Retiming and Resynthesis with Sweep Are Complete for Sequential Transformations

Hai Zhou
EECS
Northwestern University

Nov. 18, 2009

The Transformations

Retiming

Relocate registers from fanins of a subcircuit to fanouts, or vice versa.

The Transformations

Retiming

Relocate registers from fanins of a subcircuit to fanouts, or vice versa.

Resynthesis (aka Combinational Synthesis)

Restructure combinational circuit without changing its function.

The Transformations

Retiming

Relocate registers from fanins of a subcircuit to fanouts, or vice versa.

Resynthesis (aka Combinational Synthesis)

Restructure combinational circuit without changing its function.

Sweep (aka Register Sweep)

Remove registers not observable by output.

The Transformations

Retiming

Relocate registers from fanins of a subcircuit to fanouts, or vice versa.

Resynthesis (aka Combinational Synthesis)

Restructure combinational circuit without changing its function.

Sweep (aka Register Sweep)

Remove or insert registers not observable by output.

Power of Retiming and Resynthesis (RnR)

- Iterative retiming and resynthesis [Malik et al. 90] provide a powerful structural transformation

Power of Retiming and Resynthesis (RnR)

- Iterative retiming and resynthesis [Malik et al. 90] provide a powerful structural transformation
- Retiming gives combinational synthesis larger subcircuit to restructure

Power of Retiming and Resynthesis (RnR)

- Iterative retiming and resynthesis [Malik et al. 90] provide a powerful structural transformation
- Retiming gives combinational synthesis larger subcircuit to restructure
- Resynthesis gives retiming more signals to put registers on

Power of Retiming and Resynthesis (RnR)

- Iterative retiming and resynthesis [Malik et al. 90] provide a powerful structural transformation
- Retiming gives combinational synthesis larger subcircuit to restructure
- Resynthesis gives retiming more signals to put registers on

How Powerful are Retiming and Resynthesis?

Are they complete for all sequential transformations?

A Little Bit History

Leiserson & Saxe 83

A circuit transformed by retiming is steady state equivalent to original circuit.

A Little Bit History

Leiserson & Saxe 83

A circuit transformed by retiming and resynthesis is steady state equivalent to original circuit.

A Little Bit History

Leiserson & Saxe 83

A circuit transformed by retiming and resynthesis is steady state equivalent to original circuit.

Malik et al. 90

Asking whether reverse is true, proved that any state re-encoding can be done by RnR.

A Little Bit History

Leiserson & Saxe 83

A circuit transformed by retiming and resynthesis is steady state equivalent to original circuit.

Malik et al. 90

Asking whether reverse is true, proved that any state re-encoding can be done by RnR.

Malik 90

Proved (wrongly) that any cycle-preserving (CP) transformation can be done by RnR.

A Little Bit History

Zhou, Singhal, Aziz 98

Showed that there are equivalent (and CP) circuits that cannot be transformed by RnR.

A Little Bit History

Zhou, Singhal, Aziz 98

Showed that there are equivalent (and CP) circuits that cannot be transformed by RnR.
Somenzi suggested sweep to get it done.

A Little Bit History

Zhou, Singhal, Aziz 98

Showed that there are equivalent (and CP) circuits that cannot be transformed by RnR.
Somenzi suggested sweep to get it done.

Ranjan et al. 98

Corrected Malik's result to transformations only by 1-step merging, splitting, or switching.

A Little Bit History

Zhou, Singhal, Aziz 98

Showed that there are equivalent (and CP) circuits that cannot be transformed by RnR.
Somenzi suggested sweep to get it done.

Ranjan et al. 98

Corrected Malik's result to transformations only by 1-step merging, splitting, or switching.

Jiang & Brayton 06

RnR are exactly transformations by a sequence of 1-step merging and splitting.

Main Result

Theorem

Retiming and Resynthesis with Sweep are complete for steady state equivalent sequential transformations

Main Result

Theorem

Retiming and Resynthesis with Sweep are complete for steady state equivalent sequential transformations if one-cycle reachability is allowed in synthesis.

Verification Side of Story

Zhou, Singhal, Aziz 98

Proved that steady state equivalence checking is PSPACE-complete; but conjectured RnR checking is easier.

