Simultaneous Slack Budgeting and Retiming for Synchronous Circuits Optimization

Speaker: **Shenghua Liu**

Authors: Shenghua Liu, Yuchun Ma,

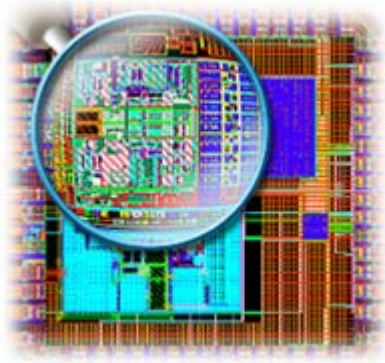
Xian-Long Hong, Yu Wang

Dept. of Computer Science & Technology, Tsinghua University

Outline

- Background
- Existing work
- Problem formulation
- Motivating examples
- Algorithm
  - Time Slack Budgeting for Estimating Potential Slack
  - Simultaneous Retiming and Slack Budgeting
- Experimental results
- Conclusion

Background



- More devices trend to be put in the small silicon area.
- The clock frequency is pushed even higher.
- More challenges in timing, area, and power dissipation.
- Timing budget aims at
  - slowing down as many components as possible without violating the system's timing constraints.
  - The slowed-down components can be further optimized to improve system's area, power dissipation, or other design quality metrics.
- Retiming
  - is one of the most powerful sequential optimization techniques
  - relocates the flip-flops (FFs) in a circuit while preserving its functionality

Outline

- Background
- **Existing work**
- Problem formulation
- Motivating examples
- Algorithm
  - Time Slack Budgeting for Estimating Potential Slack
  - Simultaneous Retiming and Slack Budgeting
- Experimental results
- Conclusion

Existing Work for slack budgeting (1)

- [R. Nair, TCAD, 1989]
  - used Zero-Slack Algorithm (ZSA) for slack budgeting
  - is suboptimal and heuristics
- [D.S. Chen, DAC, 1996] [C. Chen, TCAD, 2002] [C. Chen, Algorithmica, 2002]
  - formulated slack budgeting as maximum-independent-set (MIS) on sensitive transitive closure graph
- [K. Wang, ICCAD, 2004]
  - presented an LP-based slack budgeting and maximized the potential slack by clock skew optimization
- [E. Bozorgzadeh, DAC, 2003]
  - solved the problem of integral delay budgeting through LP relaxation.

Existing Work for slack budgeting (2)

- [S. Ghiasi, TCAD, 2006]
  - proposed combinatorial methods based on net flow approach to handle the slack budget problem.
- The existing slack budgeting algorithms
  - are either used for combinatorial circuit, or
  - limited to fixed FF locations
- At the early design stages, there is flexibility to schedule pipeline or timing distribution to obtain more timing slack

Existing Work for Retiming Optimization

- [H. Zhou, DAC, 2005] [H. Zhou, TODAES, 2008]
  - proposed a very efficient retiming algorithm for minimal period by algorithm derivation
- [C. Lin, DAC, 2006]
  - presented efficient incremental retiming for min-period retiming under setup and hold constraints
- [J. Wang, DAC, 2008]
  - incremental retiming for min-area retiming under a given clock period
- [N. Chabini, GLS-VLSI, 2003] [Y. Hu, DAC, 2006]
  - proposed the ILP/MILP-based algorithms of retiming for power reduction
- [C.Y. Yeh, IEEE Trans. on VLSI, 2004]
  - proposed an LP formulation for optimizing the slack budgeting on interconnects between gates

Contribution of Our Algorithm

- Maximizing the potential slack by retiming for synchronous sequential circuit
- Simultaneously slack budgeting and incremental retiming
- Extending the formulation of MIS (maximum-independent-set) to sequential circuit for potential slack estimation
- Optimizing the potential slack by incremental retiming

Outline

- Background
- Existing work
- **Problem formulation**
- Motivating examples
- Algorithm
  - Time Slack Budgeting for Estimating Potential Slack
  - Simultaneous Retiming and Slack Budgeting
- Experimental results
- Conclusion

