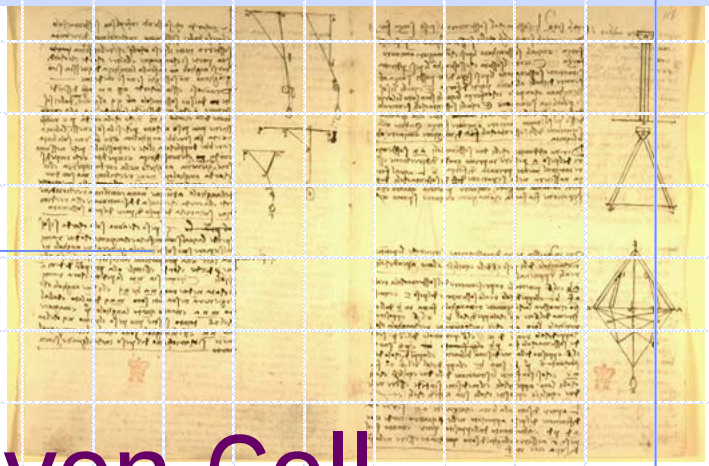


Ultra-Fast Interconnect Driven Cell Cloning For Minimizing Critical Path Delay

Zhuo Li[1], David A. Papa[2,1], Charles J. Alpert[1], Shiyan Hu[3], Weiping Shi[4], C. N. Sze[1] and Ying Zhou[1]

IBM Austin Research Lab[1]

Dept. EECS, University of Michigan[2]

Dept. ECE, Michigan Technological University[3]

Dept. ECE, Texas A&M University[4]

Best Value Toys

Global placement

Routability analysis / recovery

Buffering

Cell movement

Vt assignment

Gate Sizing

Layer assignment

$15K

Cloning?