Verification Side of Story

Zhou, Singhal, Aziz 98

Proved that steady state equivalence checking is PSPACE-complete; but conjectured RnR checking is easier.

Jiang & Brayton 06

Proved that RnR checking is also PSPACE-complete, disproving the conjecture.

Verification Side of Story

Zhou, Singhal, Aziz 98

Proved that steady state equivalence checking is PSPACE-complete; but conjectured RnR checking is easier.

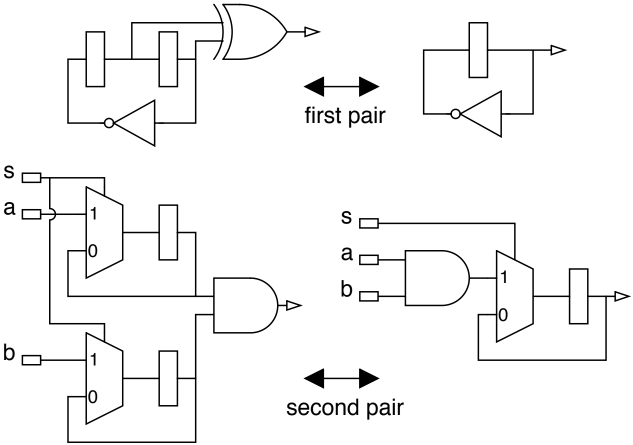
Jiang & Brayton 06

Proved that RnR checking is also PSPACE-complete, disproving the conjecture.

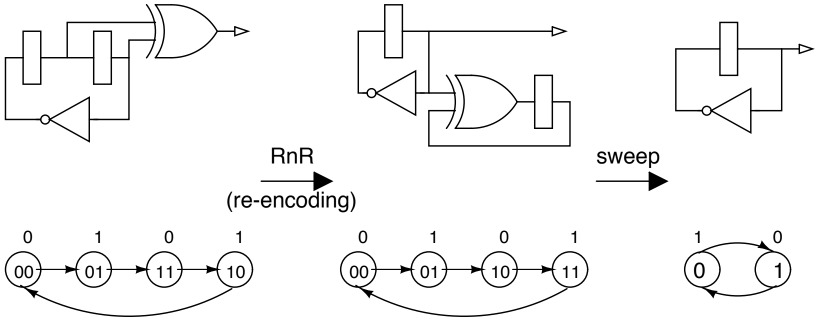
We point out in paper

Re-encoding checking is PSPACE-hard, but the complexity of RnR checking is still open.