Problem Formulation

- Given
  - a directed graph $G = (V, E, d, w)$ representing a circuit
  - each node $v \in V$ represents a gate
  - each edge $(u, v) \in E$ represents a signal passing from gate u to gate v
  - vertex weights $d : V \rightarrow R^*$, representing the gate delays
  - edge weights $w : E \rightarrow Z^*$, representing the number of flip-flops (FFs) on edges
  - the period constraint T
- Find
  - a relocation of flip-flops $w' : E \rightarrow N$
- Such that
  - the potential slack P_s obtained by slack budgeting is maximized
  - under the given clock period T

Labels Definition

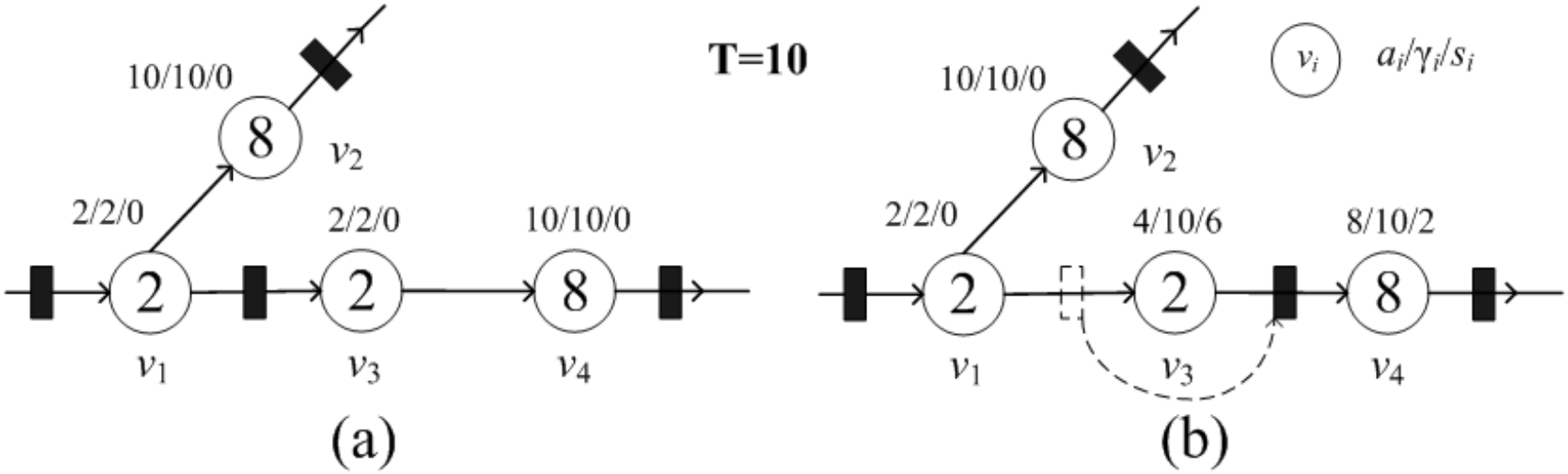
- Three non-negative labels, $a_i / \gamma_i / s_i : V \rightarrow R^*$
  - the latest arrival time, require time, and slack of gate i.

- An integer label $r : V \rightarrow Z^*$
  - the number of FFs are moved from the outgoing edges to the incoming edges of each node
  - when r is negative value, it means FFs are moved back from the incoming edges to outgoing edges
  - the number of FFs on edge (v_i, v_j) with current label r is as formula

