

# Wire Retiming for System-On-Chip by Fixpoint Computation

Chuan Lin and Hai Zhou

Electrical and Computer Engineering  
Northwestern University

# Outline

- Motivation
- Problem formulation
- Fixed period retiming as a fixpoint computation
- Algorithm
- Experimental results
- Conclusions

# Motivation

- Industry trend
  - Frequency: 2X/generation, Die size: 1.25X/generation
  - Problem: global signal communication

# Motivation

- Industry trend
  - Frequency: 2X/generation, Die size: 1.25X/generation
  - Problem: global signal communication
- Recent research
  - Insert flip-flops on global wires (Intel, IBM, etc.)
  - How to keep functional correctness

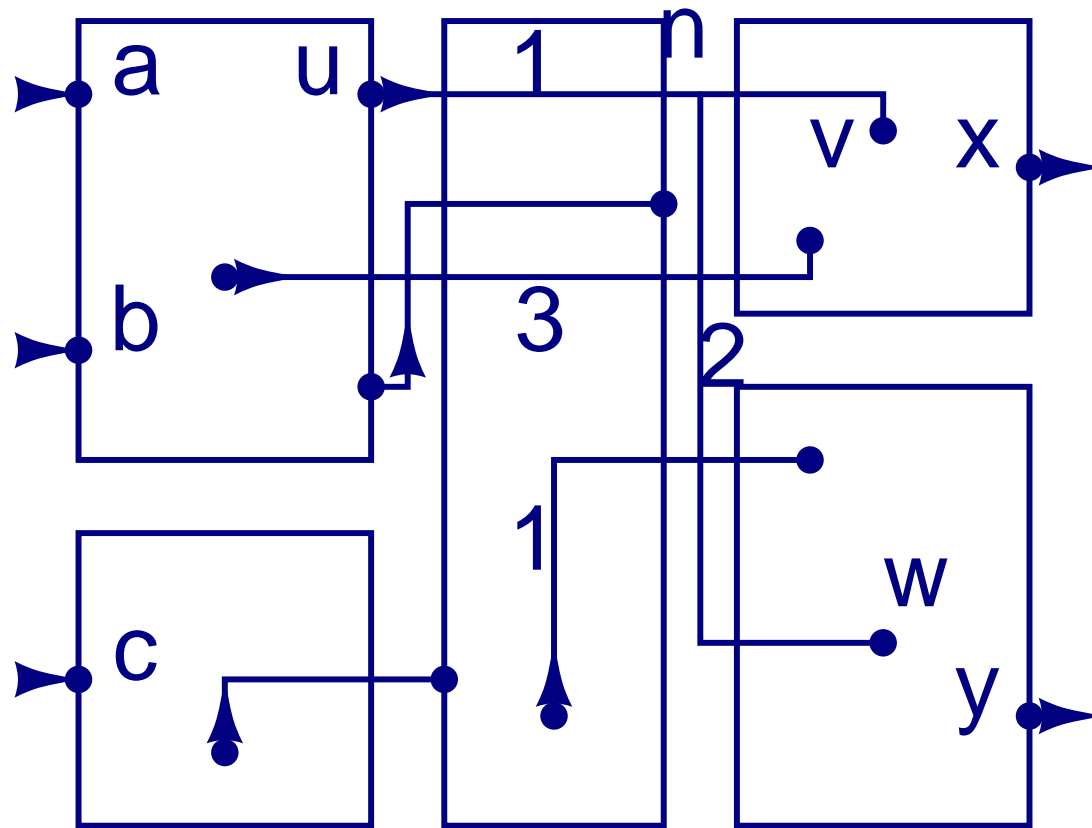
## Motivation

- Retiming can move flip-flops w/o changing functionality
- Traditional retiming mainly consider gate delay
- We want flip-flops to pipeline long wire
  - Multiple flip-flops on a wire
  - Positions of flip-flops on a wire

