Online Learning and Feature Optimization in Lithography Hotspot Detection

Hang Zhang **Bei Yu** Evangeline F. Y. Young

Department of Computer Science and Engineering,
The Chinese University of Hong Kong

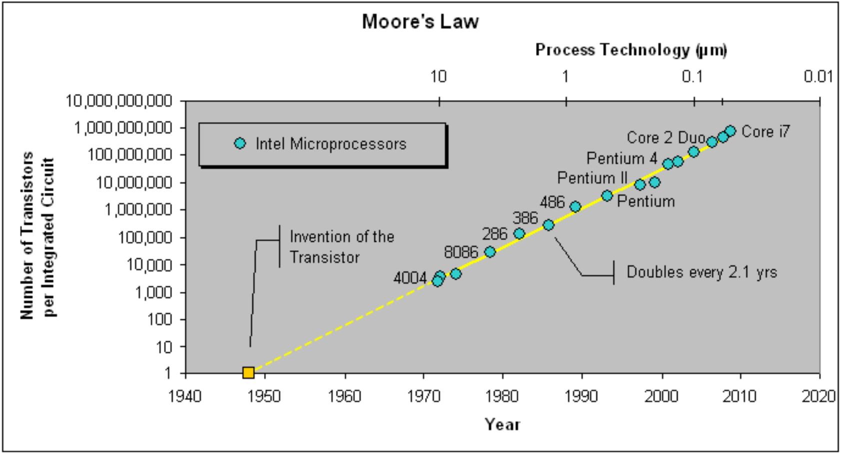
Nov. 8, 2016

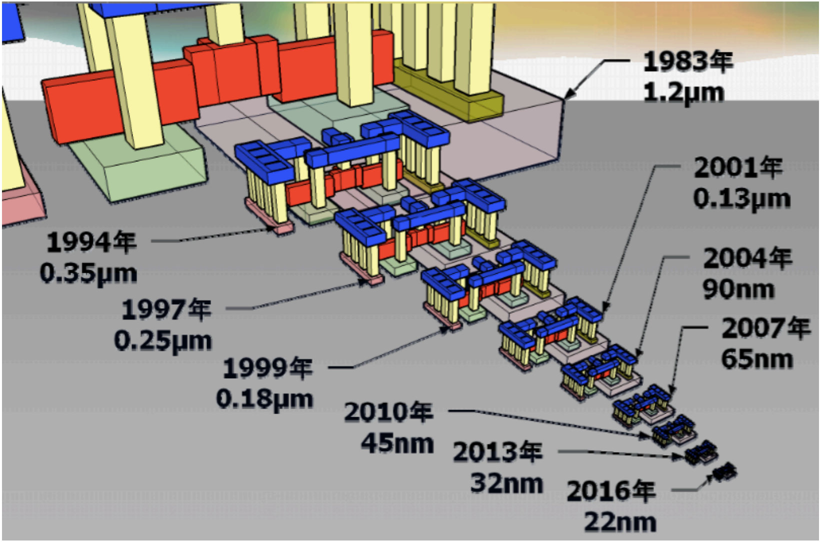
香港中文大學
The Chinese University of Hong Kong

Outline

Outline

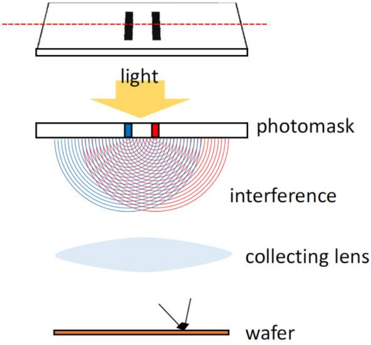
Moore’s Law to Extreme Scaling

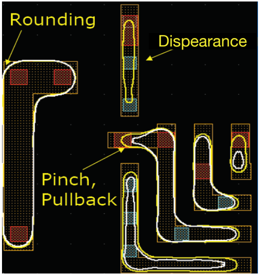



1983年
1.2μm
2001年
0.13μm
2004年
90nm
2007年
65nm
1994年
0.35μm
1997年
0.25μm
1999年
0.18μm
2010年
45nm
2013年
32nm
2016年
22nm