Outline

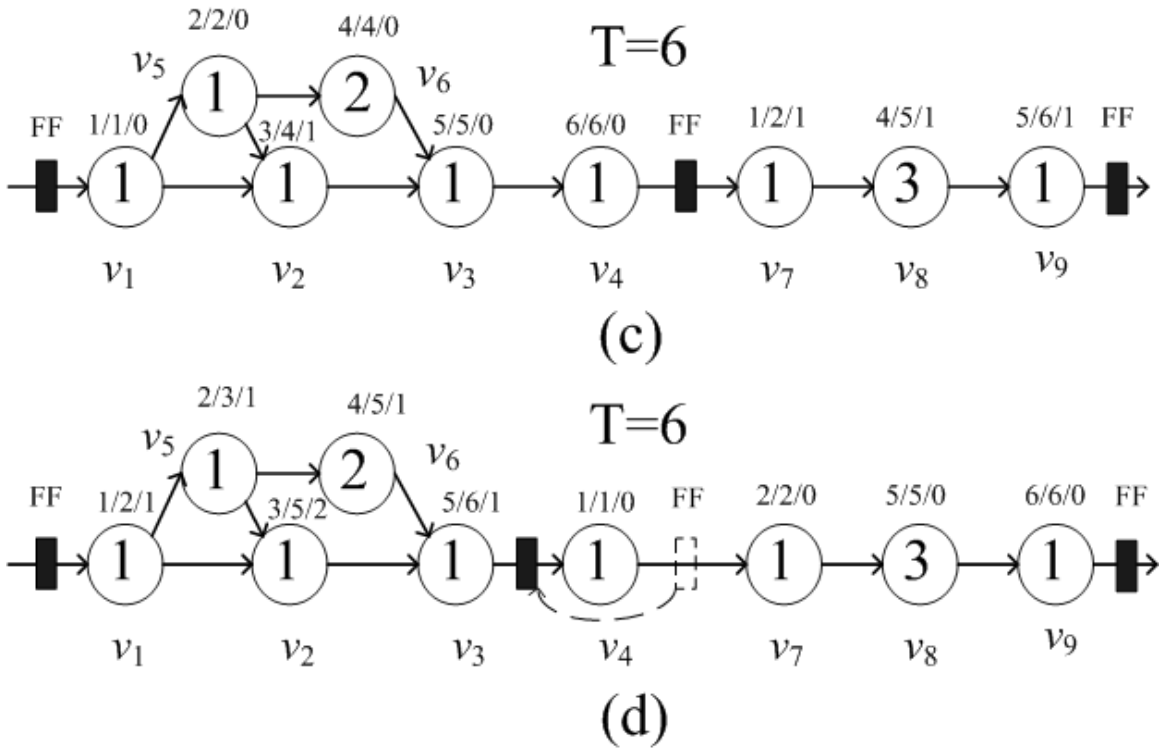
- Background
- Existing work
- Problem formulation
- **Motivating examples**
- Algorithm
  - Time Slack Budgeting for Estimating Potential Slack
  - Simultaneous Retiming and Slack Budgeting
- Experimental results
- Conclusion

Motivating Examples (1)



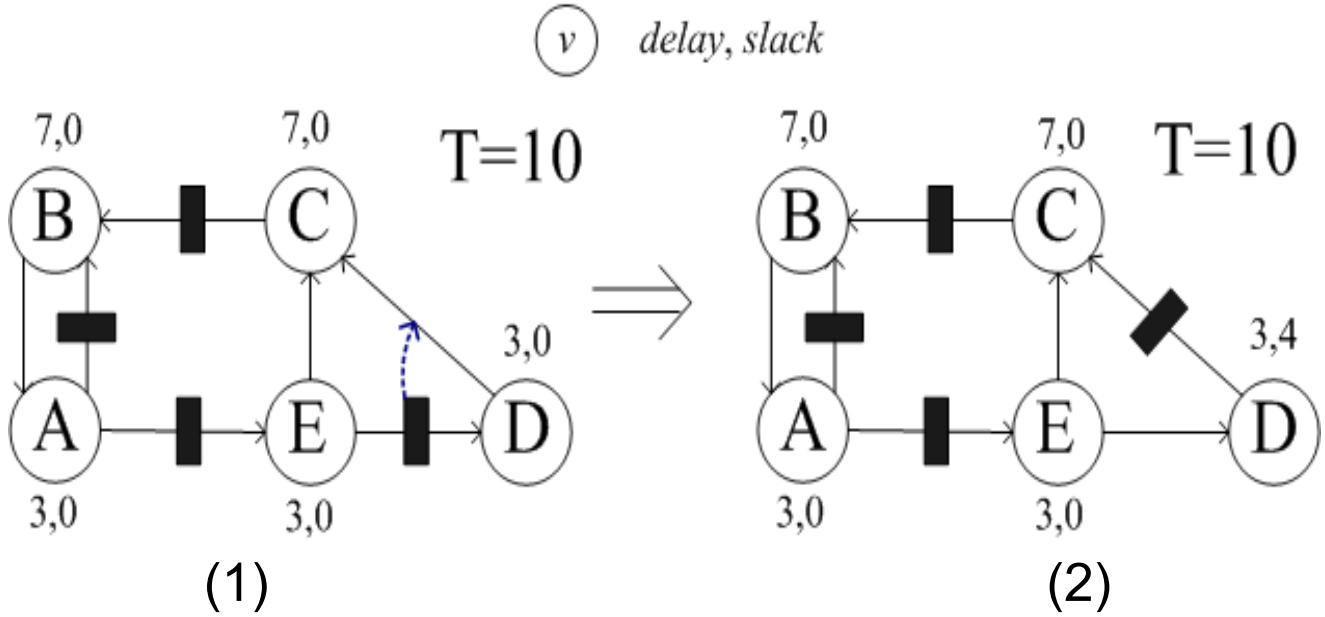
- *Scenario* I: make use of "slack" on an output.
  - In (a), there is no slack for each gate
  - the timing "slack" on output (v_1, v_3) is 8
  - with moving FF from (v_1, v_3) to (v_3, v_4), the timing "slack" can be used by the gates v_3 and v_4, as (b) shows