Circuits Demonstrating Incompleteness of RnR

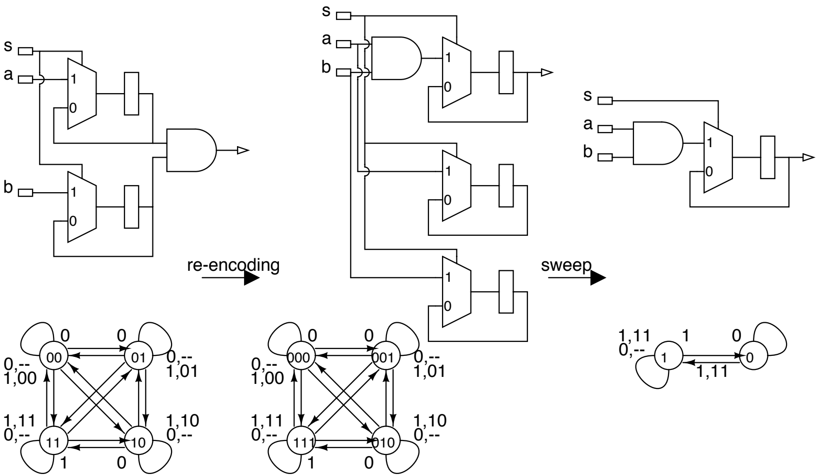


Sweep is Necessary

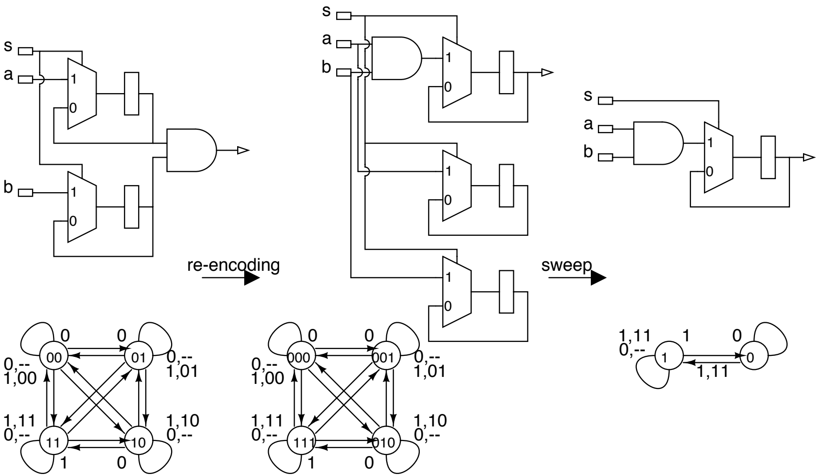


Is Sweep Sufficient?

Is Sweep Sufficient?



Is Sweep Sufficient?



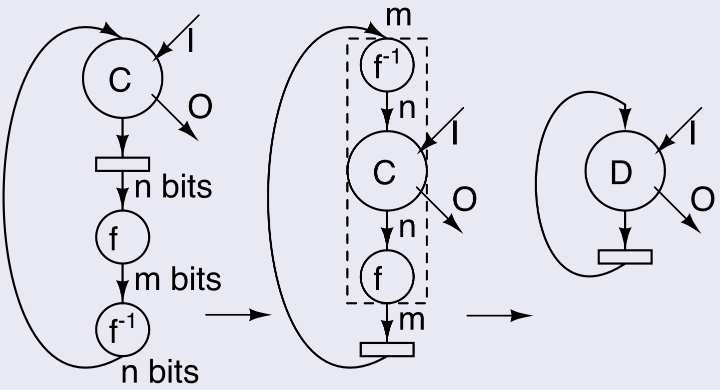
Warning

Re-encoding with different length is needed!

Is RnR Complete for Re-encoding with Different Length?

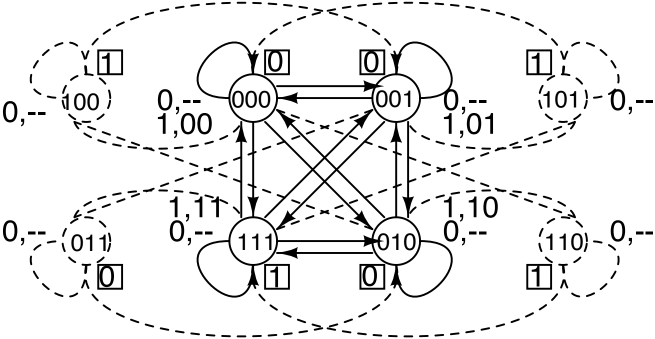
Is RnR Complete for Re-encoding with Different Length?

Proof Sketch



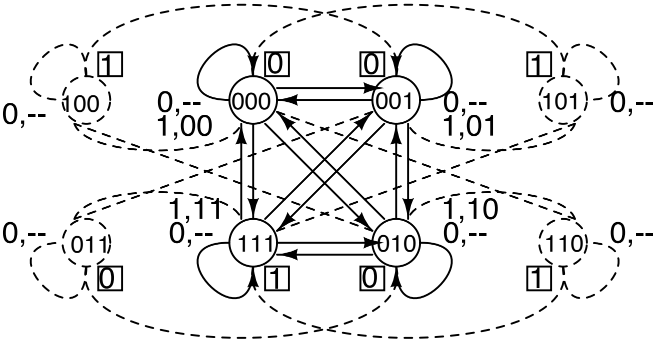
Re-encoding with Different Code Length

- Extra shadow states are introduced:



Re-encoding with Different Code Length

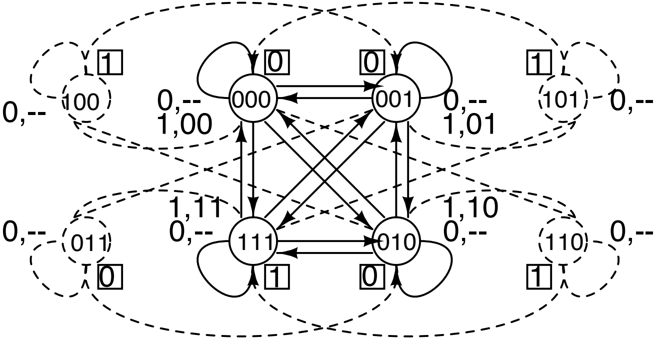
- Extra shadow states are introduced:



- They cannot be generated by 1-step mergings or splittings!