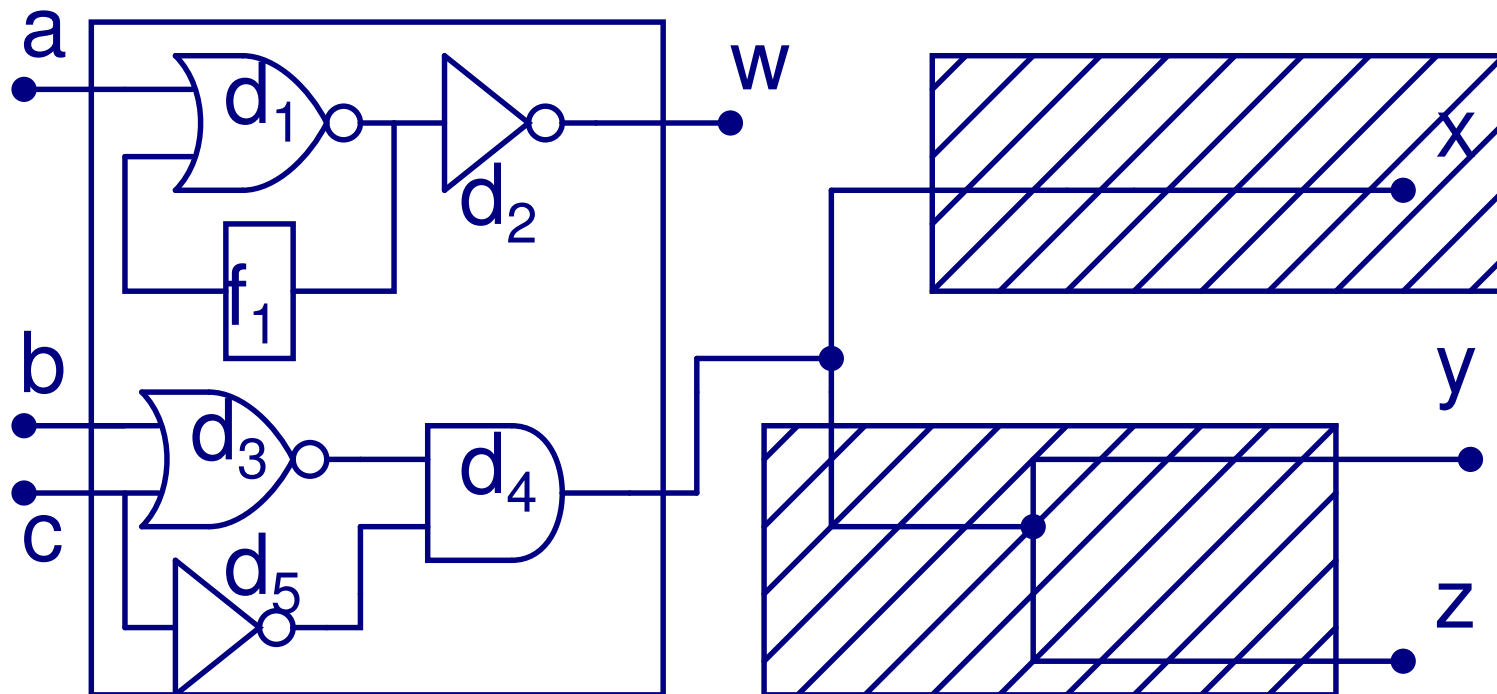
# Motivation

- Previous work
  - Timing macro models
  - Polynomial time algorithm for feasibility checking
  - Long running time: limited application