Motivating Examples (2)



- *Scenario* II: redistribute the slack to the sub-graph with more branches.
  - branches are actually the independent set, in which vertices are not slack sensitive to each other
  - set $\{v_6, v_2\}$ is an independent set in the left sub-graph
  - (c) has a potential slack，$s_2 + s_7 = 2$, while slack is increased from 2 to $s_2 + s_6 = 3$ in (d)

Motivating Examples (3)



- Even in a clock-minimized circuit in Fig. (1),
  - the slack in Fig. (1) is 0
  - the potential slack can be increased from 0 to 4, by moving the FF from edge ED to DC, as Fig. (2) shows
  - clock period is still minimized

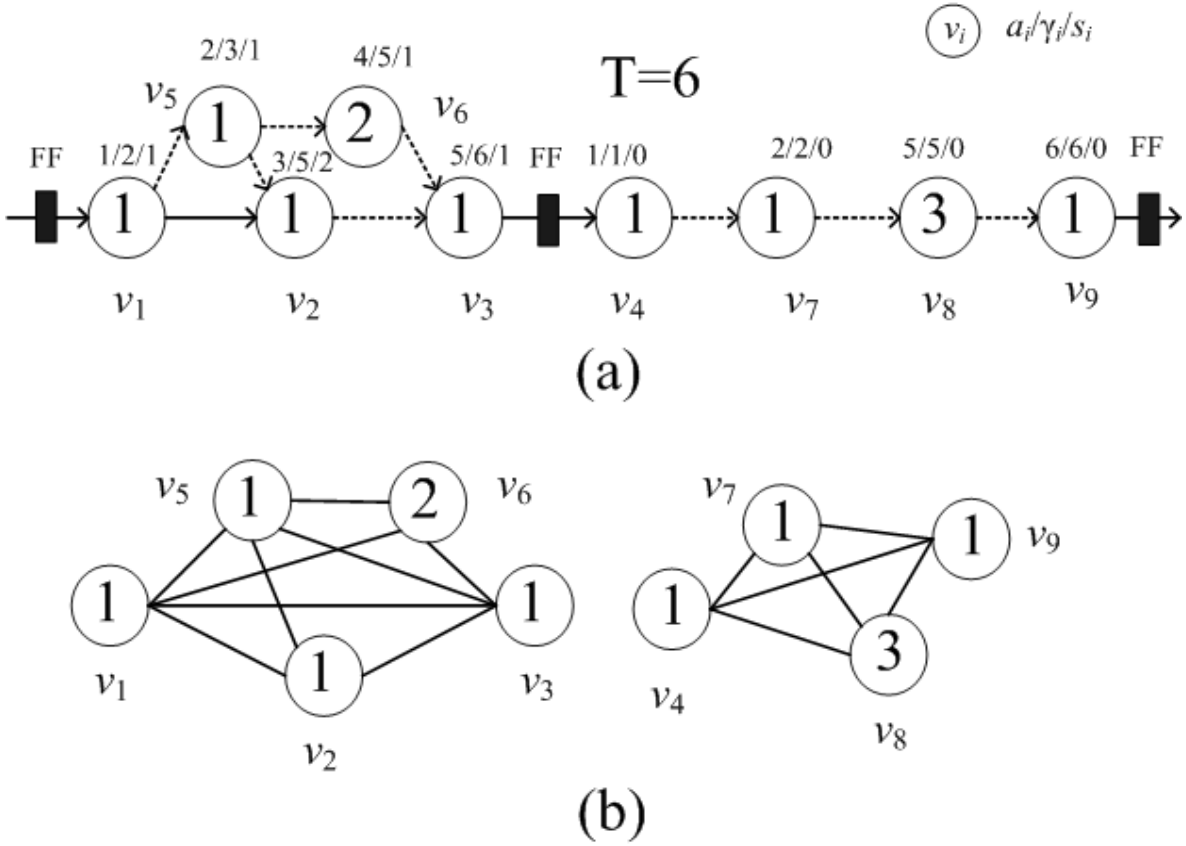
Outline

- Background
- Existing work
- Problem formulation
- Motivating examples
- **Algorithm**
  - Time Slack Budgeting for Estimating Potential Slack
  - Simultaneous Retiming and Slack Budgeting
- Experimental results
- Conclusion

Sensitive Transitive Closure Graph on Sequential Circuit

- Define edge $(v_i, v_j) \in E$ to be slack sensitive if and only if the condition (6) is satisfied.
  - $w_{ij} + r_j - r_i = 0$ and $(a_j - a_i = d_j$ or $\gamma_j - \gamma_i = d_j)$
  - where $w_{ij} + r_j - r_i = 0$ means there is no FF on edge (v_i, v_j)
- The sensitive transitive closure graph (STCG), $G_s(V, E_s)$ of G
  - there is an edge (v_i, v_j) in G_s if and only if there is a directed path from v_i to v_j in G and the path consists of only sensitive edges
- Two vertices v_i and v_j are called slack insensitive if there is no edge from v_i to v_j or from v_j to v_i in G_s
- A set $SI = \{v_1, v_2, \ldots, v_k\}$ of vertices is called slack insensitive if all vertices in the set are pair-wise slack insensitive
  - *SI* is an independent set of graph G_s

Examples for STCG



- (a) highlights the slack sensitive edges with dotted lines