Best Value Toys

Global placement

Routability analysis / recovery

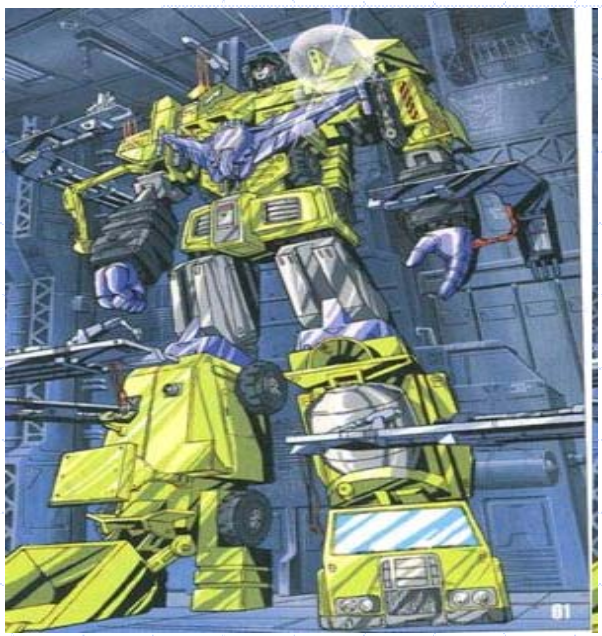
Buffering

Cell movement

Vt assignment

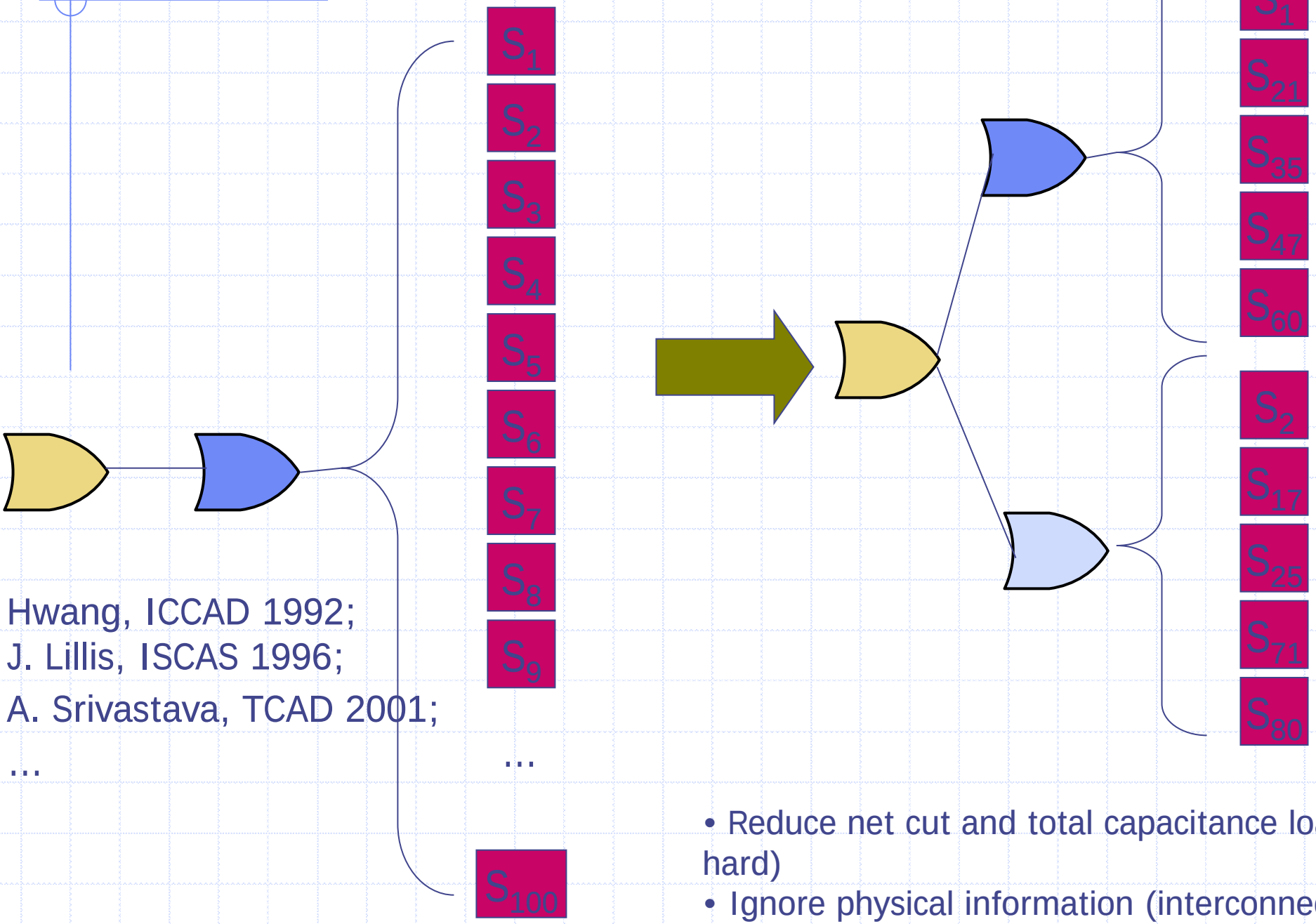
Gate Sizing

Layer assignment



Cloning

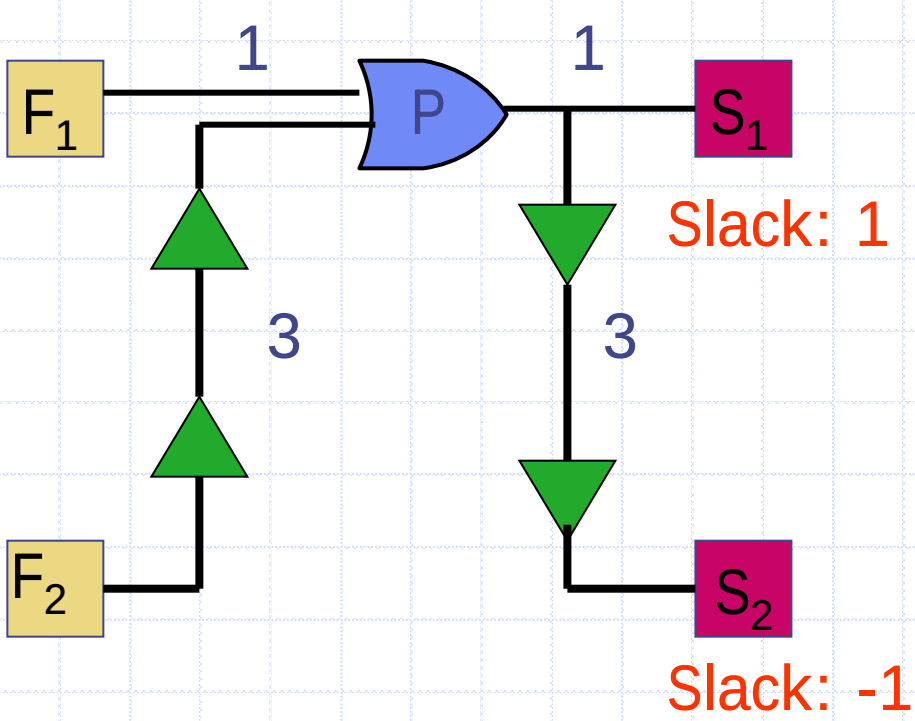
Cloning in Logic Synthesis

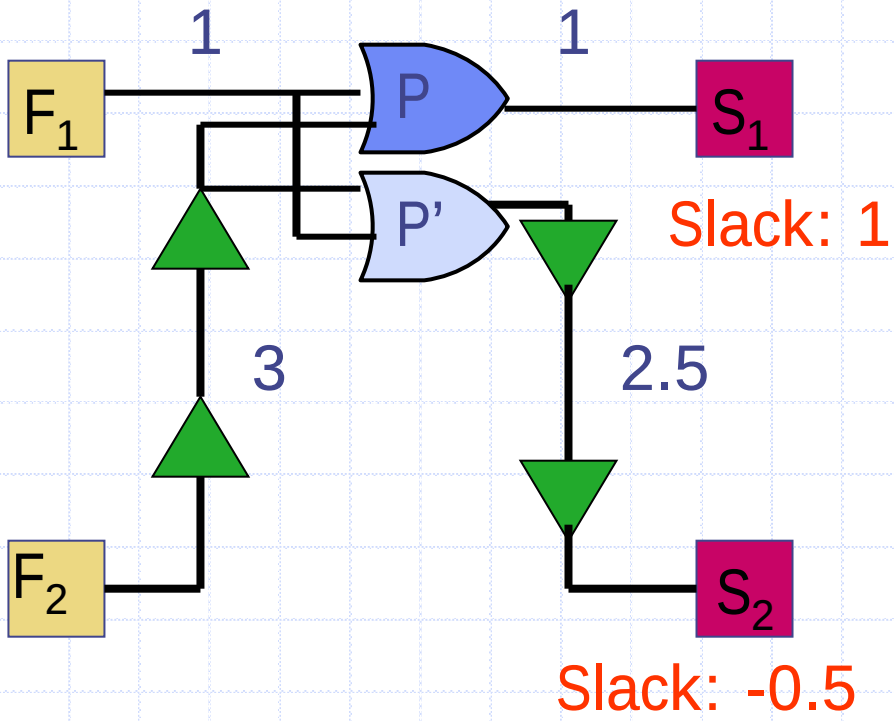


Hwang, ICCAD 1992;
J. Lillis, ISCAS 1996;
A. Srivastava, TCAD 2001;
...

- Reduce net cut and total capacitance load (NP-hard)
- Ignore physical information (interconnect, buffering, ...)

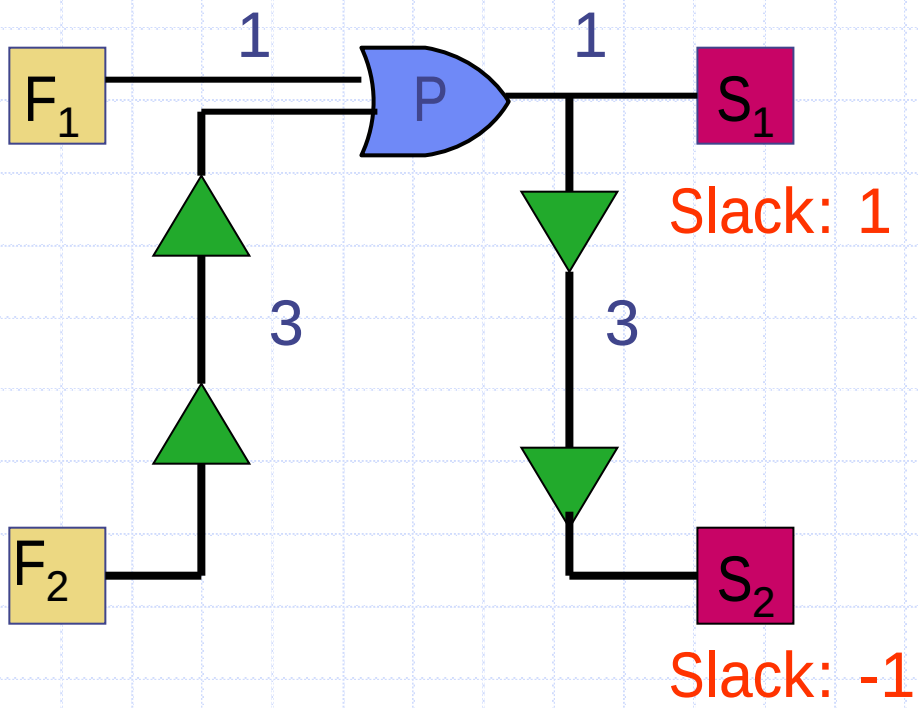
Interconnect Driven Cloning

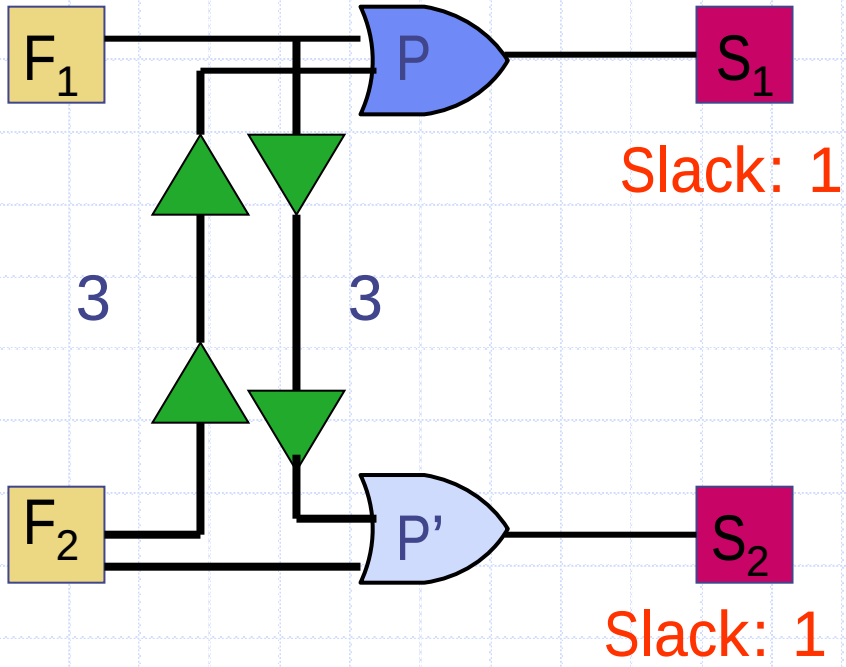




$AT(D_1) = AT(D_2) = 0$ $RAT(S_1) = RAT(S_2) = 5$

Interconnect Driven Cloning





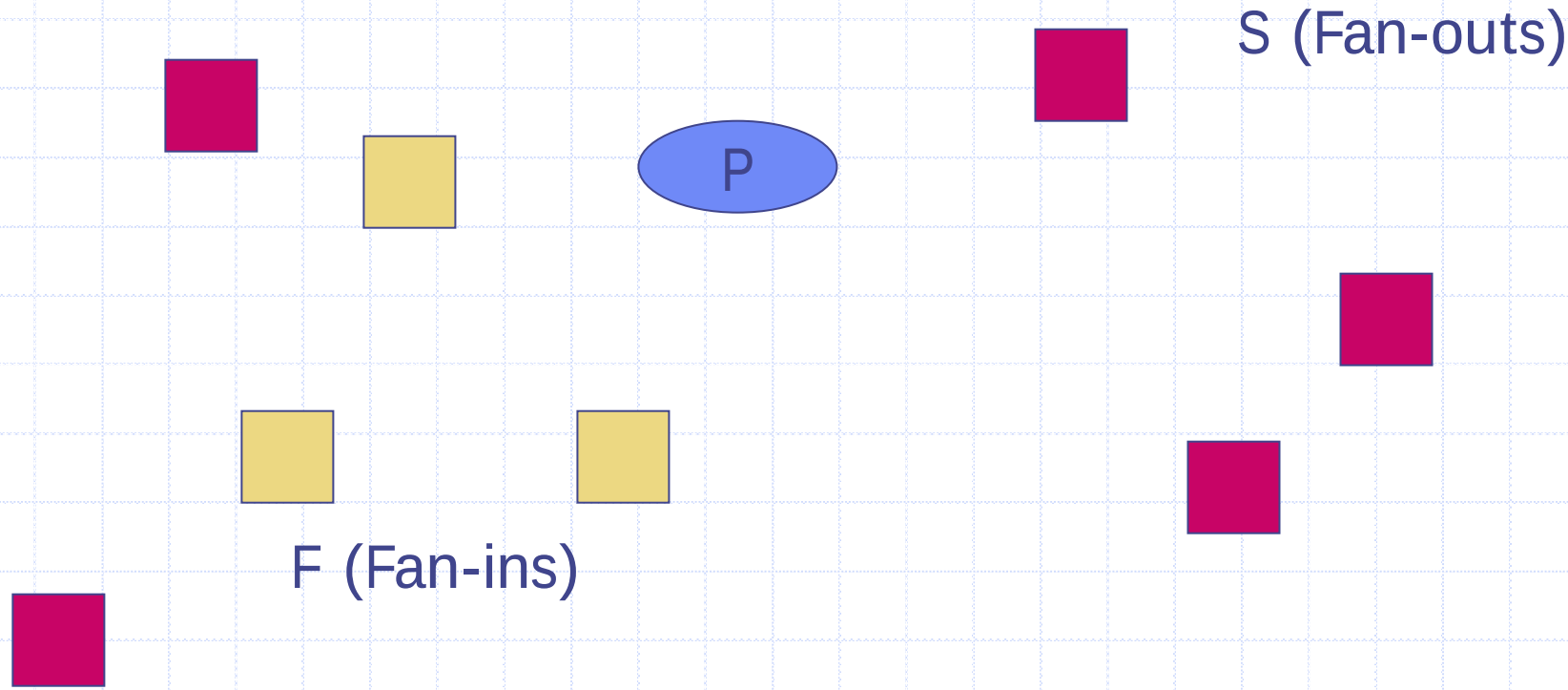
$AT(D_1) = AT(D_2) = 0$ $RAT(S_1) = RAT(S_2) = 5$