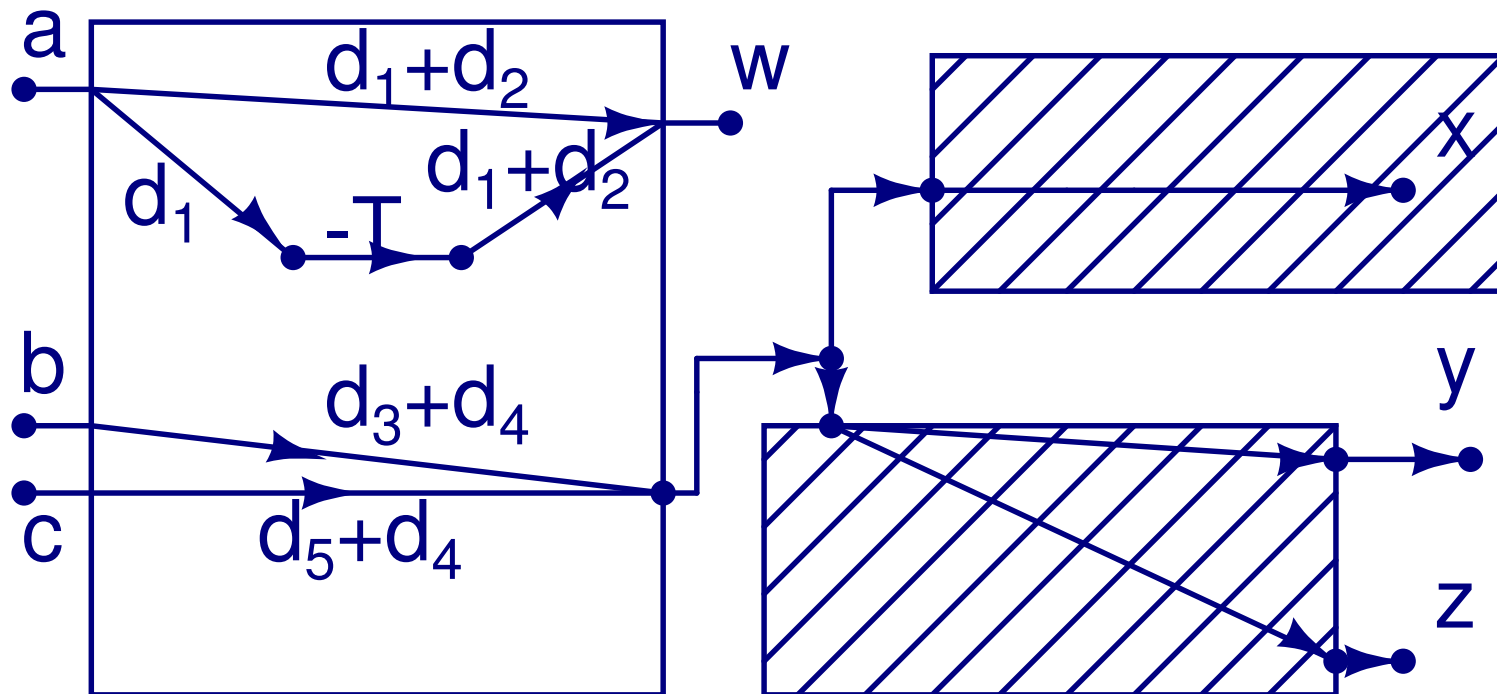
## An SOC design example



## A specific circuit configuration



## The timing model of the circuit



## Problem formulation

### [Minimum Period Wire Retiming]

- $G = (V, E)$ ,  $E = E_1 \cup E_2$ ,  $E_1 \cap E_2 = \emptyset$   
delay:  $d(e)$ , #flip-flop:  $w(e)$ ,  $\forall e \in E$

## Problem formulation

### [Minimum Period Wire Retiming]

- $G = (V, E)$ ,  $E = E_1 \cup E_2$ ,  $E_1 \cap E_2 = \emptyset$   
delay:  $d(e)$ , #flip-flop:  $w(e)$ ,  $\forall e \in E$
- $\forall e \in E_2$ ,  $d(e)$  is proportional to its length
  - Buffers can be inserted to make it linear

# Problem formulation

## [Minimum Period Wire Retiming]

- $G = (V, E)$ ,  $E = E_1 \cup E_2$ ,  $E_1 \cap E_2 = \emptyset$   
delay:  $d(e)$ , #flip-flop:  $w(e)$ ,  $\forall e \in E$
- $\forall e \in E_2$ ,  $d(e)$  is proportional to its length
  - Buffers can be inserted to make it linear
- Find a relocation of flip-flops
  - No flip-flop change on any  $e \in E_1$
  - Minimize the maximum delay between any two consecutive flip-flops(clock period)