Re-encoding with Different Code Length

- Extra shadow states are introduced:

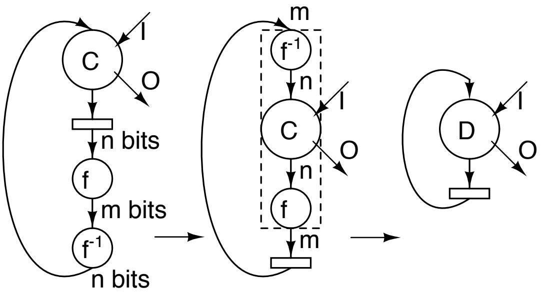


- They cannot be generated by 1-step mergings or splittings!

Contradicting w/ Jiang & Brayton 06

What is wrong?

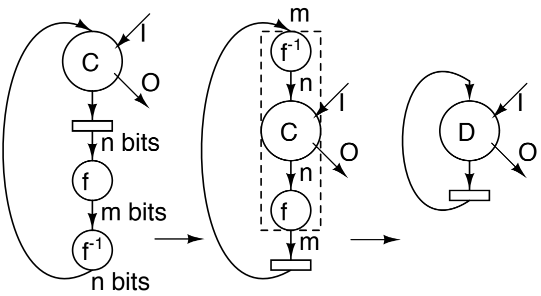
Encoding Representation Is Important



Observation

Treating Boolean functions as abstract discrete functions turns to boast the power of synthesis!
A discrete function may have a range of $2^n + 1$ symbols, but a corresponding Boolean one will have 2^{n+1} values.

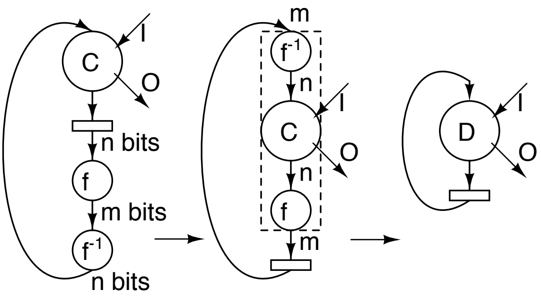
Solution



One-Cycle Reachability (OCR)

We need to look into previous cycle to find the domain of f^{-1} which was the range of f!

Solution



One-Cycle Reachability (OCR)

We need to look into previous cycle to find the domain of f^{-1} which was the range of f!

Lemma

Without OCR, RnR is not complete for transforming between two given circuits that are re-encodings with different code lengths.

Inspiration by Abadi & Lamport 91

The existence of refinement mappings, TCS, 82(2), 1991

Under three general hypotheses about the specifications, if S_1 implements S_2 then one can add auxiliary history and prophecy variables to S_1 to form equivalent specification S_1^{hp} and find a refinement mapping from S_1^{hp} to S_2.

Completeness for Sequential Transformation

Theorem

Retiming and Resynthesis with Sweep are complete for steady state equivalent sequential transformations, if OCR is allowed.

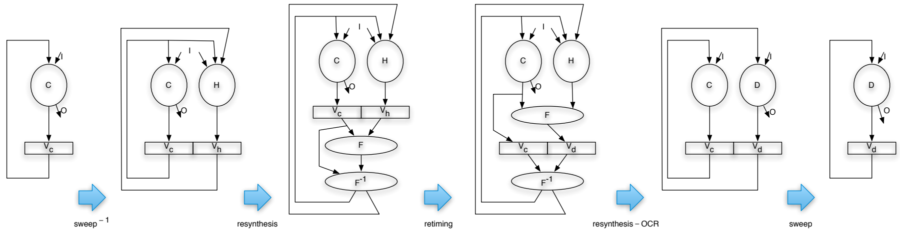
Proof.