Our Contribution

- Find the “optimal” partitioning and placement of the original and duplicated gates
  - Assuming linear-buffer-delay model
  - O(n) algorithm when original gate is fixed
  - O(nlogn) algorithm when original gate is movable
  - Just focus on worst slack
  - For interconnect delay dominant sub-circuit
  - Extensions
- Back of envelop filter
  - Logic based cloning: High fan-outs/capacitive load
  - Physical based cloning: special fan-out location distributions

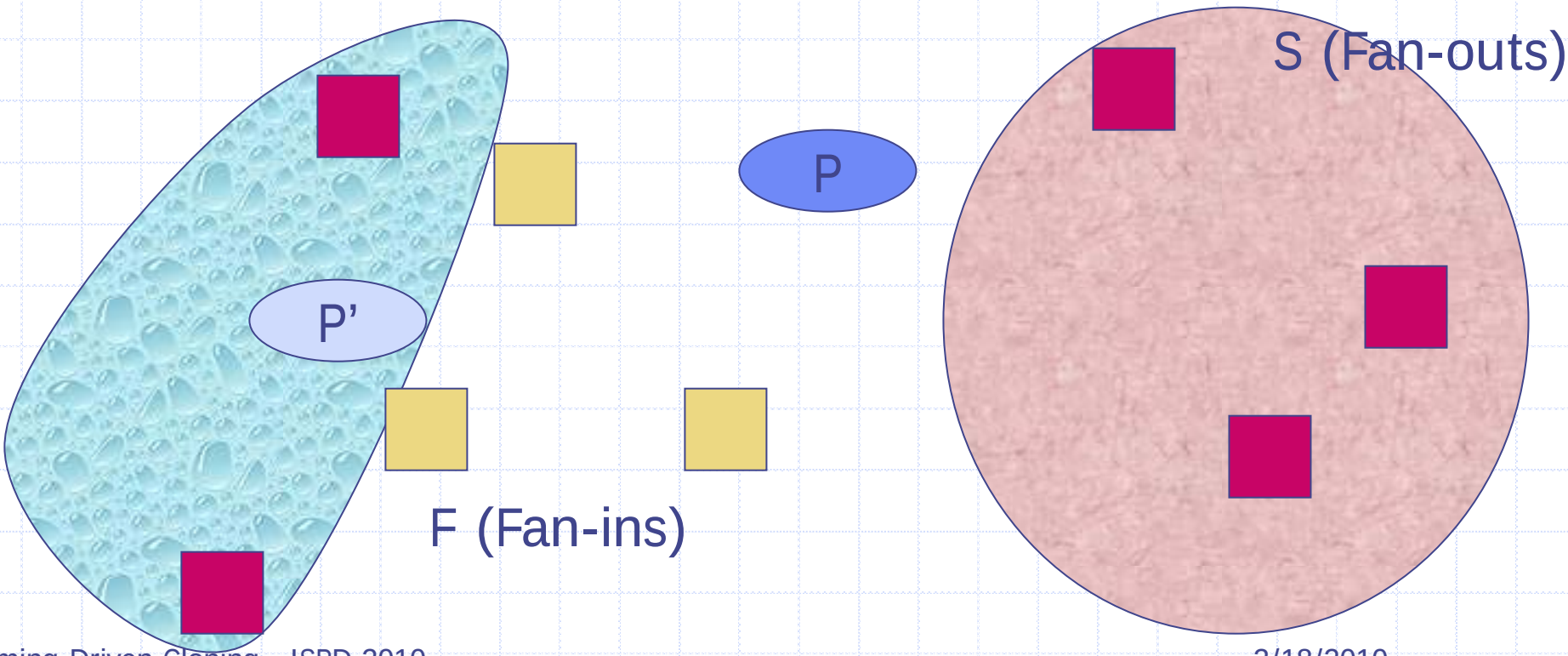
Cloning Problem

- A sub-circuit
- Two-pin timing arcs $D = \sigma \cdot dis(G_1, G_2)$
- Clone P to P', find the partitioning of *S* and locations of *P* and *P'*, to maximize sub-circuit slack



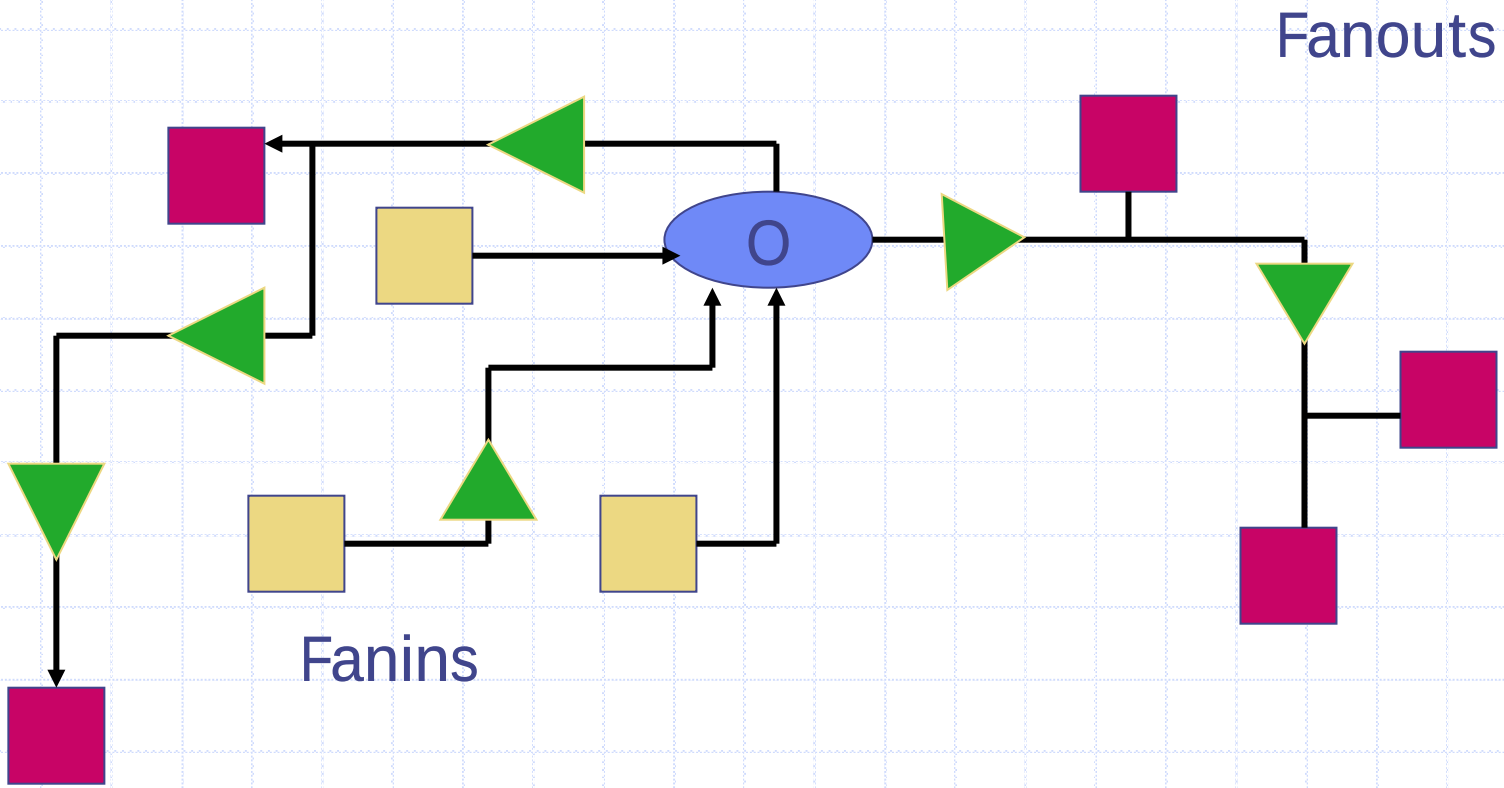
Cloning Problem

- A sub-circuit
- Two-pin timing arcs $D = \sigma \cdot dis(G_1, G_2)$
- Clone P to P', find the partitioning of *S* and locations of *P* and *P'*, to maximize sub-circuit slack



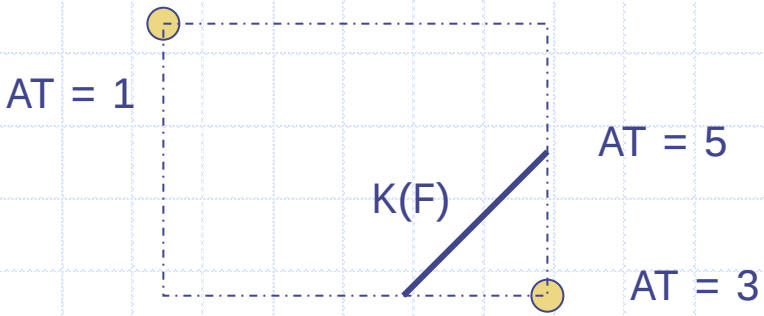
Cloning Problem

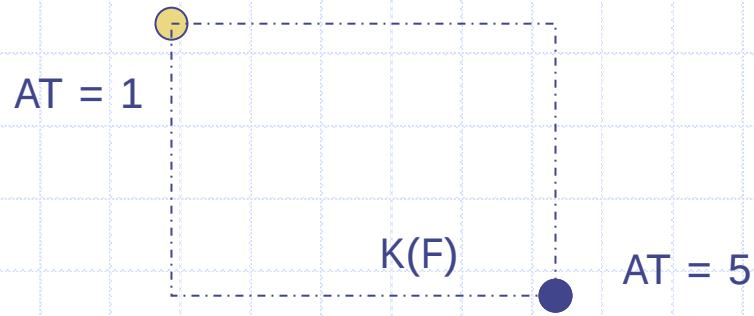
- Reduce to a gate placement problem when the partitioning is given (RUMBLE ISPD08 and Pyramids ICCAD08)
- Perform real buffering after cloning