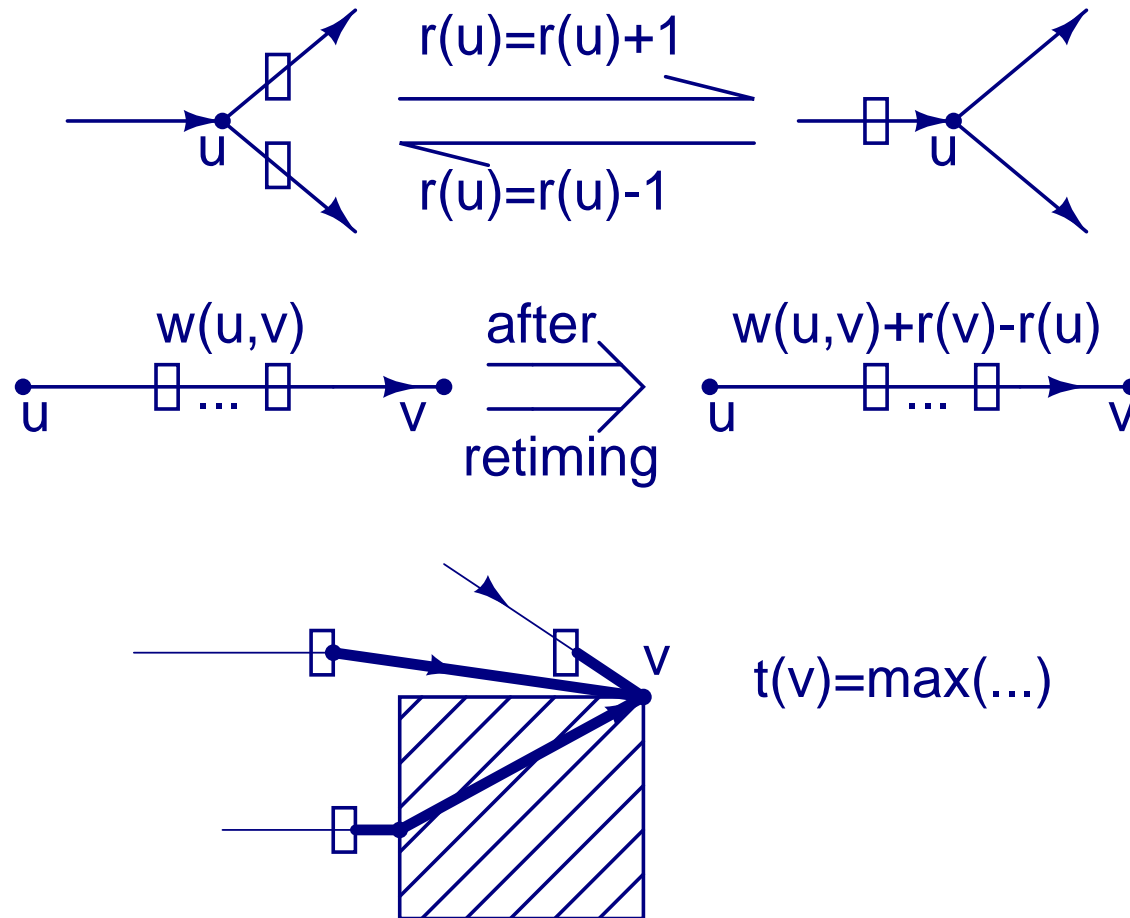
## Strategy of solving the problem

- Fixed Period Wire Retiming
  - Determine if exists a retiming under a fixed clock period( $T$ )

## Strategy of solving the problem

- Fixed Period Wire Retiming
  - Determine if exists a retiming under a fixed clock period( $T$ )
- Binary Search
  - Over the range from lower to upper bound

## Two essential variables $r$ and $t$



## Requirements for $r$ and $t$

$$r(x) = r(y), \quad \forall (x, y) \in E_1$$

$$w(u, v) + r(v) - r(u) \geq 0, \quad \forall (u, v) \in E_2$$

$$t(y) \geq t(x) + d(x, y), \quad \forall (x, y) \in E_1$$

$$t(v) \geq t(u) + d(u, v) - [w(u, v) + r(v) - r(u)]T, \forall (u, v) \in E_2$$

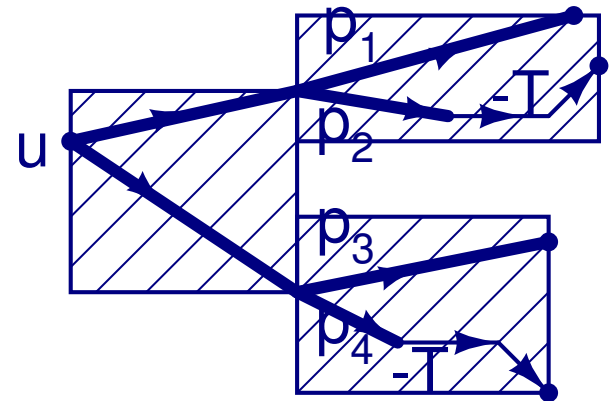
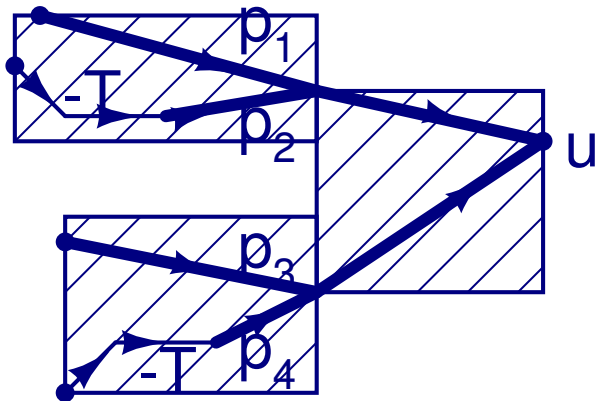
$$0 \leq t(u) \leq \mathbf{T}, \quad \forall u \in V$$

## Requirements for $r$ and $t$

$$\mathbf{fd}(u) \leq t(u) \leq T - \mathbf{bd}(u), \quad \forall u \in V, \quad (1)$$

$$r(u) = r(v), \quad \forall (u, v) \in E_1, \quad (2)$$

$$t(v) \geq t(u) + d(u, v) - (w(u, v) + r(v) - r(u))T, \forall (u, v) \in E. \quad (3)$$



$$\mathbf{fd}(u) = \max(d(p_1), d(p_2), d(p_3), d(p_4)) \quad \mathbf{bd}(u) = \max(d(p_1), d(p_2), d(p_3), d(p_4))$$

