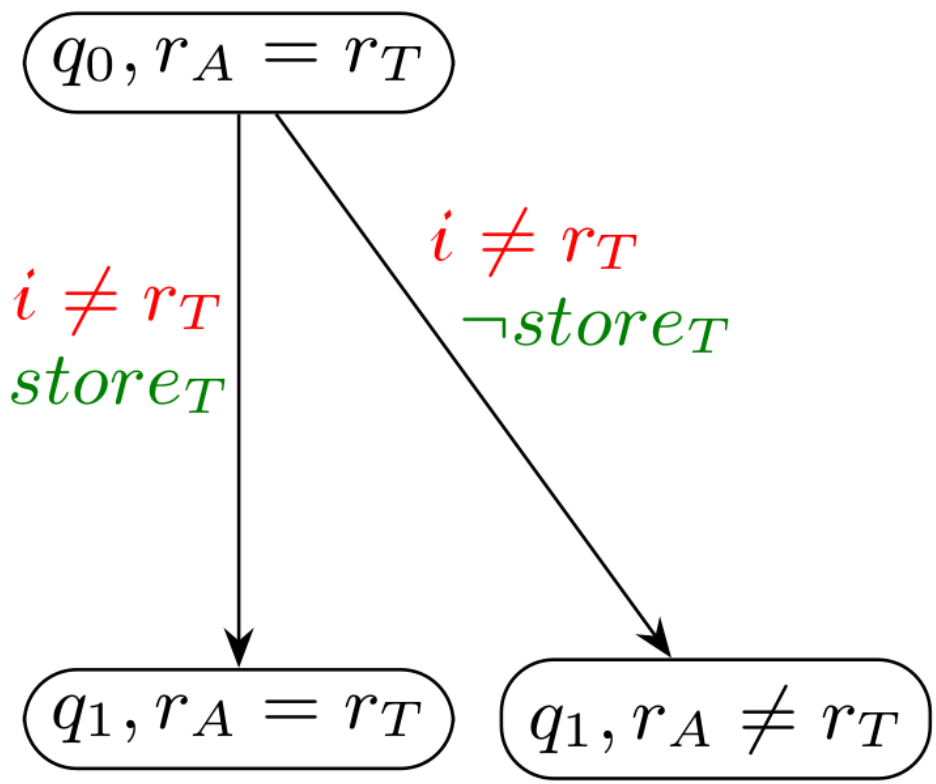
abstracting a single transition



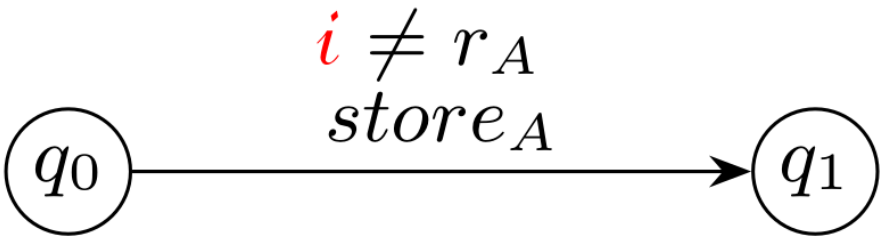


abstracting a single transition



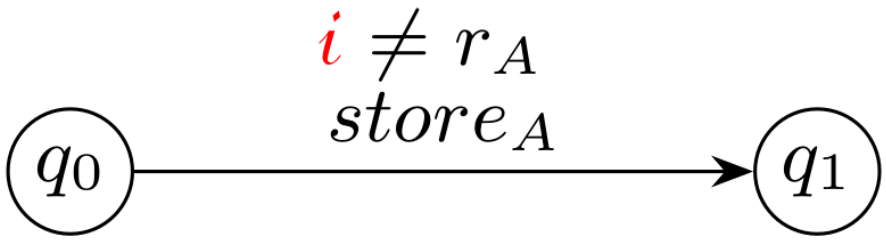


abstracting a single transition



$$q_0, r_A \neq r_T$$

abstracting a single transition

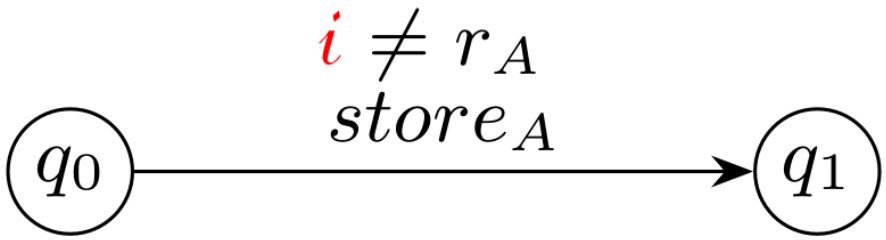


Two possibilities:

- $i = r_T$
- $i \neq r_T$

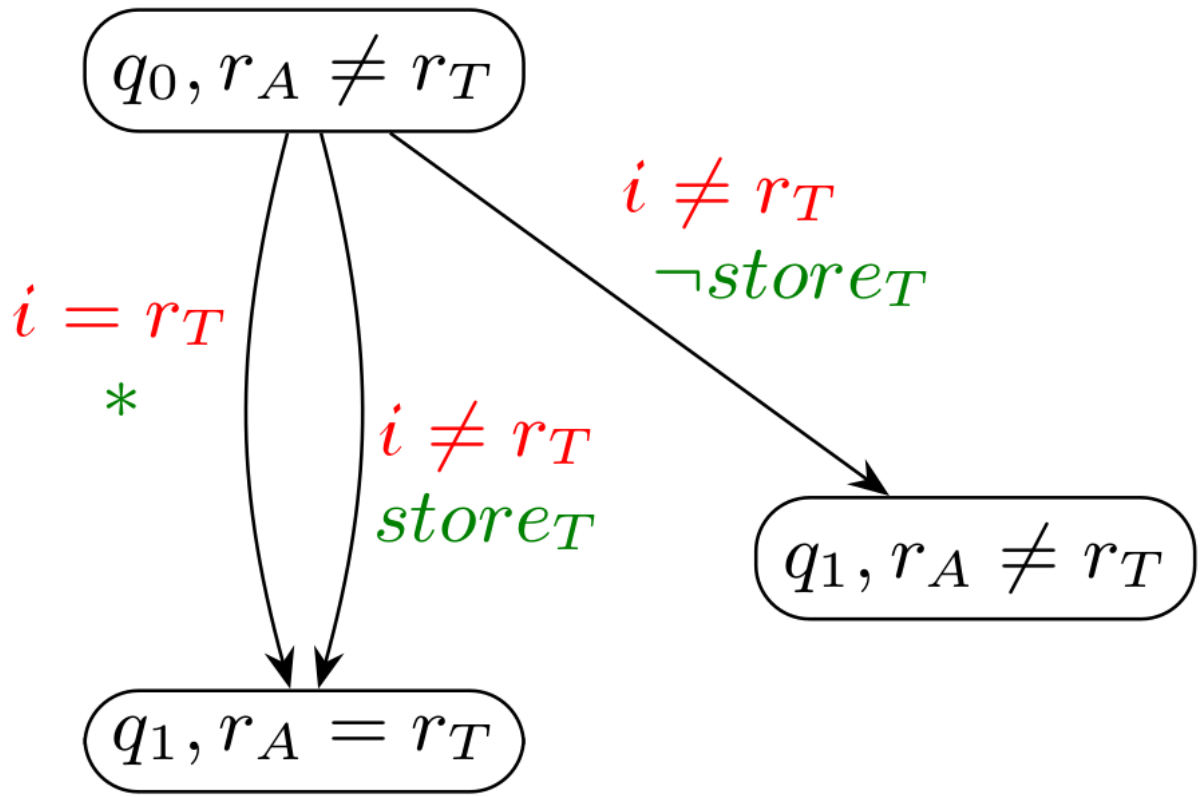
$q_0, r_A \neq r_T$

abstracting a single transition