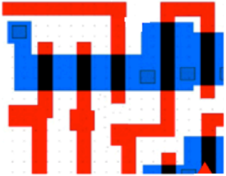
Lithographic Mechanism

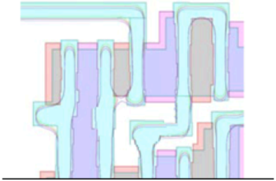
- light diffraction when through photomask
- May cause performance degradation, or even **yield loss**
- What you see $\neq$ what you get

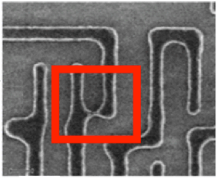




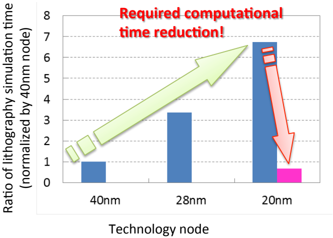
Lithography Hotspot Detection



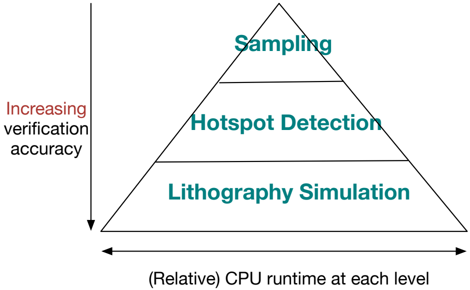


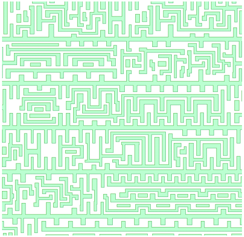


- RET: OPC, SRAF, MPL
- Still hotspot: low fidelity patterns
- Simulations: extremely CPU intensive



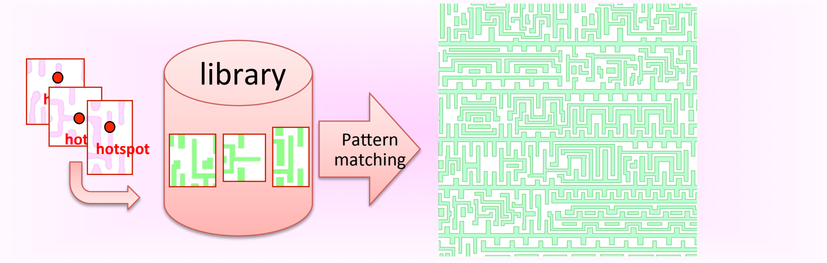
Layout Verification Hierarchy



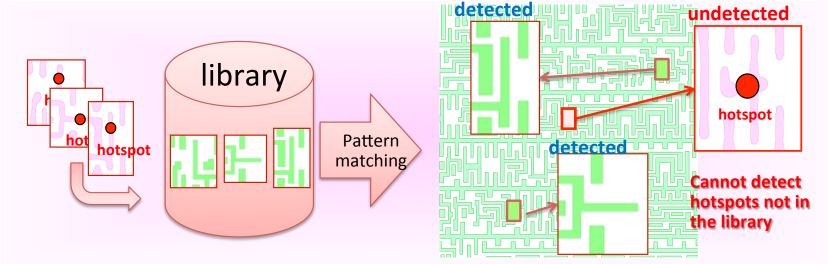


- **Sampling**:
  scan and rule check each region
- **Hotspot Detection**:
  verify the sampled regions and report potential hotspots
- **Lithography Simulation**:
  final verification on the reported hotspots

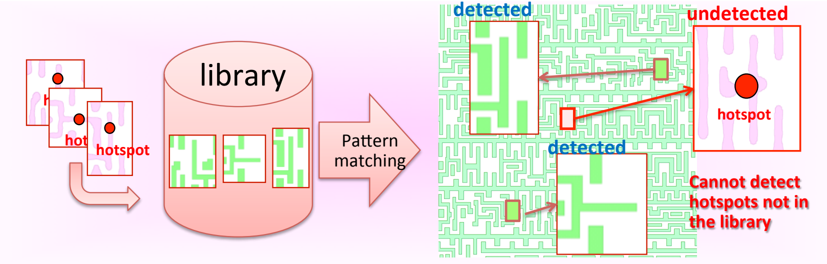
Pattern Matching based Hotspot Detection