- (b) is the undirected STCG which consists of two connected sub-graphs

Estimating Potential Slack (EPS)

1. $p_s = 0$;
2. let vertices with positive slack be Q;
3. While (Q is nonempty) {
4. build induced graph of $\bar{G}_s$ on Q;
5. find maximum independent set SI;
6. $\mathcal{S} = min\{s_i \mid v_i \in SI\}$
7. p_s += $\mathcal{S}$ * sizeof(SI);
8. Foreach $v_i \in Q$ { d_i += $\mathcal{S}$; }
9. update $a/\gamma/s$ and Q;
10. }
11. recover delay d of graph G;
12. output potential slack p_s;

Subroutine for estimating potential slack (EPS)

- A well-know polynomial-time MIS solver by Dharwadker is employed in step 4

Outline

- Background
- Existing work
- Problem formulation
- Motivating examples
- **Algorithm**
  - Time Slack Budgeting for Estimating Potential Slack
  - Simultaneous Retiming and Slack Budgeting
- Experimental results
- Conclusion

Simultaneous Retiming and Slack Budgeting

- Simultaneous slack budgeting and retiming (SSBR) problem is expressed as follows.

$$\begin{aligned} & Max \quad p_s && (7) \\ s.t. \quad & w_{i,j} + r_j - r_i \geq 0, \ \forall (v_i, v_j) \in E && (8) \\ & a_i \leq T, \ \forall v_i \in V && (9) \end{aligned}$$

  - Formula (8) shows the constraints that the number of FFs should not be negative
  - Formula (9) constrains that the delay of critical path must satisfy the timing constraint *T*

- An incremental algorithm for SSBR is proposed
  - relocate the positions of FFs iteratively without violating the timing constraint
  - estimate the potential slack for each relocation
  - until the potential slack P_s is maximized

Possible Movements

- Let the possible movement m be a FF relocation which satisfies the timing constraint
- A movement m represented by label r
  - m is positive when r is positive, otherwise is negative
- If the movement m_i is positive, the required condition of being possible movement is