## Solving the Fixed Period Wire Retiming problem

**Theorem 1** (1)-(3) have a solution if and only if the following equation has a solution for any  $v \in V$ ,

$$(r(v), t(v)) = \begin{cases} (r_1(v), t_1(v)) & \text{if } t_1(v) \leq T - bd(v) \\ (r_1(v) + 1, fd(v)) & \text{otherwise} \end{cases} \quad (4)$$

## Solving the Fixed Period Wire Retiming problem

**Theorem 1** (1)-(3) have a solution if and only if the following equation has a solution for any  $v \in V$ ,

$$(r(v), t(v)) = \begin{cases} (r_1(v), t_1(v)) & \text{if } t_1(v) \leq T - bd(v) \\ (r_1(v) + 1, fd(v)) & \text{otherwise} \end{cases} \quad (4)$$

where

$$r_1(v) = \max \left( \begin{aligned} & \max_{\forall (u,v), (v,u) \in E_1, r(v) < r(u)} r(u), \\ & \max_{\forall (u,v) \in E_2} \left[ \frac{t(u) + d(u,v)}{T} - w(u,v) + r(u) \right] - 1 \end{aligned} \right),$$

$$t_1(v) = \max \left( fd(v), \right.$$

$$\left. \max_{\forall (u,v) \in E} t(u) + d(u,v) - (w(u,v) + r_1(v) - r(u))T \right).$$

# Fixpoint formulation

- Notations

- Retiming variable:  $x_v = (r(v), t(v)), \forall v \in V$

- Equation (4):  $x_v = f_v(x_{v_1}, \dots, x_{v_k}),$   
where  $(v_i, v)$  or  $(v, v_i) \in E, 1 \leq i \leq k$

# Fixpoint formulation

- Notations

- Retiming variable:  $x_v = (r(v), t(v)), \forall v \in V$
- Equation (4):  $x_v = f_v(x_{v_1}, \dots, x_{v_k}),$   
where  $(v_i, v)$  or  $(v, v_i) \in E, 1 \leq i \leq k$
- Retiming vector:  $X = (x_1, x_2, \dots, x_n),$  where  $n = |V|$
- System transformation:  $\mathbf{X} = \mathbf{F}(\mathbf{X}),$  where  $F = f_1, f_2, \dots, f_n$ 
  - \* Its solution is a solution to Fixed Period Wire Retiming problem

## Fixpoint formulation

- Partial order ( $\leq$ )
  - $x_u \leq x'_u : (r(u) < r'(u)) \vee (r(u) = r'(u) \wedge t(u) \leq t'(u))$
  - $X \leq Y : x_i \leq y_i, 1 \leq i \leq n$

## Fixpoint formulation

- Partial order ( $\leq$ )
  - $x_u \leq x'_u : (r(u) < r'(u)) \vee (r(u) = r'(u) \wedge t(u) \leq t'(u))$
  - $X \leq Y : x_i \leq y_i, 1 \leq i \leq n$
  - Bottom element( $\perp$ ):  $t(u) = r(u) = 0, \forall u \in \{PI\}$  and  $t(v) = r(v) = -\infty, \forall v \in V - \{PI\}$
  - Top element( $\top$ ):  $t(v) = r(v) = \infty, \forall v \in V$

## Fixpoint formulation

- Partial order ( $\leq$ )
  - $x_u \leq x'_u : (r(u) < r'(u)) \vee (r(u) = r'(u) \wedge t(u) \leq t'(u))$
  - $X \leq Y : x_i \leq y_i, 1 \leq i \leq n$
  - Bottom element( $\perp$ ):  $t(u) = r(u) = 0, \forall u \in \{PI\}$  and  $t(v) = r(v) = -\infty, \forall v \in V - \{PI\}$
  - Top element( $\top$ ):  $t(v) = r(v) = \infty, \forall v \in V$
  - Solution space  $P$  is a **Complete Partially Ordered set**,  $\forall X \in P, \perp \leq X \leq \top$