Pattern Matching based Hotspot Detection

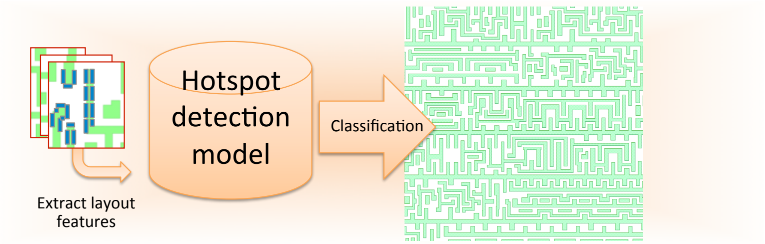


Pattern Matching based Hotspot Detection

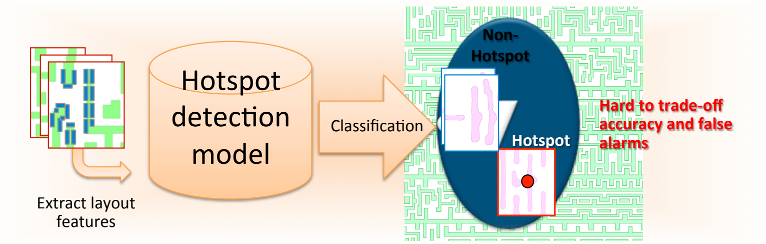


- Fast and accurate
- [Yu+,ICCAD'14] [Nosato+,JM3'14] [Su+,TCAD'15]
- Fuzzy pattern matching [Wen+,TCAD'14]
- Hard to detect non-seen pattern

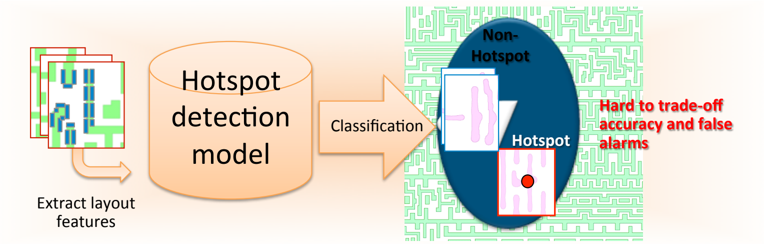
Machine Learning based Hotspot Detection



Machine Learning based Hotspot Detection



Machine Learning based Hotspot Detection



- Predict new patterns
- Decision-tree, ANN, SVM, Boosting ...
- [Drmanac+,DAC'09] [Ding+,TCAD'12] [Yu+,JM3'15] [Matsunawa+,SPIE'15] [Yu+,TCAD'15]
- Hard to balance accuracy and false-alarm

Rethinking Performance Metrics

- **Accuracy**: The rate of correctly predicted hotspots among the set of actual hotspots.
- **False Alarm**: The number of incorrectly predicted non-hotspots.
- **Detection Runtime**: CPU runtime of hotspot detection.

Rethinking Performance Metrics

- **Accuracy**: The rate of correctly predicted hotspots among the set of actual hotspots.
- ~~**False Alarm**: The number of incorrectly predicted non-hotspots.~~
- ~~**Detection Runtime**: CPU runtime of hotspot detection.~~

Rethinking Performance Metrics

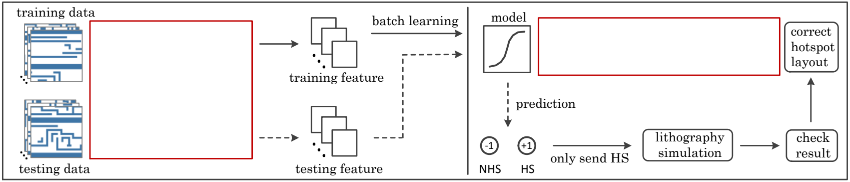
- **Accuracy**: The rate of correctly predicted hotspots among the set of actual hotspots.
- ~~**False Alarm**: The number of incorrectly predicted non-hotspots.~~
- ~~**Detection Runtime**: CPU runtime of hotspot detection.~~

Overall Detection and Simulation Time (ODST)

Includes: (1) Detection runtime; (2) Lithography simulation time for hotspots in testing.