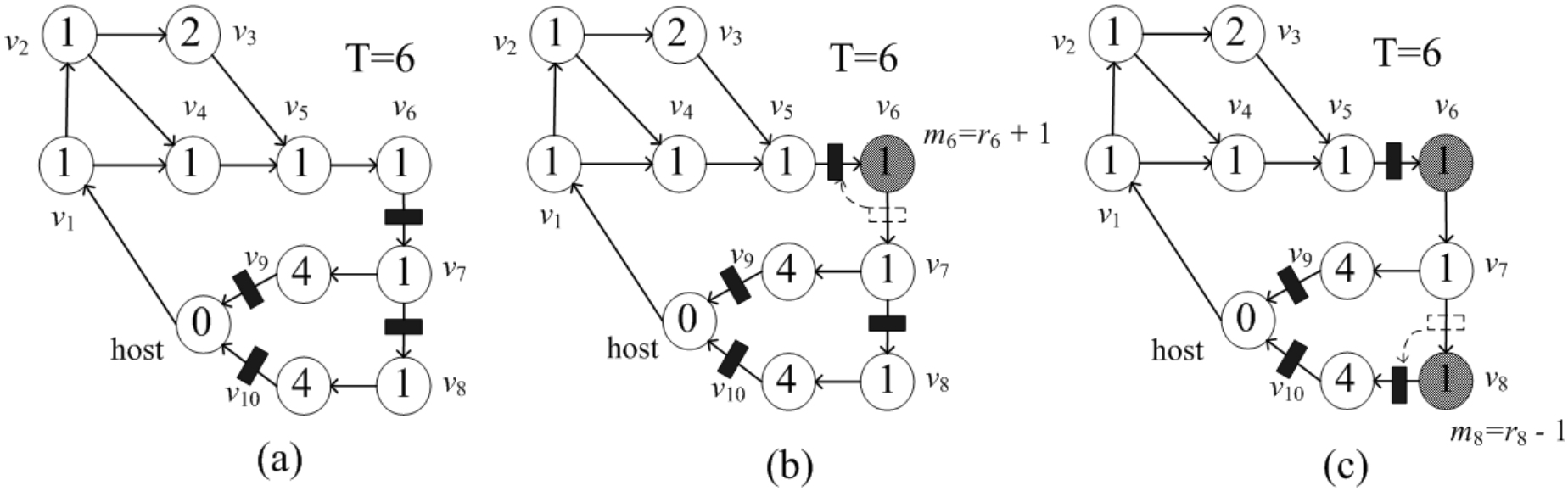
$$\gamma_j - d_j - d_i \geq 0, \quad v_j \in FO(v_i)$$

  - where γ_j is the require time, and $FO(v_i)$ is the set of adjacent vertices on fan-out edges of v_i
- If the movement m_i is negative, the required condition of being possible movement is

$$a_j + d_i \leq T, \quad v_j \in FI(v_i)$$

  - where a_j is the arrival time, and $FI(v_i)$ is the set of adjacent vertices on fan-in edges of v_i

Examples for SSBR algorithm



	M, SI, P_s	comments
1	φ, $\{v_4, v_7, v_8\}$, 3	$r_i = 0$, $v_i \in V$
2	$\{r_6+1, r_8-1\}$, $\{v_4, v_7, v_8\}$, 3	after step 3 in Fig. 7
3	$\{r_6+1, r_8-1\}$, $\{ v_3, v_4, v_8\}$, 4	taking movement r_6+1, and $r_6=1$
4	$\{r_6+1, r_8-1\}$, $\{ v_3, v_4, v_8, v_{10}\}$, 8	taking movement r_6+1, and $r_6=1$, $r_8=-1$
5	$\{ r_6-1, r_8+1\}$, $\{ v_3, v_4, v_8, v_{10}\}$, 8	after step 3 in Fig. 7
6	φ, $\{ v_3, v_4, v_8, v_{10}\}$, 8 (optimal)	condition (10) is unsatisfied, and *isImproved* is false.