## Fixpoint formulation

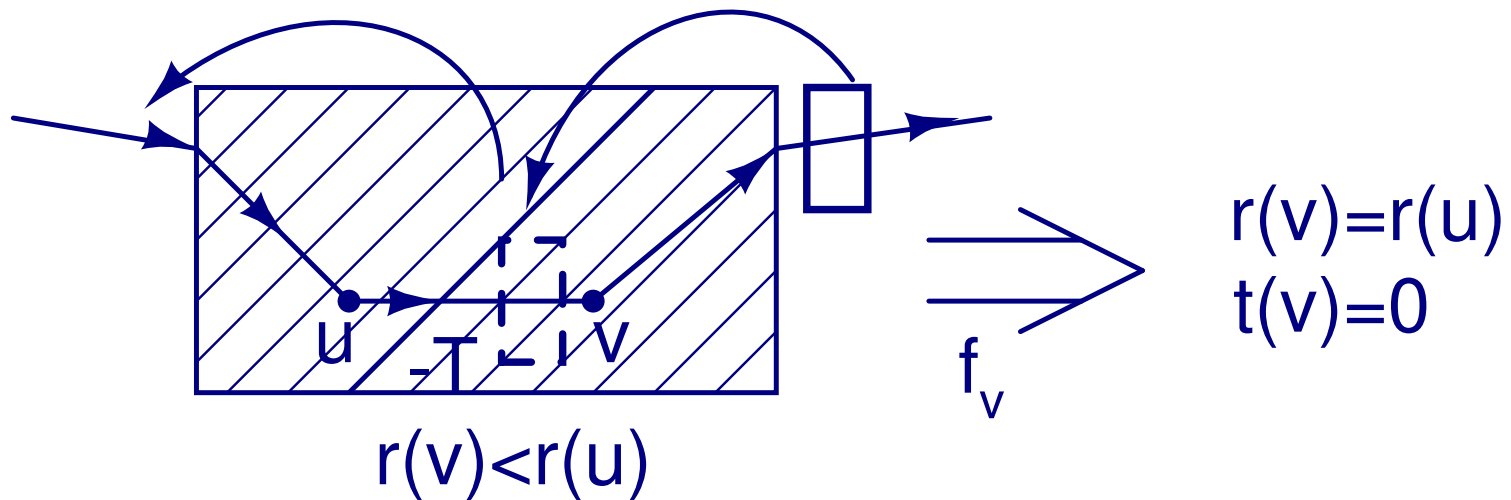
- $F$  is monotonic:  $X \leq Y \Rightarrow F(X) \leq F(Y)$
- Iterative method
  - Initial vector:  $X_0 = \perp$
  - Iterative computation:  $X_1 = F(X_0), X_2 = F(X_1), \dots$ , until  $X_m = X_{m-1}$ , a **fixpoint** of  $F$
  - $F$  is **finitely convergent** if  $m$  is finite
- $F$  is finitely convergent if and only if  $T$  is feasible

## Chaotic iteration scheme

- $X_{i+1} = F(X_i)$  requires updating every vertex at the same time
- Partial transformation  $F_S$  updates only a subset of vertices  $S \subset V$ .
- Same fixpoint will be reached no matter what update order is used