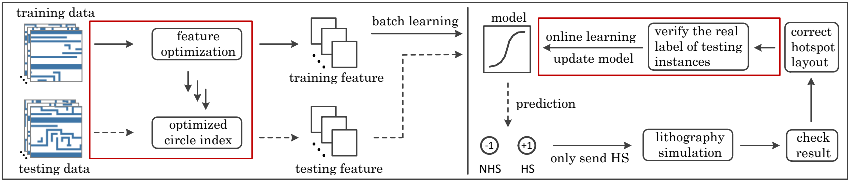
*Transfer false alarm into equivelent lithography simulation time.

Rethinking Hotspot Detection Framework



- Conventional framework: supervised learning.
- Two stages: training and testing.
- Testing hotspot (HS) is verified by litho simulator.

Rethinking Hotspot Detection Framework



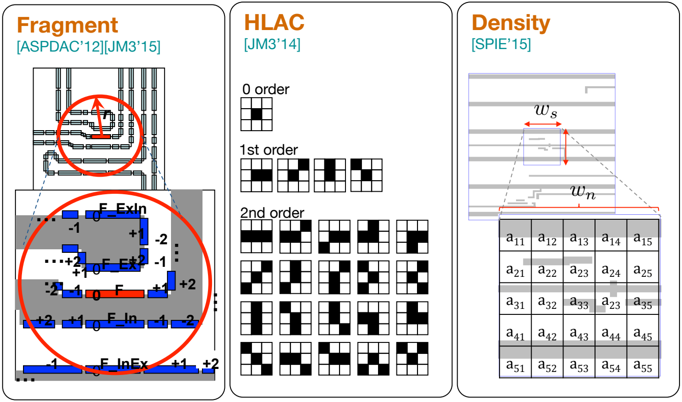
- Conventional framework: supervised learning.
- Two stages: training and testing.
- Testing hotspot (HS) is verified by litho simulator.

Proposed New Framework:

- Feature optimization.
- Online model learning.

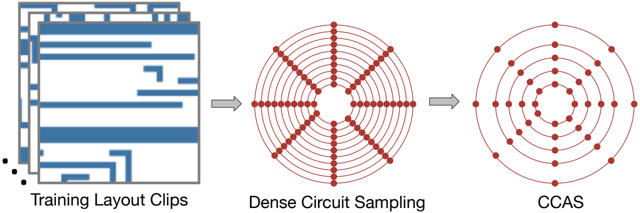
Outline

Conventional Feature Extraction



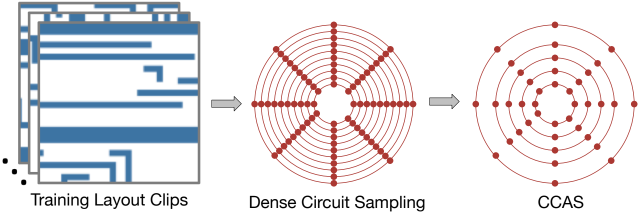
- Hard to be adaptive to different layout designs
- Too many parameters to tune
- Sometimes very complex and may cause over fitting

Rethinking CCAS



- Concentric Circle Area Sampling (CCAS) [Matsunawa+,JM3'16].
- Capture the affects of light diffraction.
- Simple rule to select circles from dense samples.

Rethinking CCAS



- Concentric Circle Area Sampling (CCAS) [Matsunawa+,JM3'16].
- Capture the affects of light diffraction.
- Simple rule to select circles from dense samples.

Question:

Can we find **correlation** between circles and hotspots, and select circles samrtly?

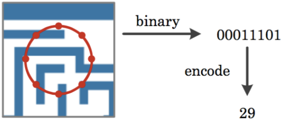
Rethinking CCAS

Measure correlation between circle and the hotspot.