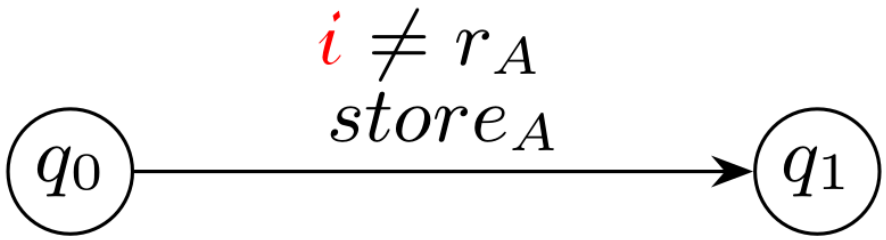


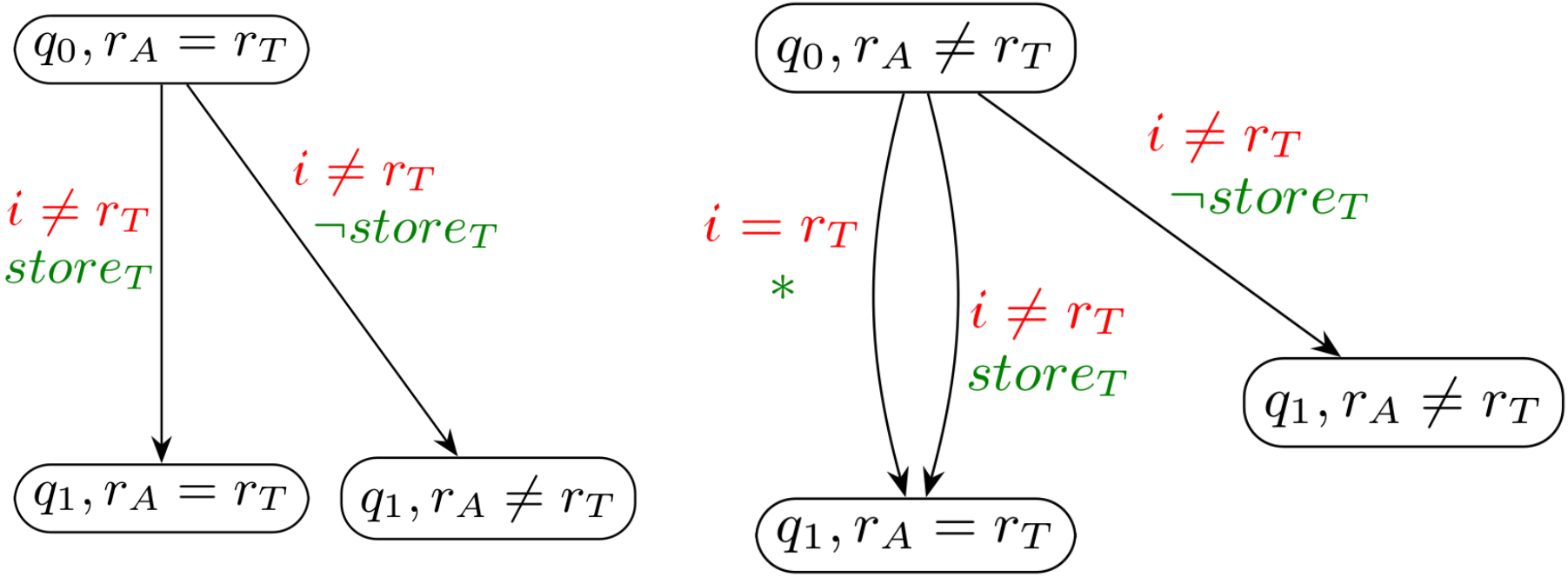
Two possibilities:

- $i = r_T$
- $i \neq r_T$

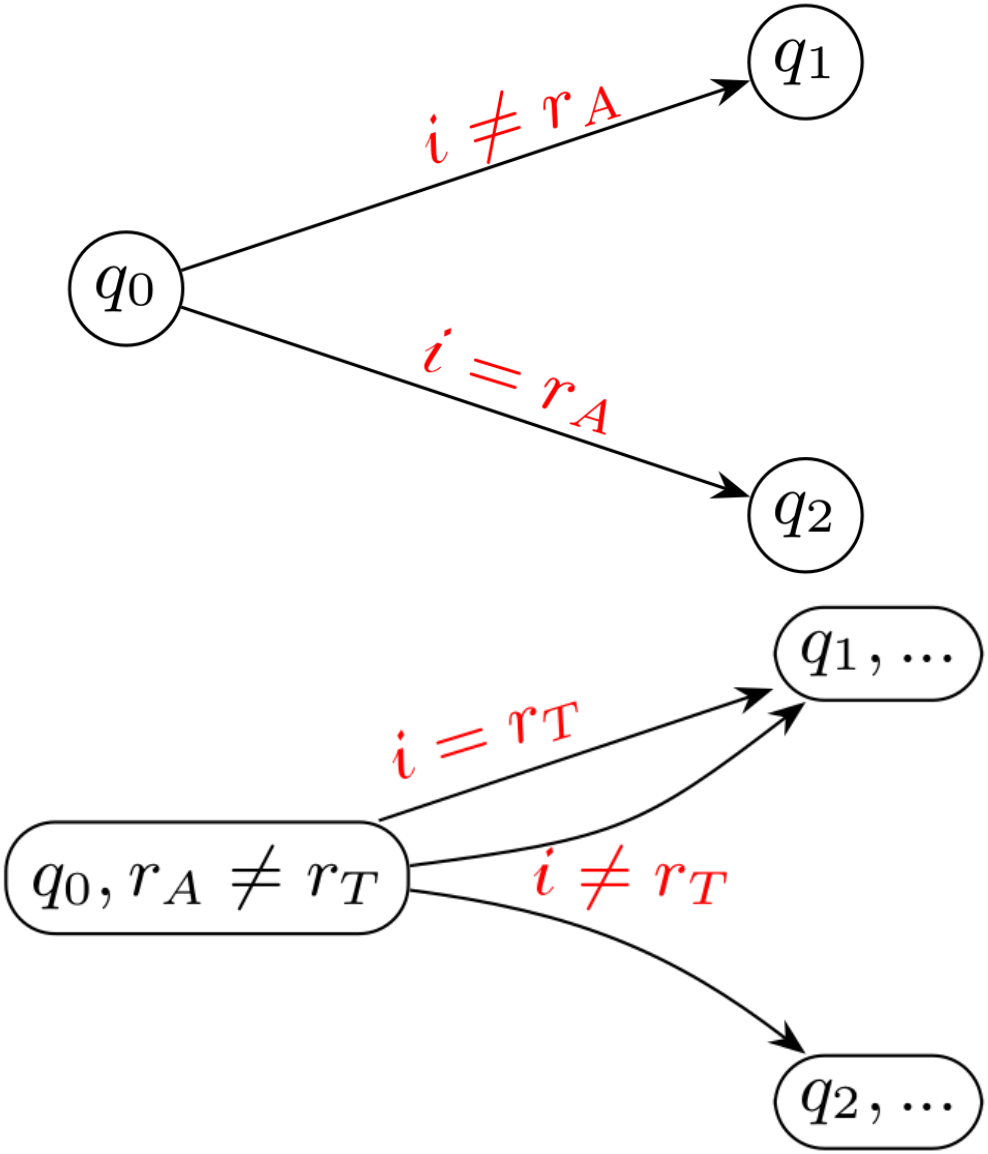


abstracting a single transition

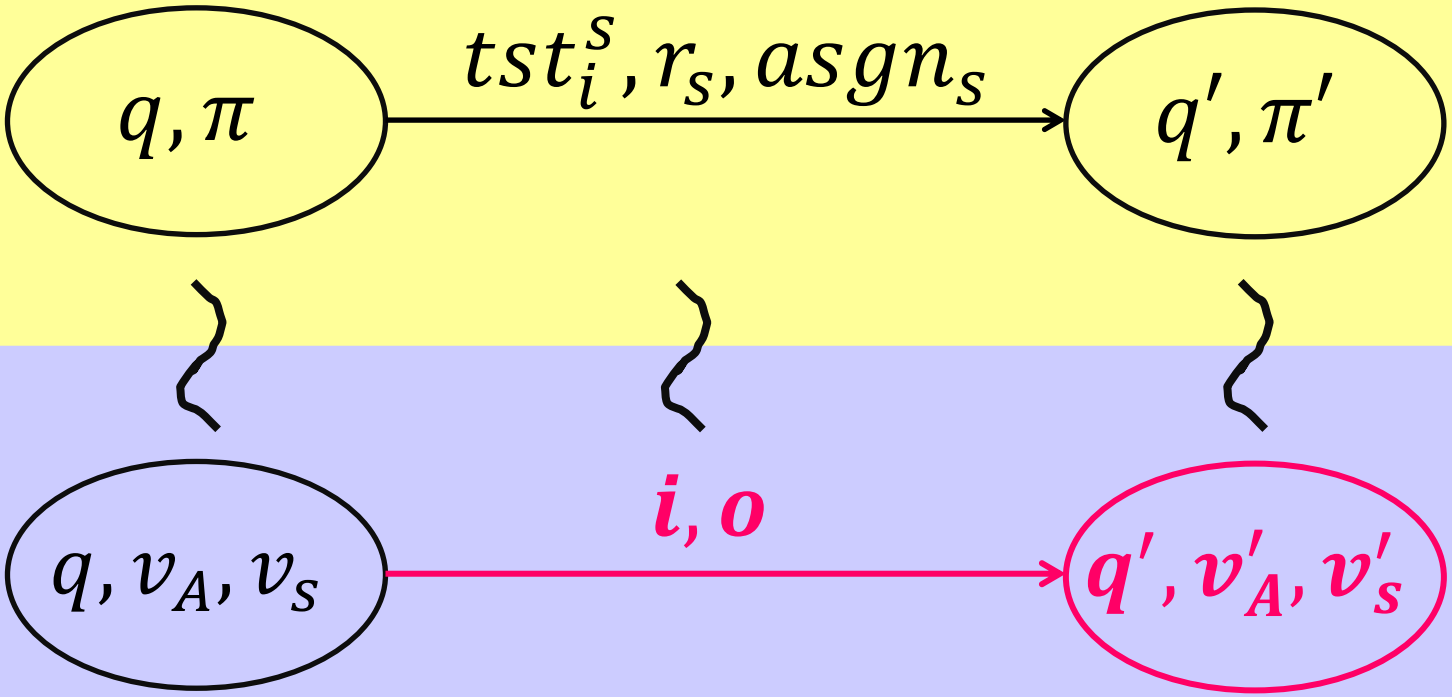