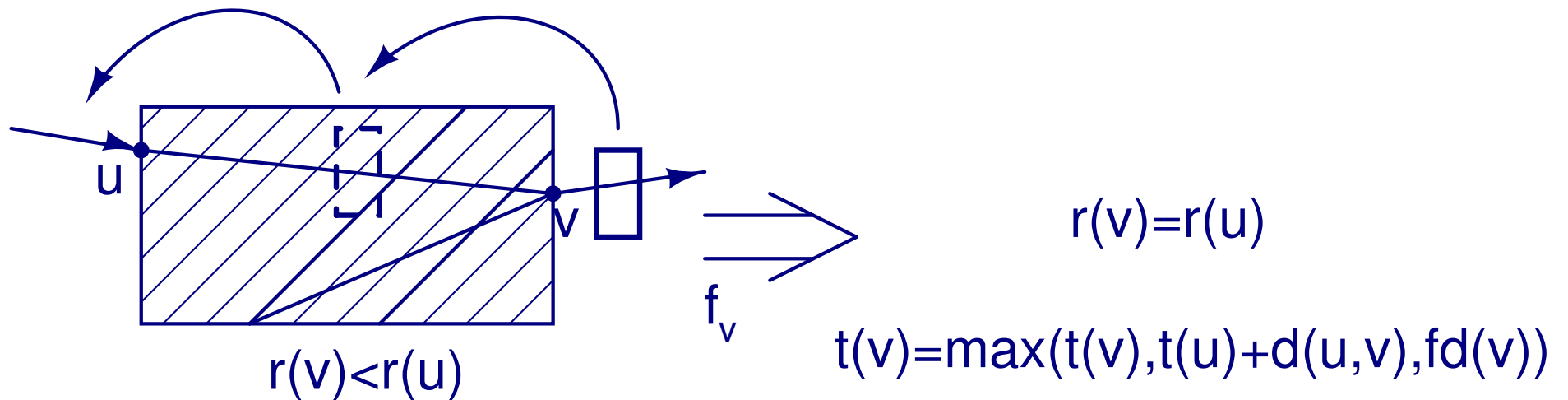
## Iteration in algorithm

- Case 1:  $(u, v) \in E_1$ ,  $d(u, v) = -T$



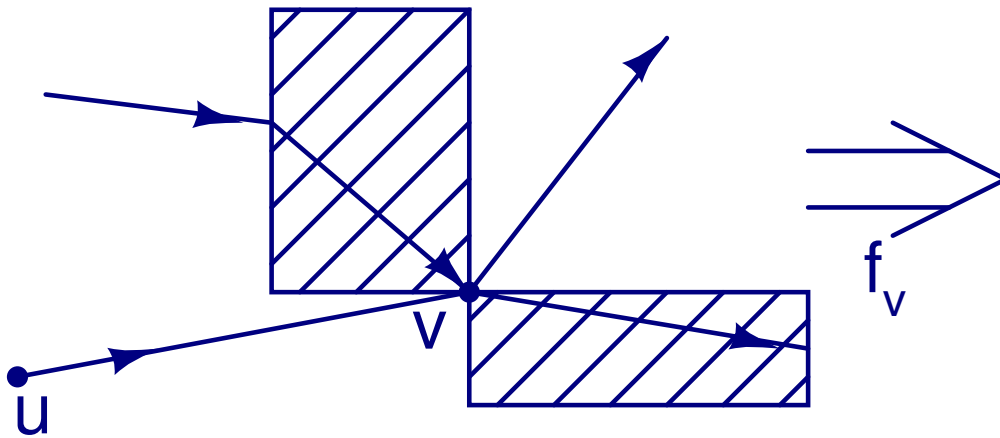
## Iteration in algorithm

- Case 2:  $(u, v) \in E_1, d(u, v) > 0$



## Iteration in algorithm

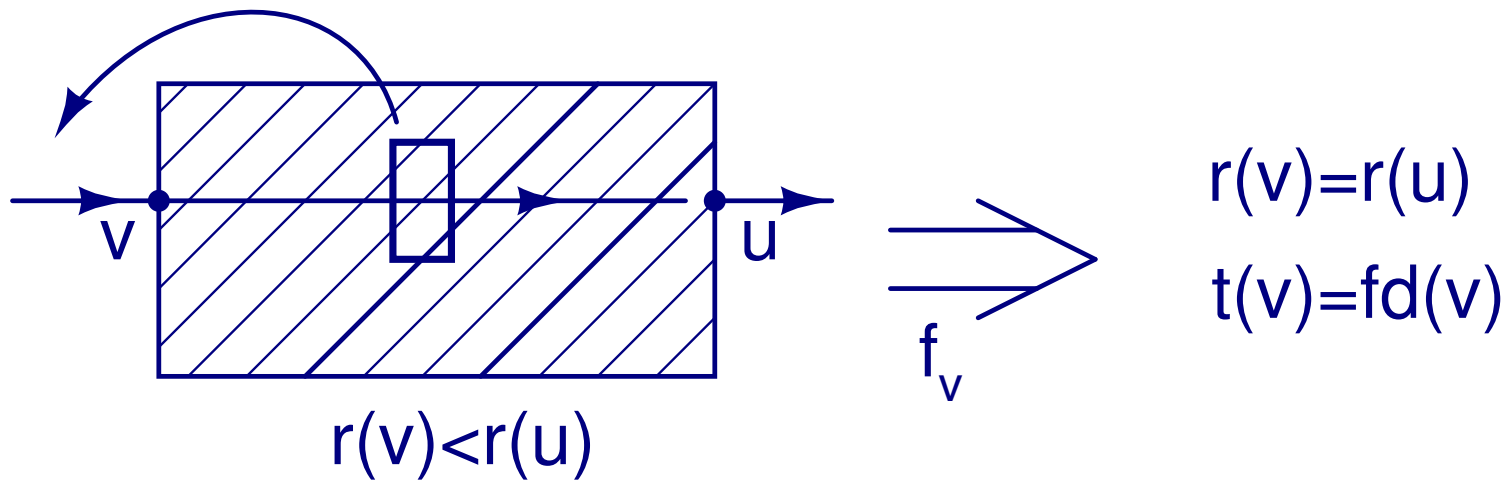
- Case 3:  $(u, v) \in E_2$



Select minimal  $x_v$  that satisfies equations (1)-(3) on edge  $(u, v)$

## Iteration in algorithm

- Case 4:  $(v, u) \in E_1$



## Iteration stop condition

- Solution does not change  $\Rightarrow$  fixpoint found
- T is infeasible if :
  - $\exists v \in V, r(v) > \max FF$ , the total # of FF in the circuit
  - $\exists v \in PO, r(v) > 0$
  - A technique similar to Shenoy and Rudell is also used

## Fixed period algorithm

**Algorithm** Fixed period wire retiming

Initialization, add  $PI$  into  $Q$ ;

While ( $Q \neq \emptyset$ )

do  $u \leftarrow \text{extract}(Q)$ ;