Mutual Information

$$I(C_i; Y) = \sum_{c_i \in C} \sum_{y \in Y} p(c_i, y) \log \frac{p(c_i, y)}{p(c_i)p(y)}$$

- c_i: one encoded decimal number in circle $\mathcal{C}$
- $p(c_i)$: probability of c_i
- y: each classification label
- $p(y)$: probability of y



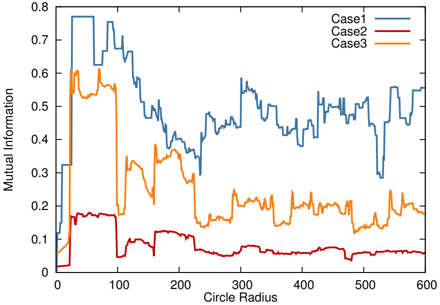
Decimal number encoding

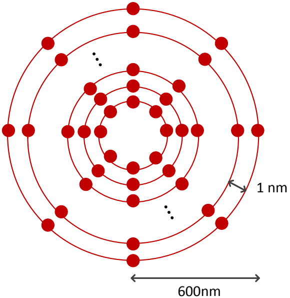
Smart CCAS Circle Selection

Higer Mutual Information

More correlation between circle and label variable.

- Mutual information curve can be drawn based on training data
- We donot want to sample circles too dense

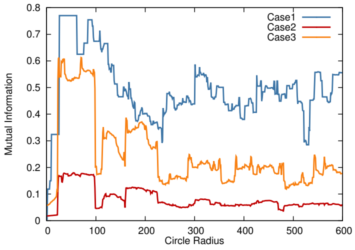




Smart CCAS Circle Selection

Mathematical Formulation

$$
\begin{aligned}
\max \quad & \mathbf{v}^\mathsf{T}\mathbf{w} \\
\text{s.t.} \quad & v_i = I(C_i;Y), && \forall v_i \in \mathbf{v}, \\
& ||w_i||_0 = n_c, && \forall\, i,\ w_i \in \{0,1\}, \\
& |i-j| \geq d, && \forall\, i \neq j,\ w_i = w_j = 1
\end{aligned}
$$



Smart CCAS Circle Selection

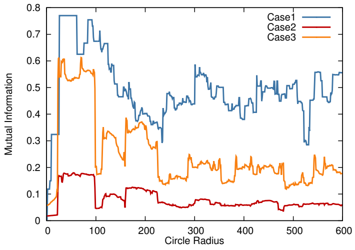
Mathematical Formulation

$$\max \quad \mathbf{v}^\mathsf{T}\mathbf{w}$$

$$\text{s.t.} \quad v_i = I(C_i;Y), \quad \forall v_i \in \mathbf{v},$$

$$||w_i||_0 = n_c, \quad \forall\, i,\ w_i \in \{0,1\},$$

$$|i-j| \geq d, \quad \forall\, i \neq j,\ w_i = w_j = 1$$

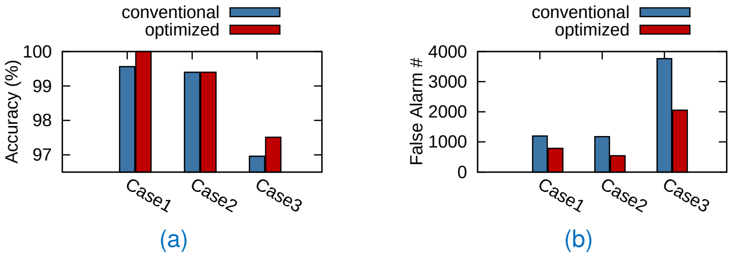


Optimally Solved by Dynamic Programming

$$D[i,j] = \max\{v[i] + D[i-d, j-1], D[i-1,j]\}\}$$

Performance of Feature Optimization

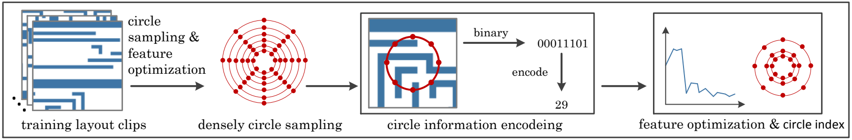
- Smart Circle Selection v.s. Conventional CCAS.



(a) The impact on accuracy; (b) The impact on false alarm.