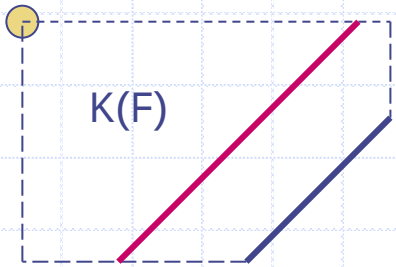


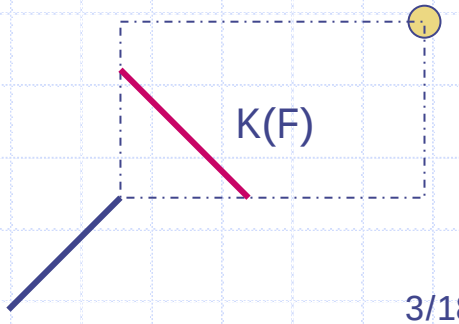
Arrival Time Arc

- Each fan-in gate has an arrival time $AT(F_i)$
- For each physical point v, $AT(v) = max(AT(F_i) + \tau \cdot Dis(AT(F_i), v)$
- The set of points minimizing $AT(v)$ is ***arrival time arc*** $K(F)$
- $K(F)$ is either an Manhattan arc or a single point
- Similar to Deferred Merge Embedding (DME)
- $K(F)$ is also the bottom of a trough $AT(v)$ (overlapping of a set of reverse pyramids)



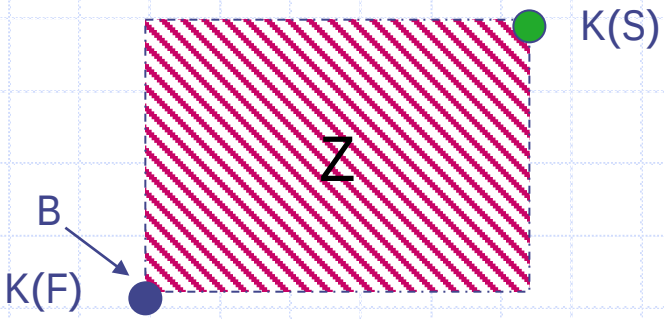


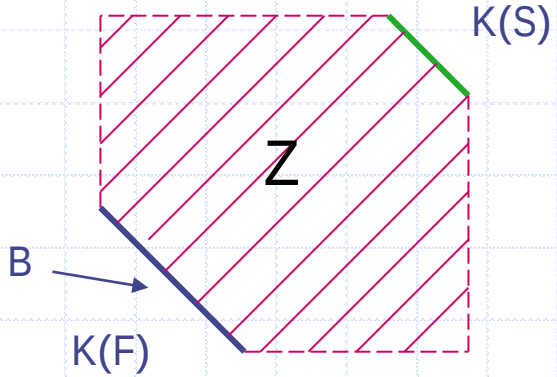


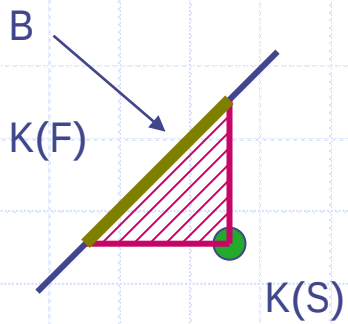


Best Region and Best Arrival Time Arc

- *K(S)* : *required arrival time arc* (maximizing *RAT(v)*)
- ***Best region*** *Z* : every point inside this region has maximum sub-circuit slack (constructed with *K(F)* and *K(S)*)
- ***Best Arrival Time arc*** *B* is the intersection of Best Region and Arrival Time Arc
- Define $K(F_i)$ as the arrival time arc for $F_1, \ldots, F_i$,
- O(n) time to compute *K(F)*, *K(S), Z and B.* Also O(n) time to compute all $K(F_i)$ and $K(S_i)$, instead of $O(n^2)$ time.

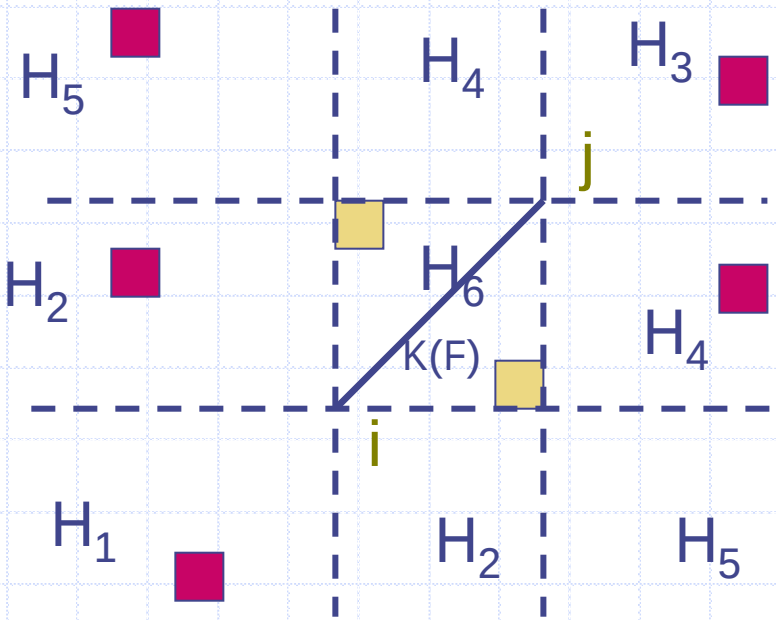


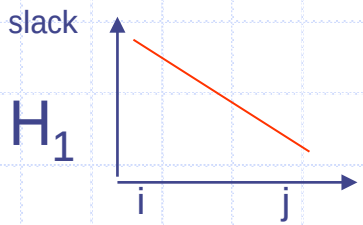


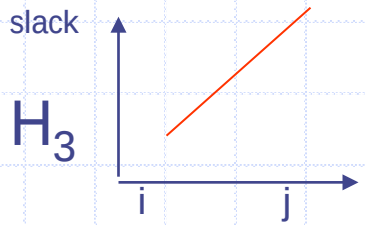


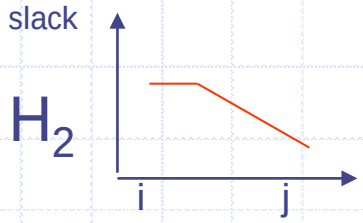
Case 1: P is movable

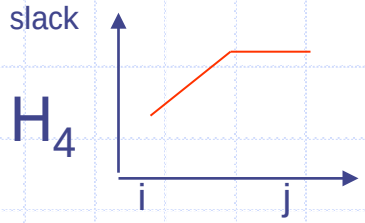
- No matter what the partitioning is, one can place P and P' on best arrival time arc, while still achieving the best slack
- Divide the whole plane into 6 regions based on slack cuves