1. Circuits C and D are steady state equivalent $\Rightarrow$ every steady state of C maps to at least one D state.
2. Use sweep (inverse) to add registers in C to make an "onto" refinement function F from C states to D states (Abadi & Lamport 91)
3. Bypass signals to make F into a bijection
4. Resynthesis $F^{-1} \circ F$ at the register output of C
5. Retime registers to outputs of F
6. Resynthesis with OCR
7. Sweep to remove unobservable registers to get D $\square$

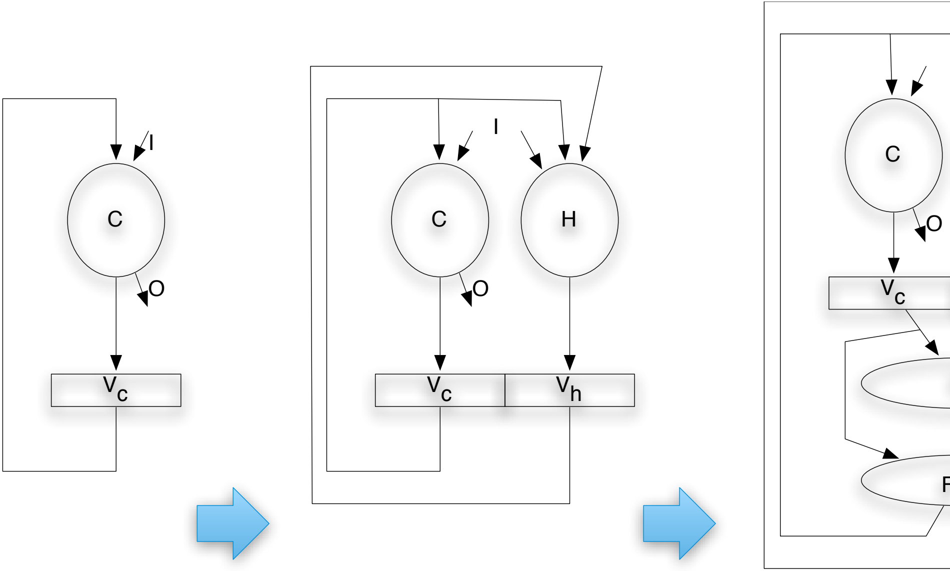
Completeness for Sequential Transformation

Theorem

Retiming and Resynthesis with Sweep are complete for steady state equivalent sequential transformations, if ORC is allowed.



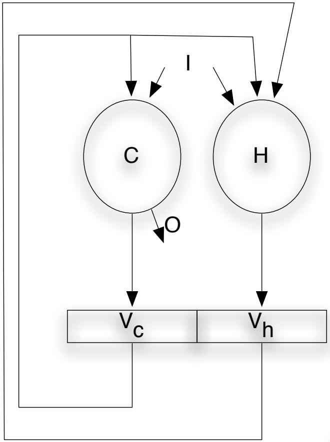
Completeness for Sequential Transformation