one tst_s can induce several tst_A



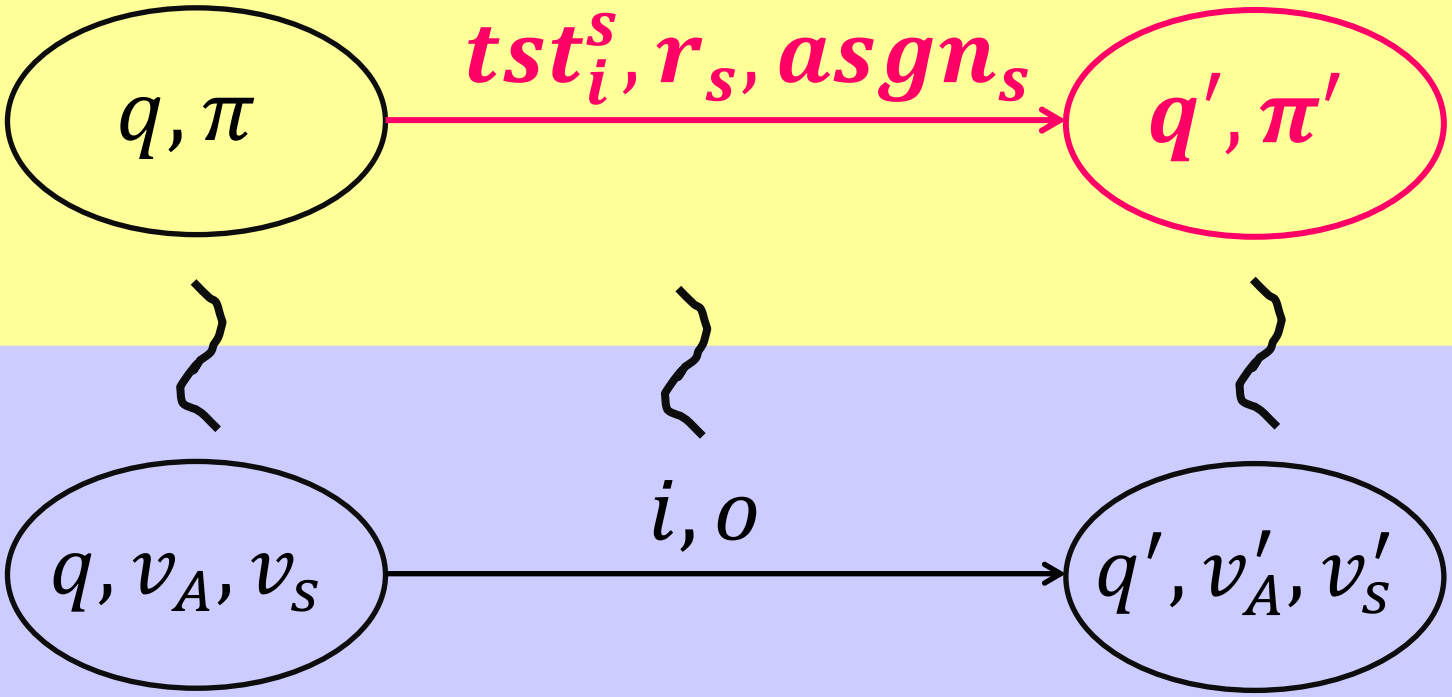
bisimulation property of the abstraction