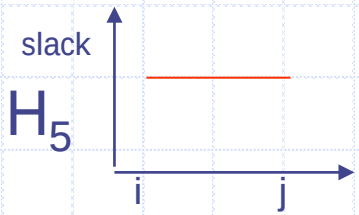


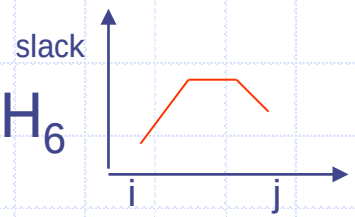






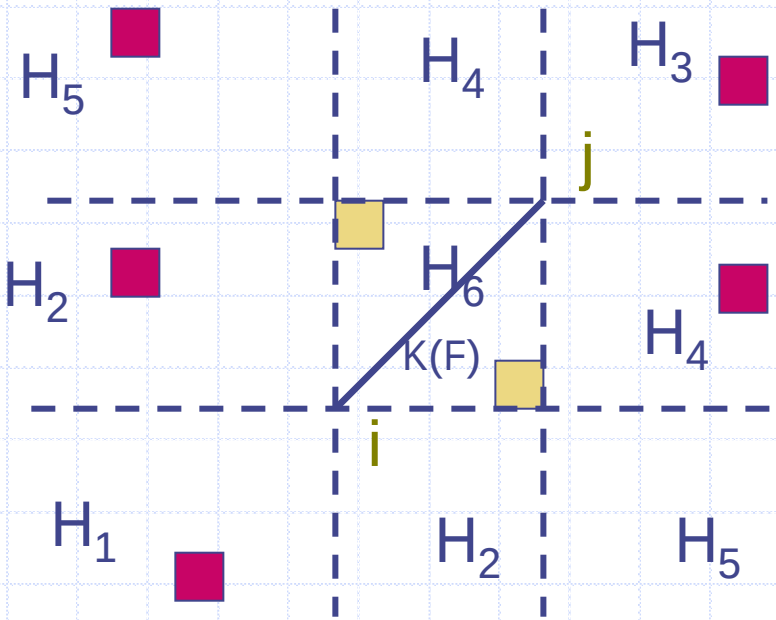


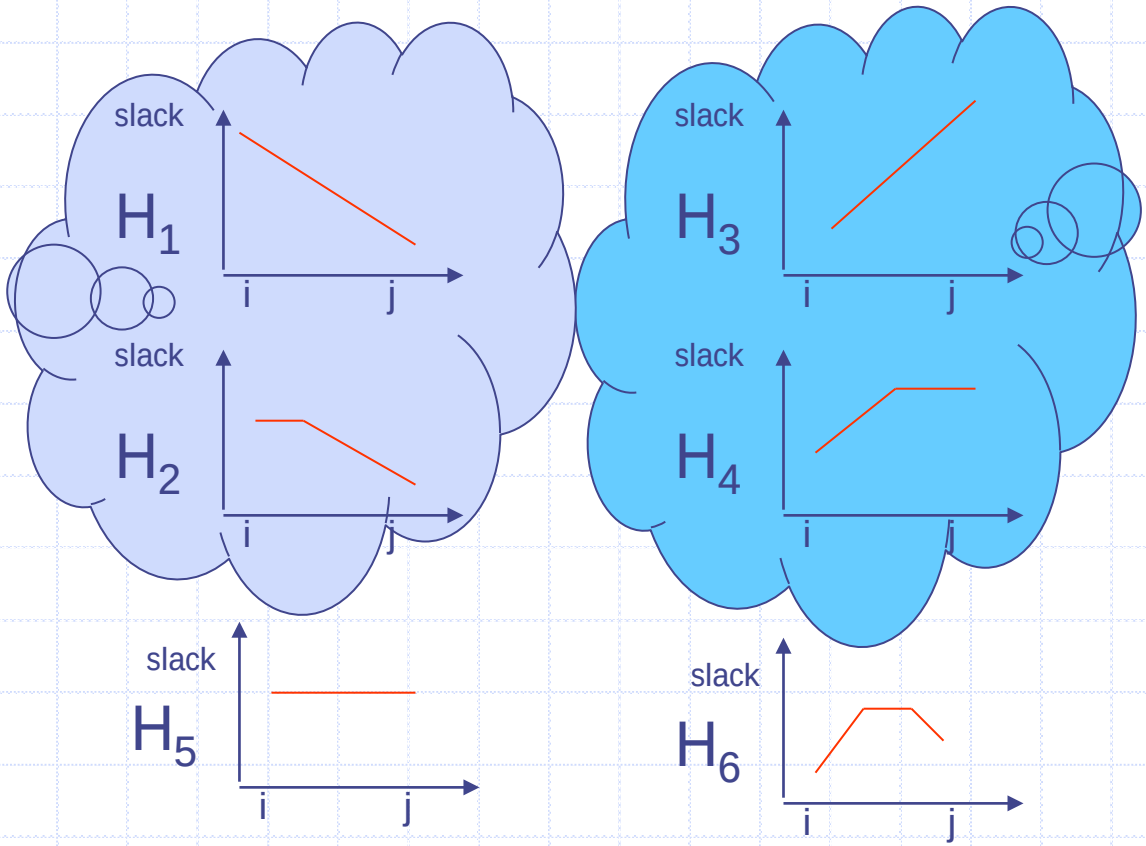




Case 1: P is movable (Cont.)

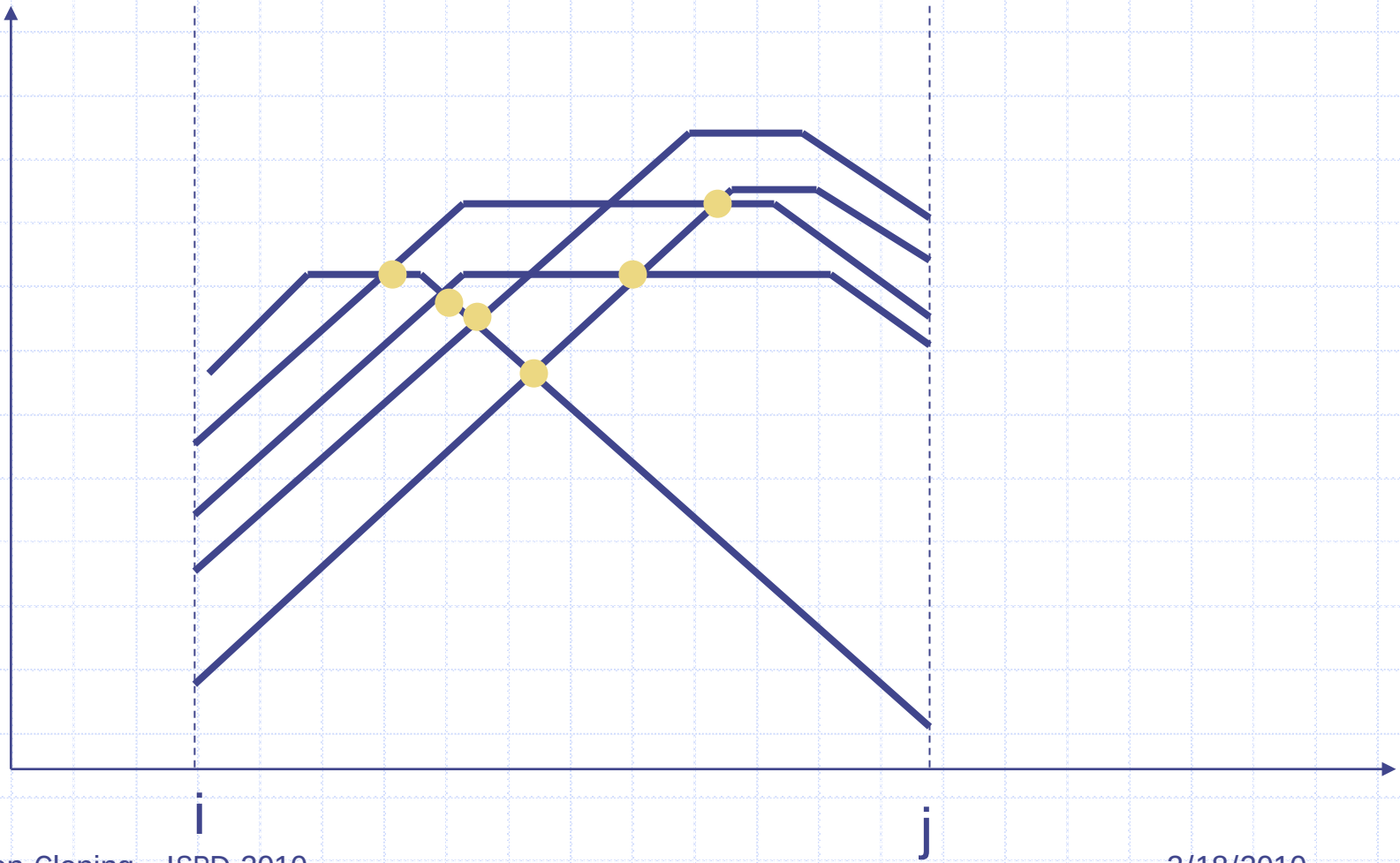
- If no gates in H_6, O(n) time algorithm





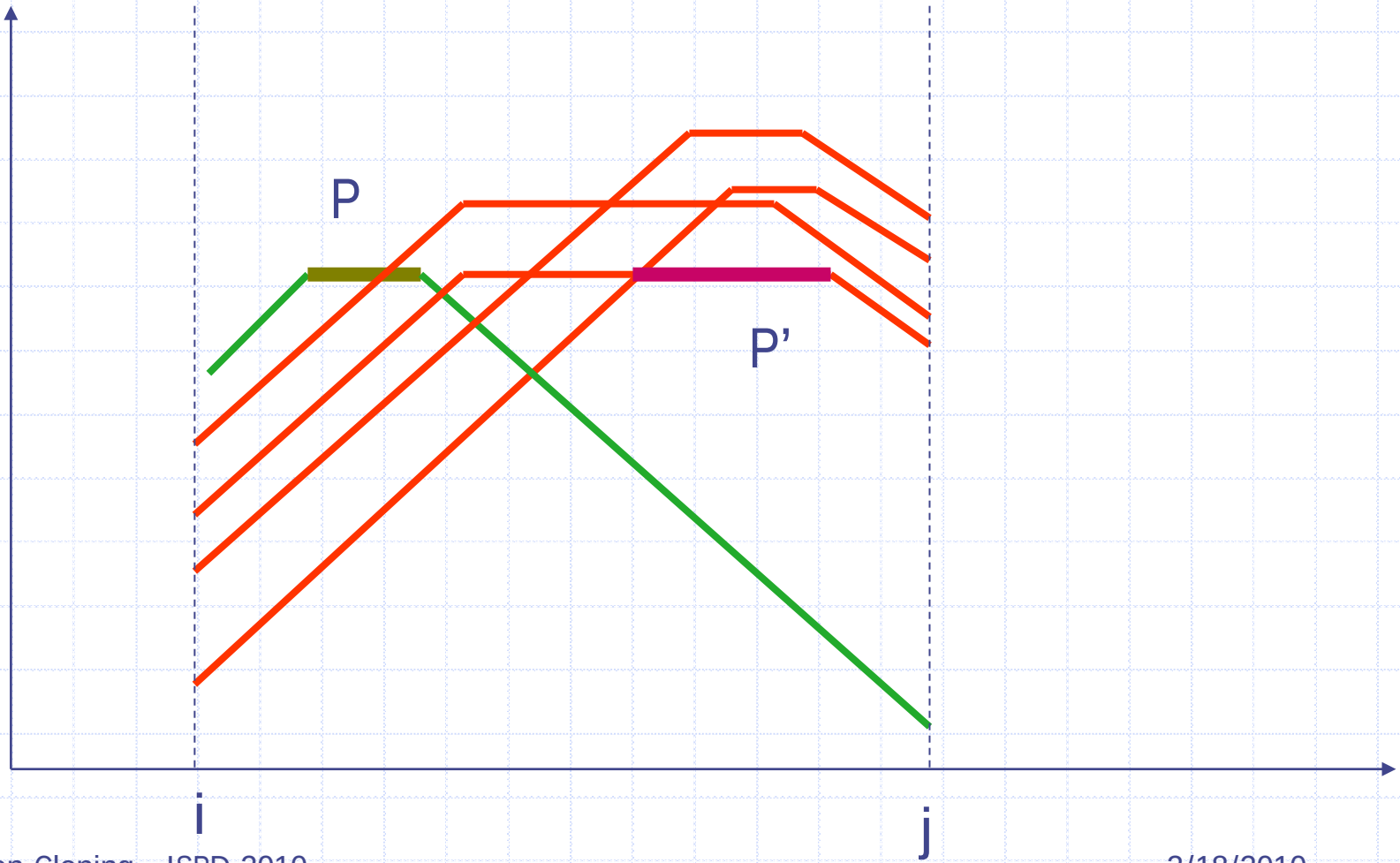
Case 1: P is movable (Cont.)