For each  $(u, v)$  or  $(v, u) \in E$

do If  $x_v \neq f_v(x_u)$  then

$x_v \leftarrow f_v(x_u)$ ;

$Q \leftarrow v$ ;

If infeasibility is discovered then

Exit;

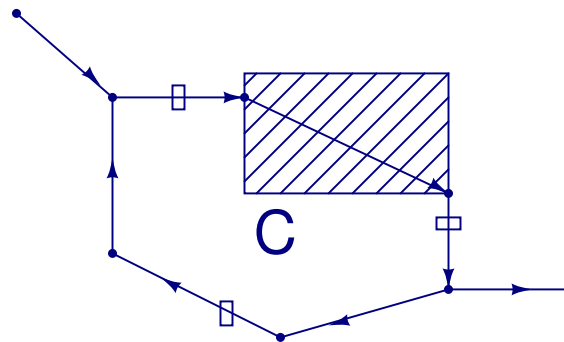
Report the fixpoint;

## Range of clock periods

- Lower bounds for  $T$

$$T \geq T_1 \stackrel{\text{def}}{=} \max_{(x,y) \in E_1} d(x, y)$$

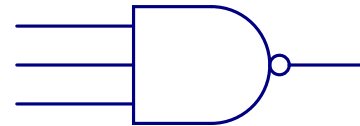
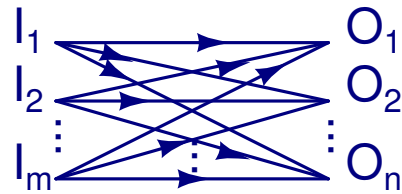
$$T \geq T_2 \stackrel{\text{def}}{=} \max_{c \in \text{cycle}} \frac{d(c)}{w(c)} \quad (5)$$



## Range of clock periods

- Upper bound for  $T$

**Lemma 1** *If each connected component in the subgraph  $G_1 = (V, E_1)$  is a **complete bipartite graph**, the optimal clock period can be upper bounded by  $T_1 + T_2$ .*



## Min period algorithm

**Algorithm** Minimum Period Wire Retiming

Find the upper bound  $T_u$  and lower bound  $T_l$ ;

**do**

$$T = \frac{T_u + T_l}{2};$$

Run Fixed Period Wire Retiming under  $T$ ;

If a fixpoint is reported then

$$T_u = T;$$

Else

$$T_l = T;$$

**while**  $T_u - T_l > \epsilon$ ;

## Experimental results

- Benchmark: ISCAS-89
  - 1st test set: treat gates as blocks
  - 2nd test set: circuits w/ non-complete bipartite blocks
    - \* Use hMETIS to partition a circuit into groups
    - \* Treat each group as a block
- Binary search precision:  $\epsilon = 0.1$

## Experimental results

- Optimal clock period

Circuit	$ V $	$ E $	$T_2$	w/o non-CB $T_{opt}$	w/ non-CB $T_{opt}$
s386	519	700	51.0	51.1	55.0
s400	511	665	32.2	32.2	50.6
s444	557	725	35.0	35.2	63.2
s838	1299	1206	76.0	76.0	84.0
s953	1183	1515	60.6	60.6	69.5
<b>s1488</b>	2054	2780	<b>70.1</b>	<b>70.6</b>	<b>73.3</b>
s1494	2054	2792	76.8	76.9	79.9
s5378	7205	8603	111.0	111.2	115.3
s13207	19816	22999	239.5	239.5	292.8
s35932	46097	58266	148.3	148.3	163.2
<b>s38584</b>	53473	66964	<b>203.9</b>	<b>204.0</b>	<b>264.0</b>

## Experimental results

- Running time improvement (**with non-CB blocks**)

Circuit	$t_{old}$ (sec)	$t_{new}$ (sec)
s386	3.67	0.01
s400	3.38	0.01
s444	4.31	0.01
s838	33.42	0.02
s953	17.56	0.07
s1488	98.88	0.05
s1494	62.86	0.09
s5378	1344.74	0.29
s13207	-	206.52
s35932	-	6.19
s38584	-	21992.67

## Summary

- Wire retiming is becoming more and more important on a SOC design

## Summary

- Wire retiming is becoming more and more important on a SOC design
- Besides the previous work, the problem can be more efficient solved

## Summary

- Wire retiming is becoming more and more important on a SOC design
- Besides the previous work, the problem can be more efficient solved
  - The equivalence between the retiming solution and fix-point computation is established

## Summary

- Wire retiming is becoming more and more important on a SOC design
- Besides the previous work, the problem can be more efficient solved
  - The equivalence between the retiming solution and fix-point computation is established
  - An algorithm using iterative method is proposed with great efficiency

**Thank you !**