transition of A' and some T'

transition of A and some T

bisimulation property of the abstraction



transition of A' and some T'

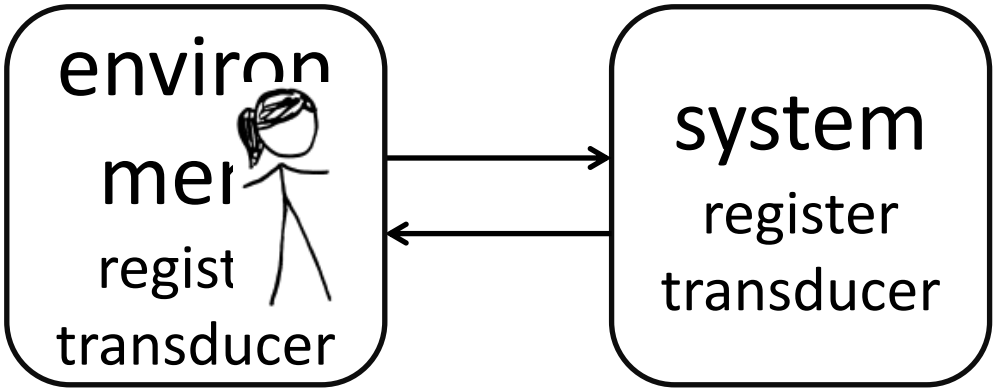
transition of A and some T

- For every T or T': $T' \vDash A'$ iff $T \vDash A$
- Recall that synthesis is EXP in $|Q'|$
- $Q' = Q \times \Pi$, where Π is the set of partitions of $R = R_s \cup R_A$
- $|\Pi|$ is EXP in $(k_s + k_A)$

=> synthesis is 2EXP in k_s and k_A

But system partitions behave deterministically
=> only **EXP in k_s**

Part 2



Environments have their own limits.

Let them be register transducers.

$$env||sys = (in_0, out_0)(in_1, out_1) \ldots$$

Note: the number of values is at most $k_s + k_e$.

env-sys-bounded synthesis problem

Given:

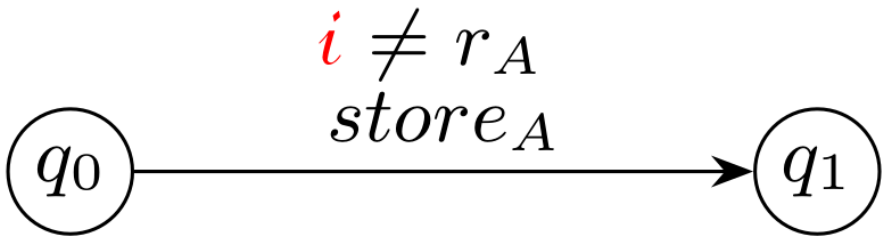
- Σ_I, Σ_O
- universal register automaton A with Σ_I and Σ_O
- the number k_s of system registers
- **the number k_e of environment registers**