- If there are gates in H_6, treat all slack curves as 3-segment trapezoid-like curves
- O(n) time algorithm



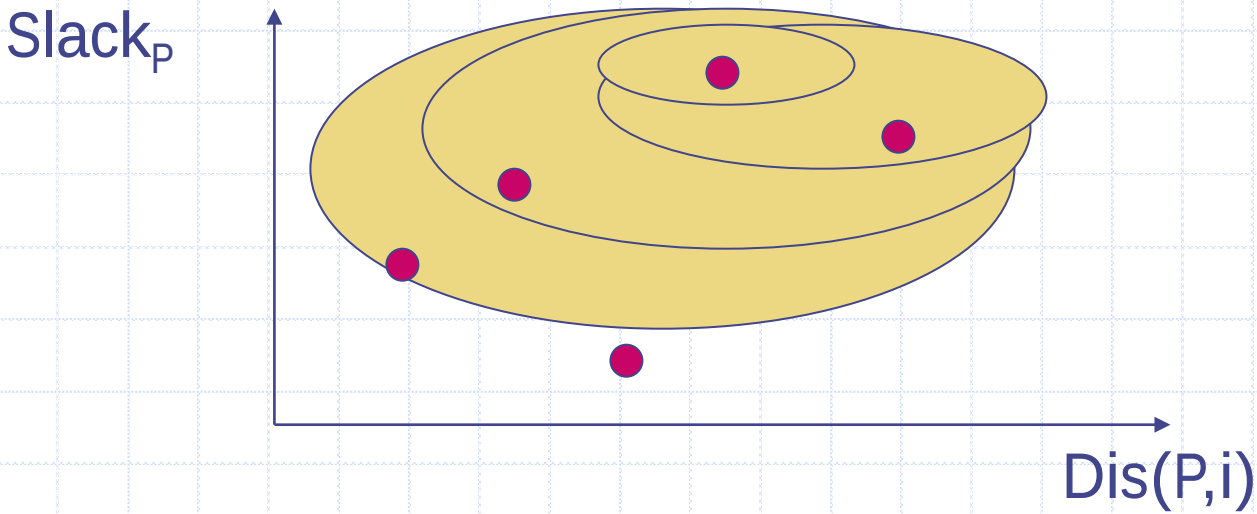
Case 1: P is movable (Cont.)

- If there are gates in H_6, treat all slack curves as 3-segment trapezoid-like curves
- O(n) time algorithm



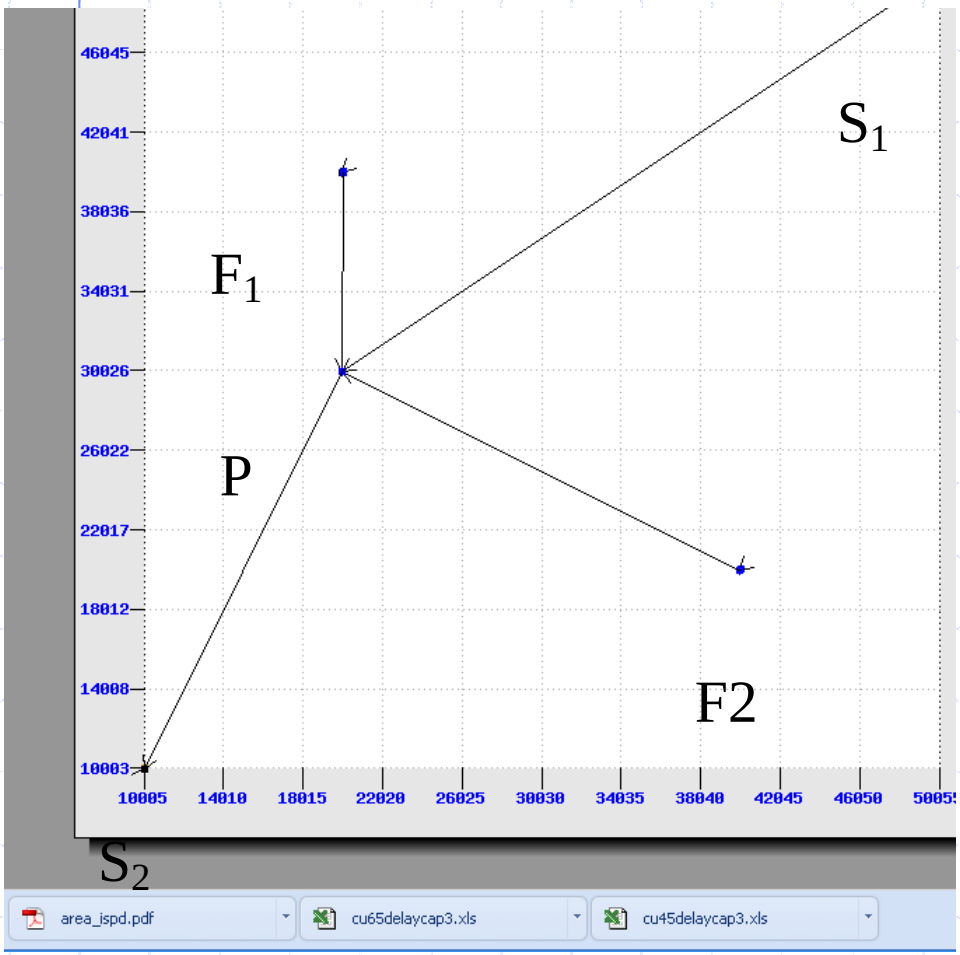
Case 2: P is fixed

- *P* may not be on *K(F)*
- $Slack_P(i) = RAT(i) - \tau \cdot Dis(P, i) - AT(P)$
- At most O(n) partitions since there are only n possible worst slack values for any partitioning
- Sort S_i accordingly
- Let *P* drive the set of fan-outs $\{S_1\}$, $\{S_1, S_2\}$, $\{S_1, S_2, S_3\}$, ...
- O(nlogn) time algorithm