Review of the Feature Optimization Framework

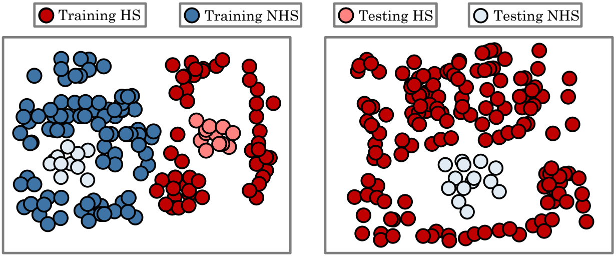
- Firstly, we densely sample the circles from the training data.
- Secondly, we optimally select circles by DP algorithm.
- Thirdly, we use the obtained circle index to extract features.



Outline

Motivation of Online Hotspot Detection

- In (a), all testing hotspots and non-hotspots can be correctly detected.
- But in (b), all testing non-hotspots become false alarms.



(a) (b)

Algorithm Flow of Smboost

Smooth Boosting [Servedio, JMLR'03]

Require: $\{(x_1, y_1), \ldots, (x_m, y_m)\}, \gamma, \theta = \frac{\gamma}{2+\gamma}, T.$

for $i \leftarrow 1$ to n **do**
  $M_1(i) \leftarrow 1;$
  $N_0(i) \leftarrow 0;$
end for
for $t \leftarrow 1$ to T **do**
  Run weak classifier to get h_t such that $\frac{1}{2}\sum_{j=1}^{n} M_t(j)|h_t(x_j) - y_j| \leq \frac{1}{2} - \gamma;$
  for $j \leftarrow 1$ to n **do**
    $N_t(j) \leftarrow N_{t-1}(j) + y_j h_t(x_j) - \theta;$
  end for
  for $j \leftarrow 1$ to n **do**
    $M_{t+1}(j) \leftarrow \min\{1.0, (1-\gamma)^{\frac{N_t(j)}{2}}\};$
  end for
end for
return $f \leftarrow \text{sign}(\frac{1}{T}\sum_{t=1}^{T} h_t);$

Algorithm Flow of Online Smboost

- Extend conventional smboost to the online scenario.

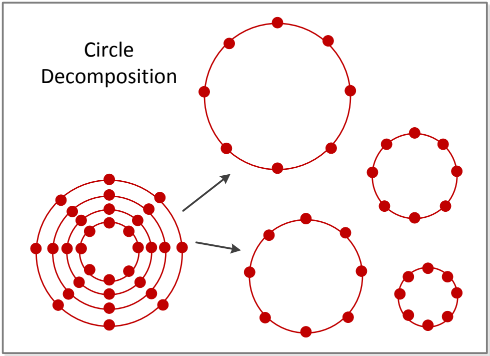
Online Smooth Boosting

Require: Streaming instance (x, y), batch smboost classifier.

1: $M_1 \leftarrow 1, N_0 \leftarrow 0$;
2: **for** $t \leftarrow 1$ to T **do**
3: online update $h_t(x, y)$;
4: $N_t \leftarrow N_{t-1} + yh_t(x) - \theta$;
5: $M_{t+1} \leftarrow \min\{1.0, (1-\gamma)^{\frac{N_t}{2}}\}$;
6: **end for**
7: **return** $f \leftarrow$sign$(\frac{1}{T}\sum_{t=1}^{T} h_t)$;

Online Weak Classifier

- Use Naive Bayes (NB) as weak classifier [Chen+,ICML'12].
- NB is a lossless [Oza, ICSMC'05] online weak classifier.
- Modify NB to work better with our proposed feature.



Outline

Comparison with [Matsunawa+,SPIE'15]

- Verified in ICCAD-2012 contest benchmark
- 4x speed-up due to the simple feature.
- Increase detection accuracy from 95.13% to 97.95%.

	[Matsunawa+,SPIE'15]			batch		
	FA#	CPU(s)	Accuracy	FA#	CPU(s)	Accuracy
Case1	**0**	**7**	**100.00%**	**0**	**7**	**100.00%**
Case2	**0**	351	98.60%	**0**	**53**	**99.40%**
Case3	**0**	297	97.20%	3	**66**	**97.51%**
Case4	1	170	87.01%	**0**	**49**	**97.74%**
Case5	**0**	69	92.86%	**0**	**27**	**95.12%**
avg.	**0.2**	178.8	95.13%	0.6	**40.4**	**97.95%**
ratio	-	4.43	0.97	-	**1.0**	**1.0%**

Comparisons with [Wen+,TCAD'14] [Yu+,TCAD'15]

- *ODST = Overall Detection and Simulation Time*
- Increase detection accuracy by at least 3.47% on average.
- Improve the performance of ODST by at least 58.80% on average.