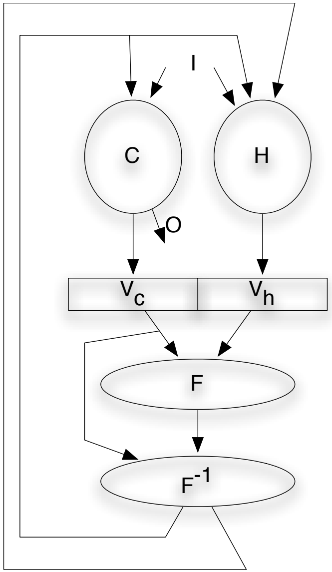


sweep $^{-1}$

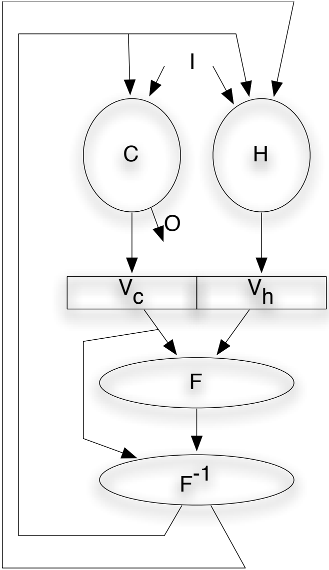
resynthesis

Completeness for Sequential Transformation



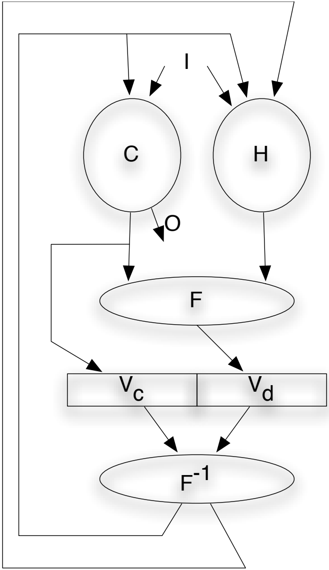


Completeness for Sequential Transformation



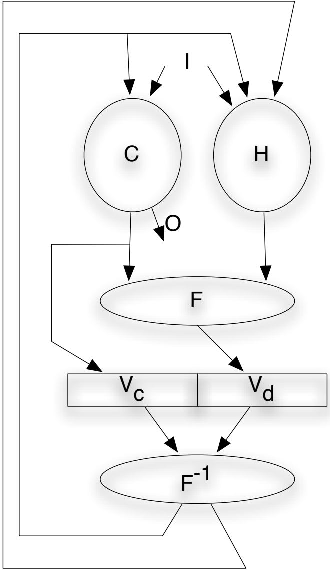
esis

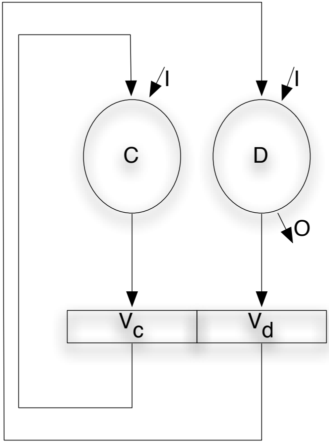
retiming



resynthe

Completeness for Sequential Transformation

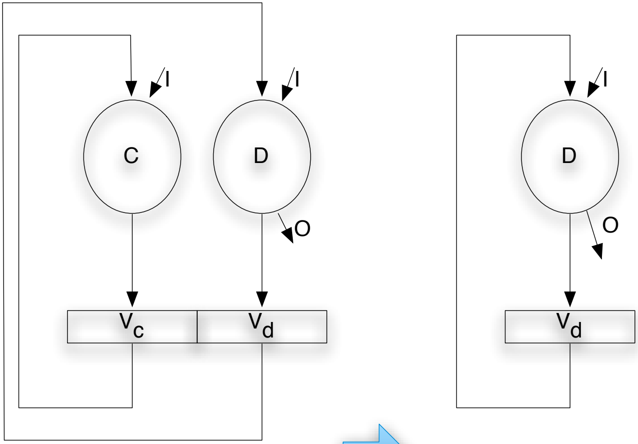




resynthesis – OCR

sweep

Completeness for Sequential Transformation



ynthesis – OCR

sweep

Implications and Future Work

- RnR-Sweep provide powerful sequential transformations, thus need to be developed as a main sequential optimization tool.

Implications and Future Work

- RnR-Sweep provide powerful sequential transformations, thus need to be developed as a main sequential optimization tool.
- OCR needs to be used commonly.

Implications and Future Work

- RnR-Sweep provide powerful sequential transformations, thus need to be developed as a main sequential optimization tool.
- OCR needs to be used commonly.
- Efficiently verifiable subset of RnR-Sweep transformations?

Implications and Future Work

- RnR-Sweep provide powerful sequential transformations, thus need to be developed as a main sequential optimization tool.
- OCR needs to be used commonly.
- Efficiently verifiable subset of RnR-Sweep transformations?
- How powerful are RnR-Sweep without OCR?

Implications and Future Work

- RnR-Sweep provide powerful sequential transformations, thus need to be developed as a main sequential optimization tool.
- OCR needs to be used commonly.
- Efficiently verifiable subset of RnR-Sweep transformations?
- How powerful are RnR-Sweep without OCR?
- What is complexity of RnR equivalence checking?

Q & A