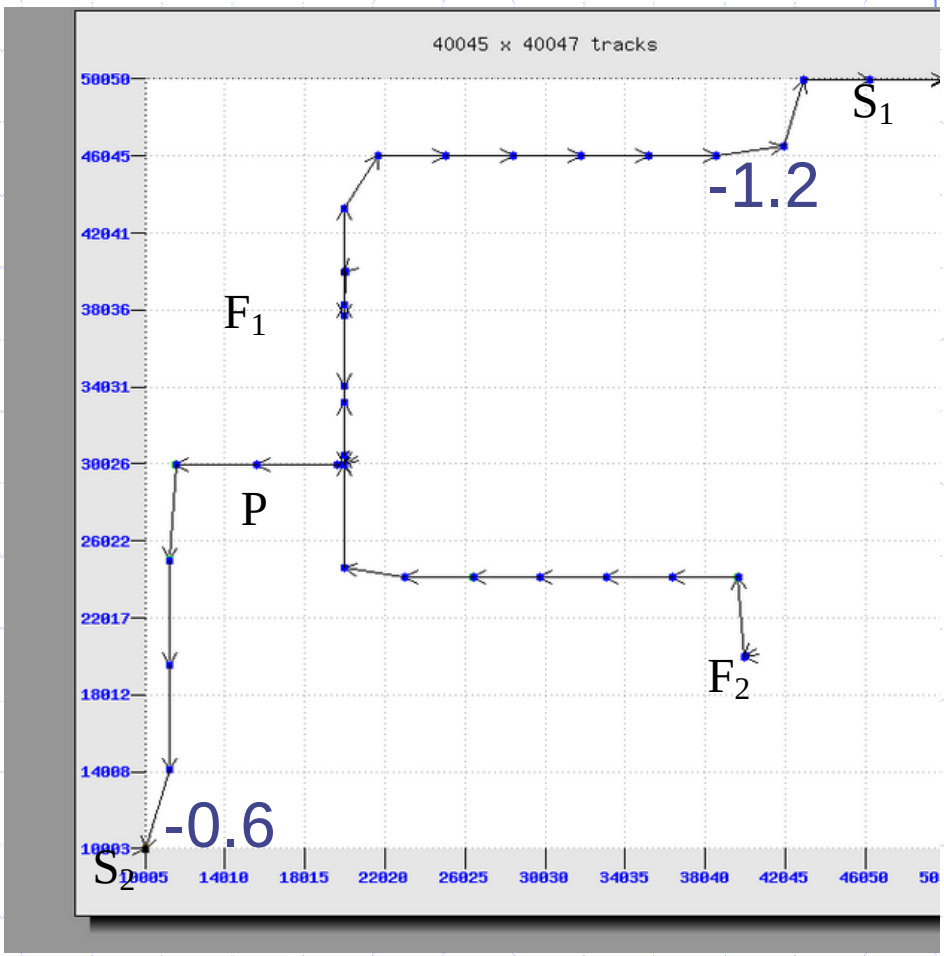


One Example

Original circuit

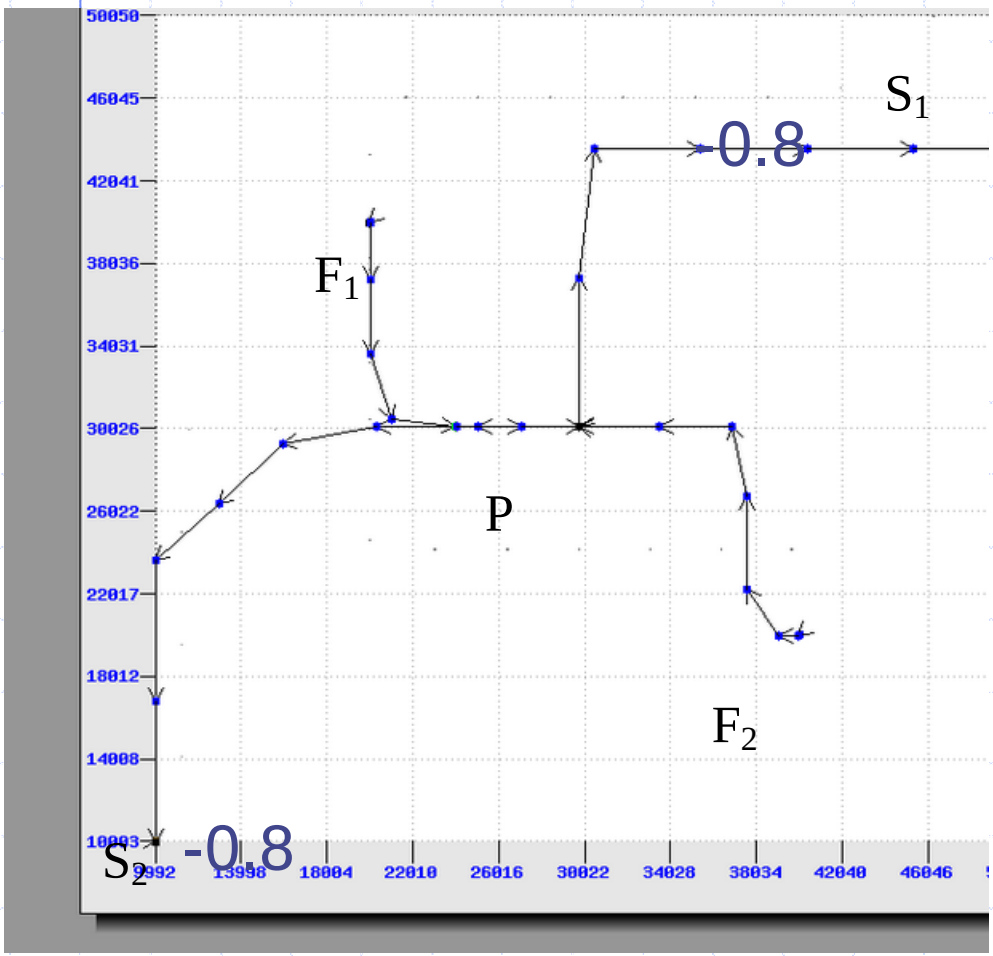


After buffering

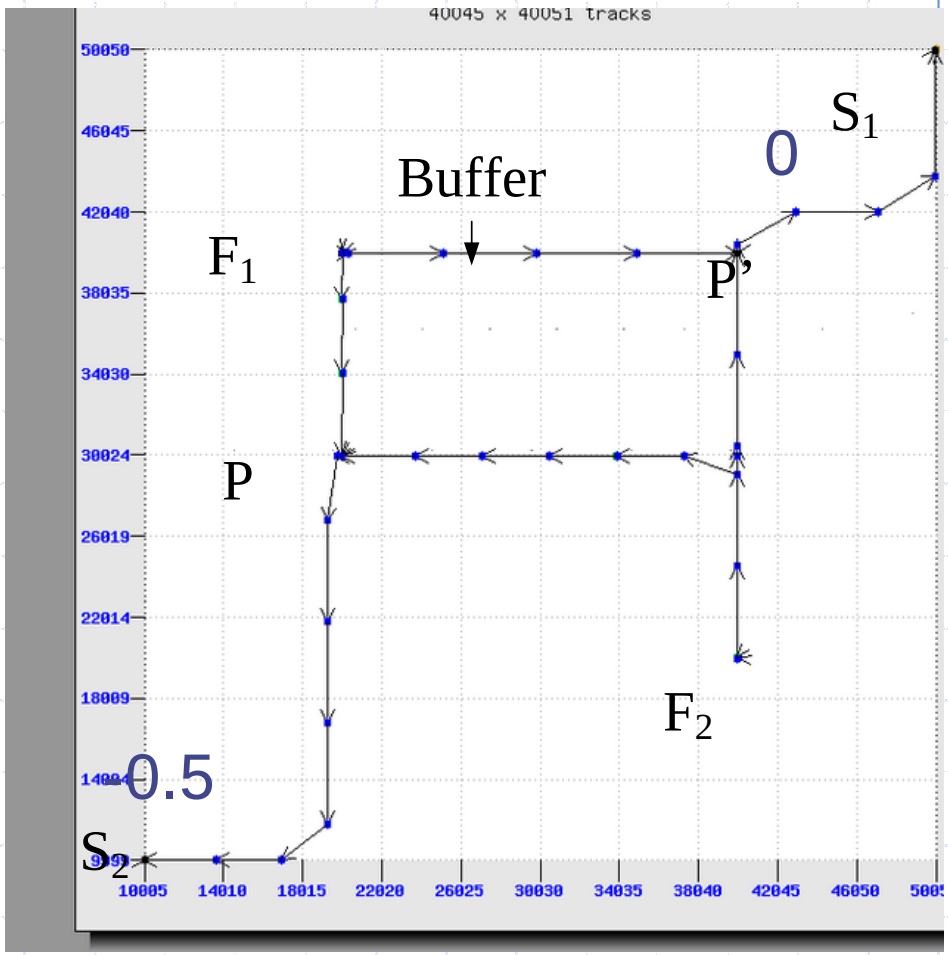


One Example

RUMBLE

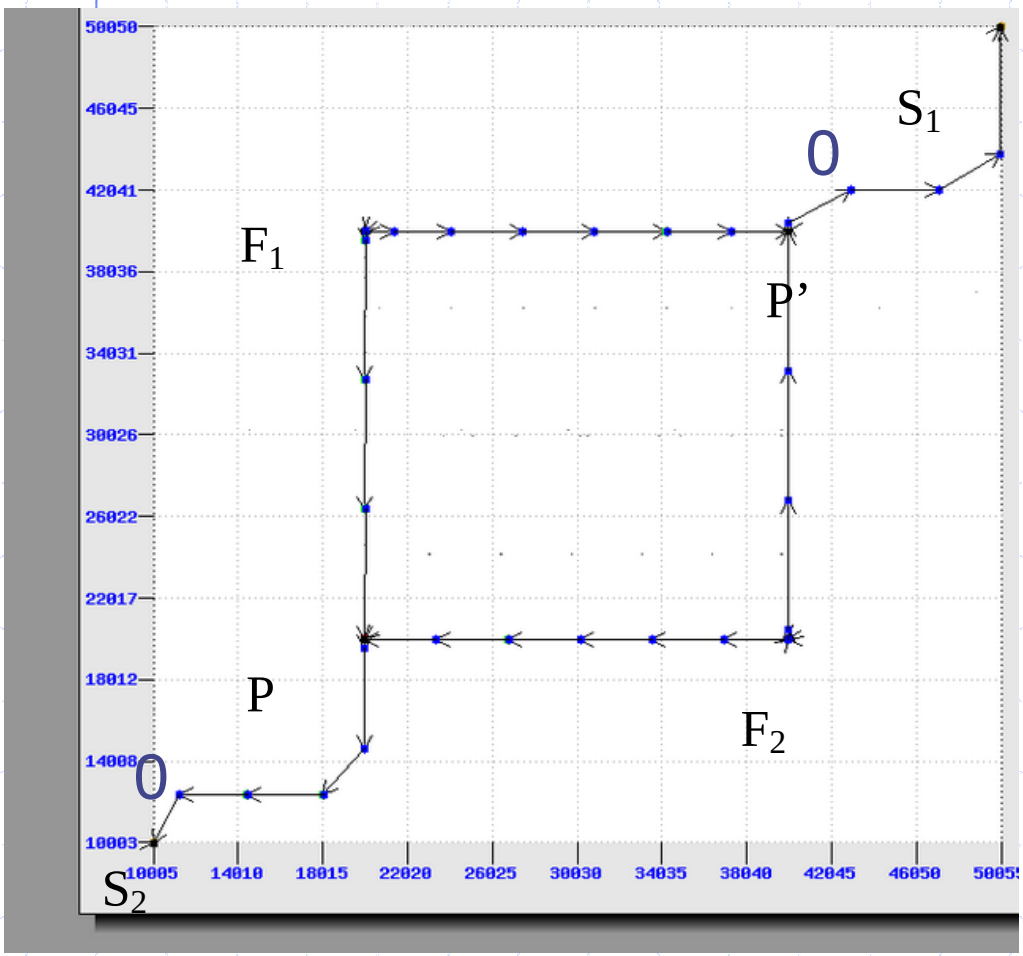


P is fixed

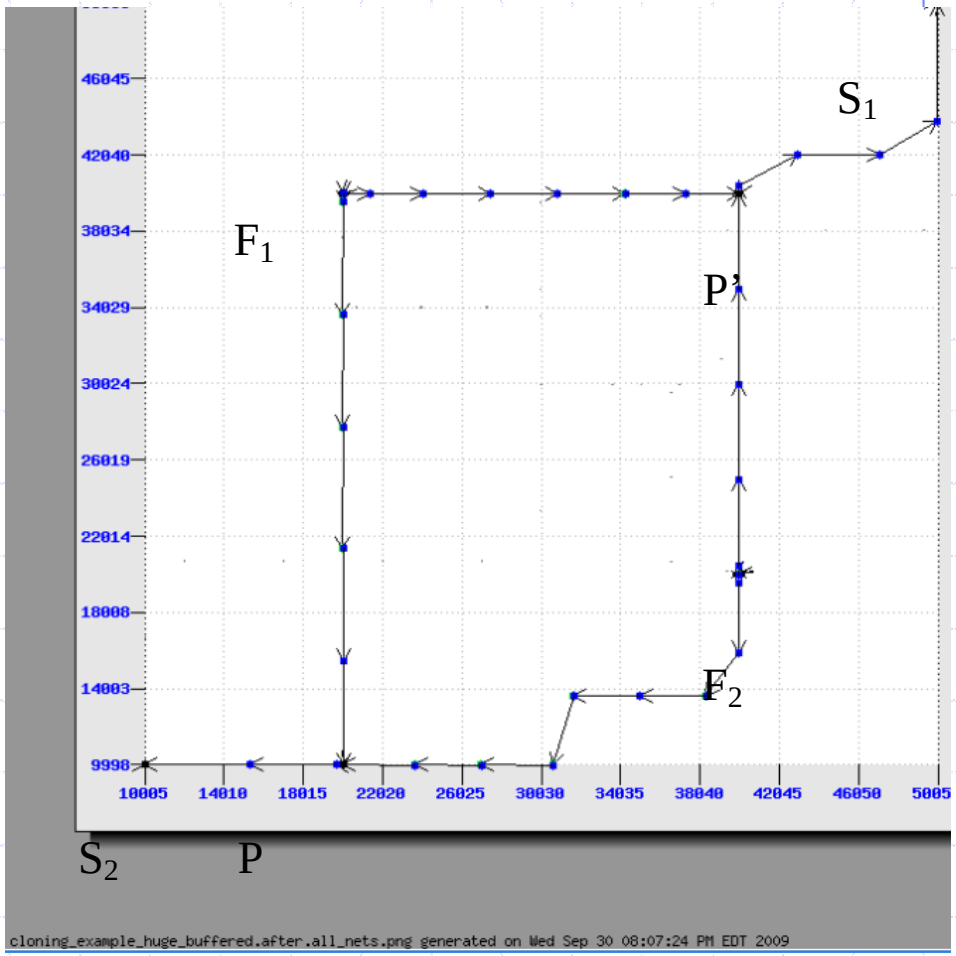


One Example

P is movable



Bad wirelength solution



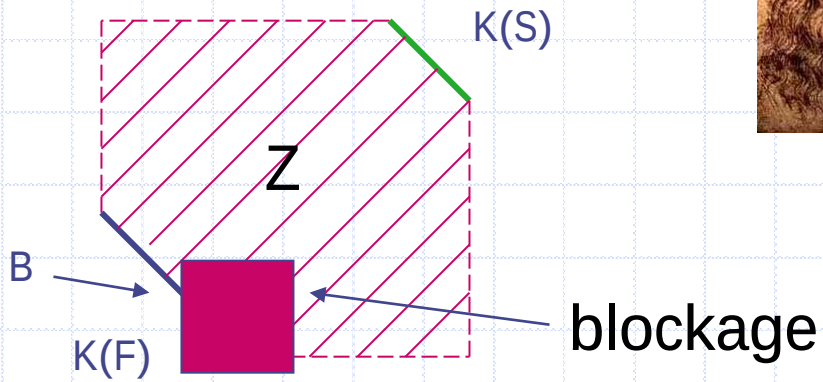
Experimental Results

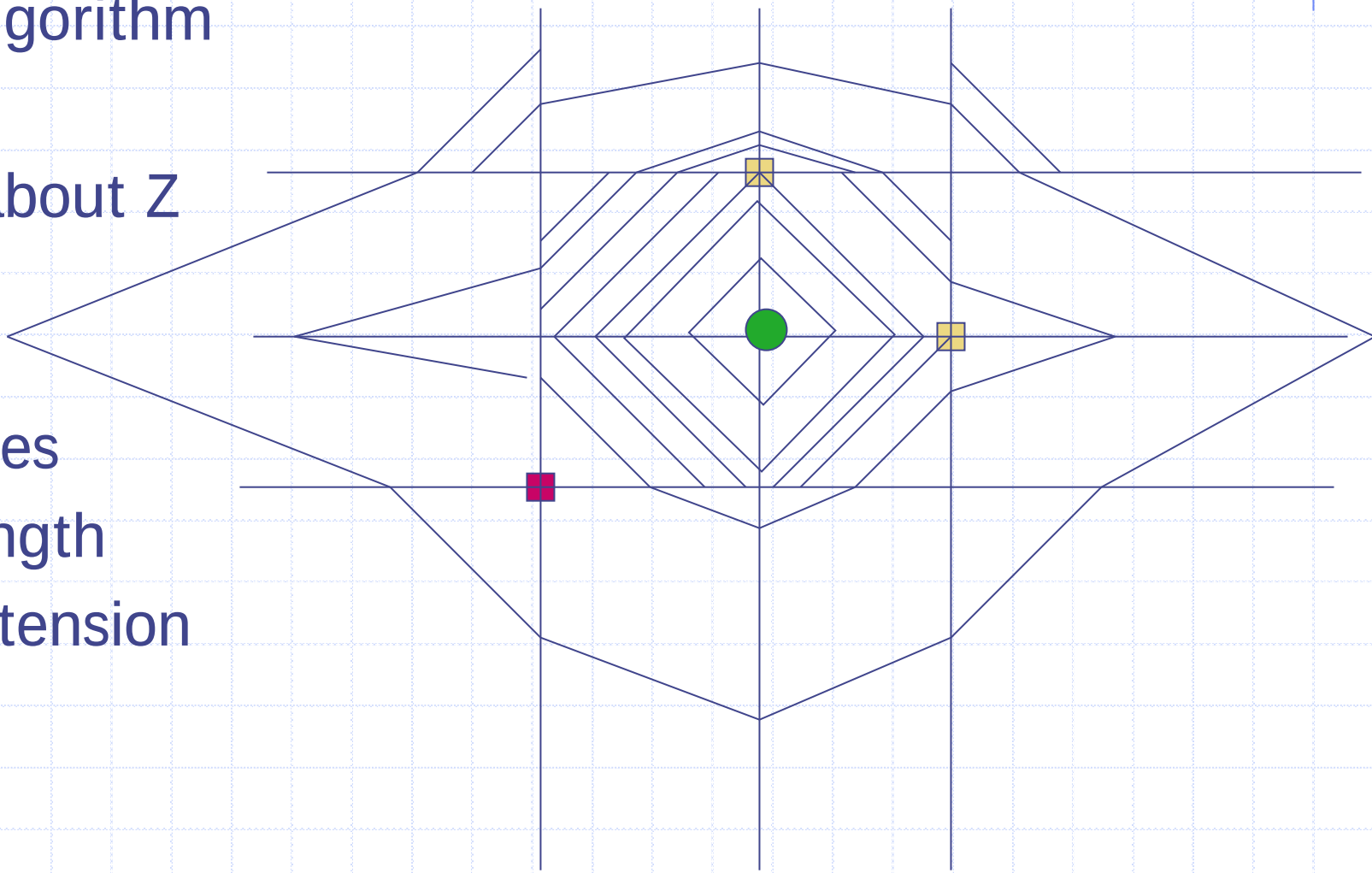
- 100 random 65 nm sub-circuits
  - P is fixed: 279 ps better than pure buffering, 87 ps better than RUBMLE on average
  - P is movable: 309 ps better than pure buffering, 117 ps better than RUMBLE on average

65 nm macros	# objs	Single transform		Compare to a flow with pure buffering		Area Increase
		Slack Imprv.	FOM Imprv.	Slack Imprv.	FOM Imprv.	
Macro 1	91k	0.480 ns	438	0.097 ns	-8	0.5%
Macro 2	231k	0.098 ns	0	0.081 ns	200	0.8%
Macro 3	191k	0.383 ns	2837	0.124 ns	280	1%

Extensions

- Duplicate more than two gates
  - $O(n^2)$ algorithm
- Be smart about Z regions
  - Latches
  - Blockages
  - Wire-length
  - FOM extension





Vironoi Diagram Partitioning

- Best slack for every sink with blockages
- If we know the locations of P and P'
  - The optimal partitioning is the Voronoi diagram between two points or a point and a diamond in Manhattan space
  - Only $O(n^3)$ possible partitionings
  - Try all partitionings and find the best one