Outline

- Background
- Existing work
- Problem formulation
- Motivating examples
- Algorithm
  - Time Slack Budgeting for Estimating Potential Slack
  - Simultaneous Retiming and Slack Budgeting
- **Experimental results**
- Conclusion

Test Cases

TABLE I
The Characteristics of Test cases

Test case name	*Gates count*	*Edges count*	*Max outputs*	*Max inputs*	T_{min}
s27.test	11	19	4	2	20
s208.1.test	105	182	19	4	28
s298.test	120	250	13	6	24
s382.test	159	312	21	6	44
s386.test	160	354	36	7	64
s344.test	161	280	12	11	46
s349.test	162	284	12	11	46
s444.test	182	358	22	6	46
s526.test	194	451	13	6	42
s526n.test	195	451	13	6	42
s510.test	212	431	28	7	50
s420.1.test	219	384	31	4	40
s832.test	288	788	107	19	98
s820.test	290	776	106	19	92
s641.test	380	563	35	24	238
s713.test	394	614	35	23	262
s838.1.test	447	788	55	4	80
s1238.test	509	1055	192	14	110
s1488.test	654	1406	56	19	166

- 19 cases from the ISCAS89 benchmarks are tested
- The minimum period is also given by H. Zhou's retiming algorithm

Experimental Results (1)

TABEL II

Experimental Results

T_{min} – the minimum period of the circuit by Zhou's min-period retiming [3]; P_0 – the potential slack after the min-period retiming

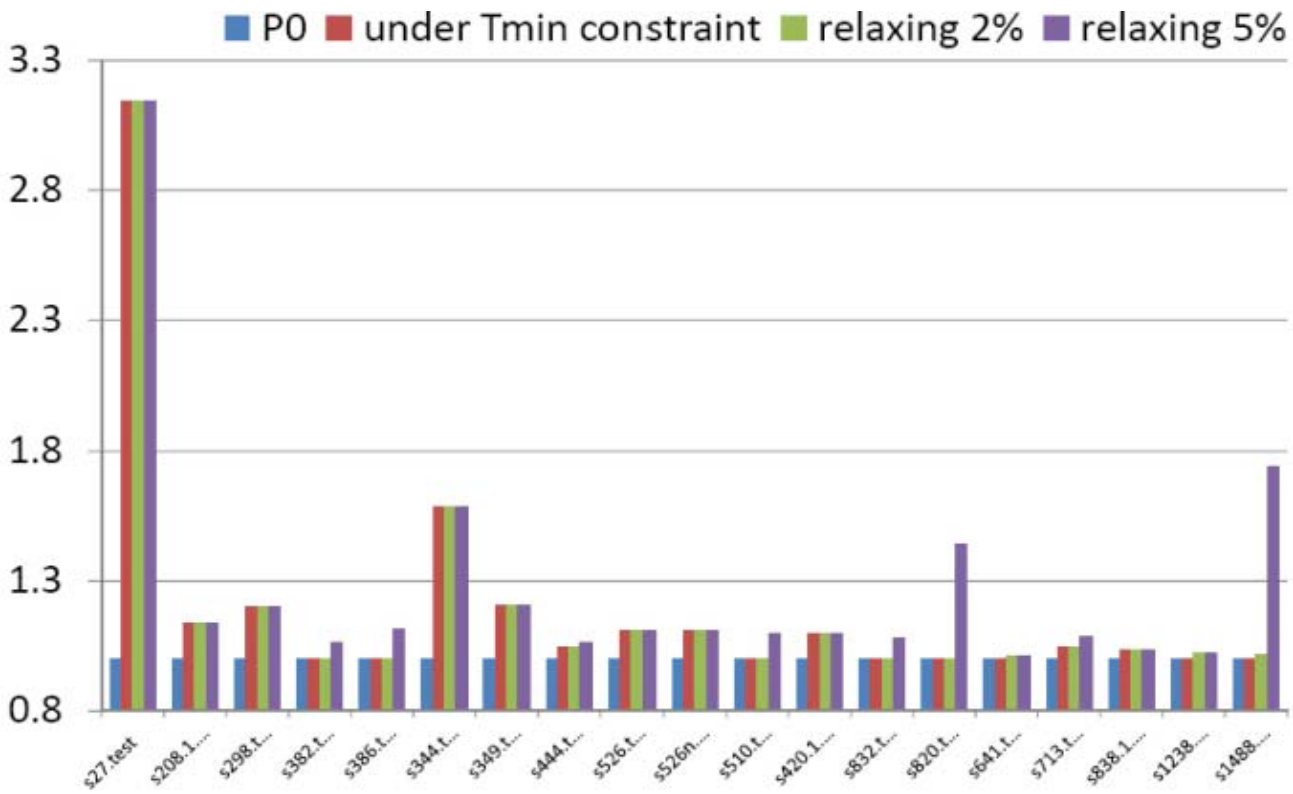
Under T_{min} constraint – max-potential-slack retiming with timing constraint T_{min}; *Relaxing x%* -- max-potential-slack retiming with timing constraint $T_{min} \times (1+x\%)$

period – the actual clock period after under max-potential-slack retiming with timing relaxation