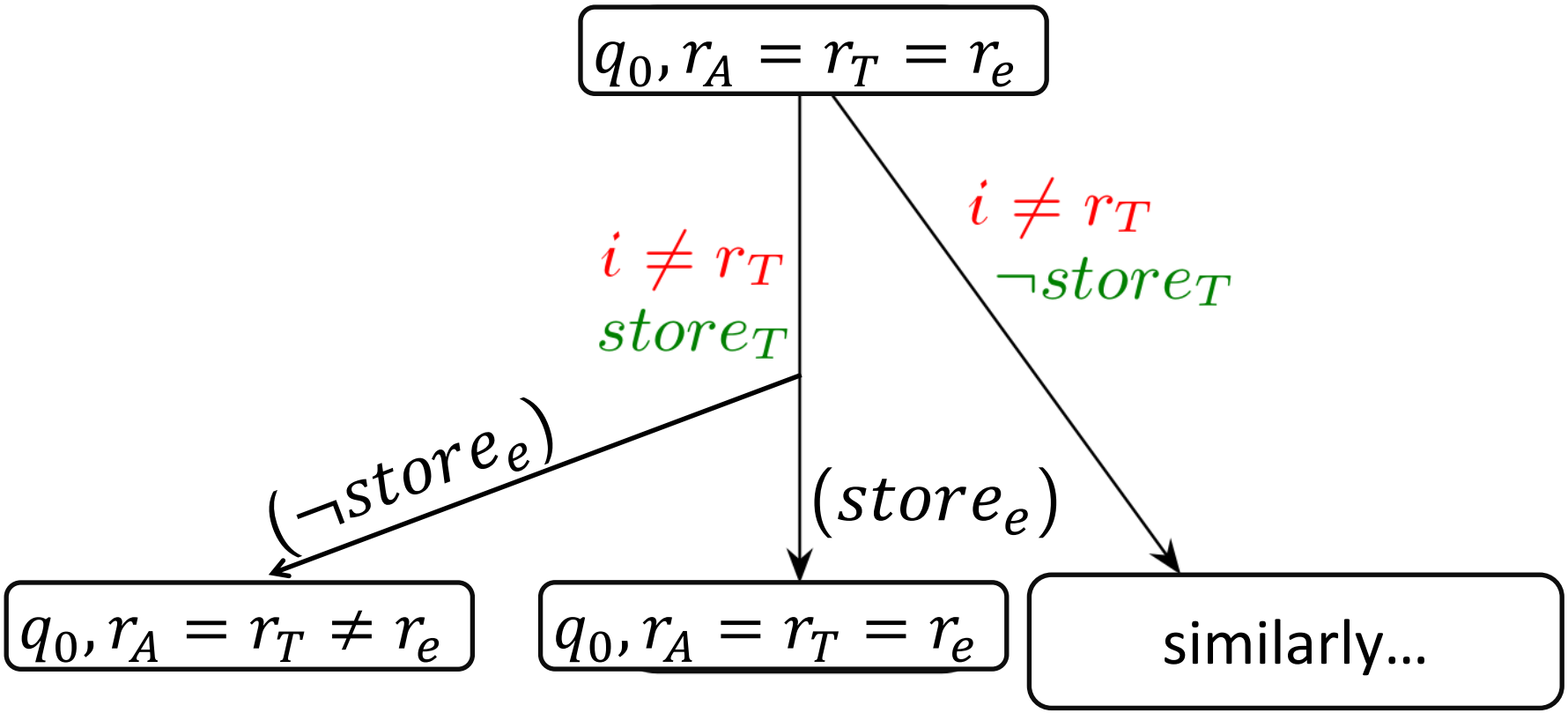
Return:

- k_s-register transducer sys such that $env||sys \vDash A$ for every k_e-register environment, or "unrealizable"

Env-sys-bounded synthesis problem is solvable in EXP in $|Q|$ and k_s, 2EXP in k_A and $\boldsymbol{k_e}$.

idea of the abstraction





conclusion

- Cleaner algorithm

  => tighter complexity analysis (only EXP in k_s)
- Solution to the environment-system-bounded synthesis problem

In the full version:

- Non-determinacy
- Hierarchy of system and environment power