Test cases	*Zhou's [3]*		*Under T_{min} constraint*			*Relaxing 2%*				*Relaxing 5%*			
	T_{min}	P_0	P_z	*increase*	*runtime(s)*	*period*	P_z	*increase*	*runtime(s)*	*period*	P_z	*increase*	*runtime(s)*
s27.test	20	14	44	214.29%	0.00	20	44	214.29%	0.00	20	44	214.29%	0.00
s208.1.test	28	422	480	13.74%	1.30	28	480	13.74%	1.33	28	480	13.74%	1.30
s298.test	24	522	628	20.31%	6.64	24	628	20.31%	6.66	24	628	20.31%	6.37
s382.test	44	1204	1204	0.00%	11.58	44	1204	0.00%	10.25	46	1282	6.48%	5.76
s386.test	64	724	724	0.00%	1.11	64	724	0.00%	1.01	66	806	11.33%	4.42
s344.test	46	742	1178	58.76%	8.19	46	1178	58.76%	7.87	46	1178	58.76%	8.21
s349.test	46	728	880	20.88%	7.72	46	880	20.88%	8.34	46	880	20.88%	6.34
s444.test	46	1208	1264	4.64%	33.55	46	1264	4.64%	39.65	48	1284	6.29%	13.92
s526.test	42	1324	1472	11.18%	27.08	42	1472	11.18%	23.49	42	1472	11.18%	13.19
s526n.test	42	1306	1446	10.72%	57.80	42	1446	10.72%	71.41	42	1446	10.72%	11.81
s510.test	50	1224	1224	0.00%	12.21	50	1224	0.00%	11.62	52	1342	9.64%	25.60
s420.1.test	40	1318	1444	9.56%	42.68	40	1444	9.56%	41.99	40	1444	9.56%	40.02
s832.test	98	6130	6130	0.00%	61.01	98	6130	0.00%	42.51	102	6618	7.96%	462.27
s820.test	92	5746	5746	0.00%	42.24	92	5746	0.00%	48.18	92	8302	44.48%	503.07
s641.test	238	9624	9624	0.00%	329.63	242	9758	1.39%	441.96	242	9758	1.39%	1979.38
s713.test	262	11234	11760	4.68%	431.90	262	11760	4.68%	504.08	270	12174	8.37%	483.73
s838.1.test	80	7038	7292	3.61%	1674.93	80	7292	3.61%	1840.07	80	7292	3.61%	1075.06
s1238.test	110	9260	9260	0.00%	3318.06	112	9462	2.18%	2178.99	112	9462	2.18%	8626.44
s1488.test	166	11436	11436	0.00%	884.97	168	11662	1.98%	2843.89	172	19880	73.84%	6255.33
Average	80.95			19.6%	365.93	81.37(+0.52%)		19.89%	427.54	82.63(+2.08%)		28.16%	1027.49

- Our algorithm improves potential slack averagely 19.6% without degrading the circuit performance
- Furthermore, at the expense of a small amount of timing performance, 0.52% and 2.08%, the potential slack is increased averagely by 19.89% and 28.16% separately

Experimental Results (2)



- Normalized P_0 and the optimized potential slack under three different timing constraints.

Conclusion

- We introduce the slack sensitive closure graph in combinatorial circuits to synchronous sequential circuits
- A MIS-based slack budgeting is performed
- A simultaneously incremental retiming and slack budgeting algorithm is proposed,
  - which is heuristic yet effective
- Comparing with the potential slack obtained by H. Zhou's min-period retiming
  - our algorithm can improve the potential slack averagely by 19.6% without degrading timing performance
  - with timing constraint relaxation of 2% and 5%, the potential slack is increased averagely by 19.89% and 28.16% separately
  - gives a view of tradeoff between a fast and low-power design

Thank You
Q&A