	[Wen+,TCAD'14]		[Yu+,TCAD'15]		batch	
	ODST(s)	Accuracy	ODST(s)	Accuracy	ODST(s)	Accuracy
Case1	17151	100.00%	14968	94.69%	**7890**	**100.00%**
Case2	40867	**99.80%**	118574	98.20%	**5572**	99.40%
Case3	95277	93.80%	139278	91.88%	**20660**	**97.51%**
Case4	**11302**	91.00%	36996	85.94%	33526	**97.74%**
Case5	2039	87.80%	12070	92.86%	**1005**	**95.12%**
avg.	33327.2	94.48%	64377.2	92.71%	**13730.6**	**97.95%**
ratio	2.43	0.96	4.69	0.95	**1.0**	**1.0**

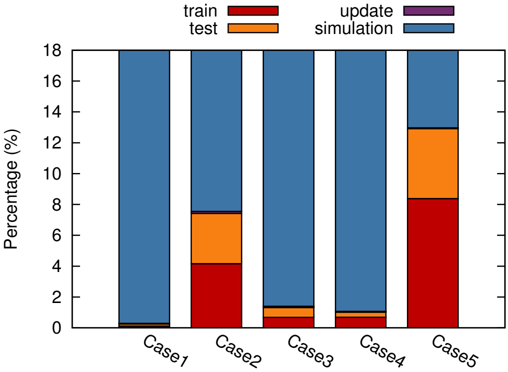
Batch Learning v.s. Online Learning

- Further improve the detection accuracy from 97.95% to 98.45%.
- Further reduce ODST by 26.1%.

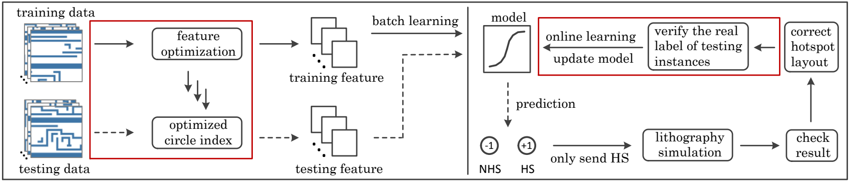
	batch				online			
	FA#	CPU(s)	ODST(s)	Acccuracy	FA#	CPU(s)	ODST(s)	Accuracy
Case1	788	10	7890	**100.00%**	**704**	13	**7050**	**100.00%**
Case2	544	132	5572	**99.40%**	**308**	152	**3251**	**99.40%**
Case3	2052	140	20660	97.51%	**1819**	180	**18379**	**97.57%**
Case4	3341	116	33526	**97.74%**	**2096**	158	**21148**	**97.74%**
Case5	94	76	1005	95.12%	**82**	78	**910**	**97.56%**
avg.	1363.8	94.8	13730.6	97.95%	**1008.8**	116.4	**10147.6**	**98.45%**
ratio	-	-	1.35	0.99	-	-	**1.0**	**1.0**

Runtime Breakdown for ICCAD Benchmark

- Online updating is only a small portion of the whole detection flow.
- False alarms of Case 2 and Case 5 are dramatically reduced.



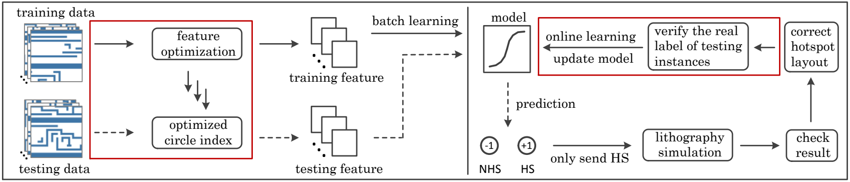
Conclusion



A New Hotspot Detection Framework

- New performance metric: runtime & performance trade-off
- Feature optimization based on mutual information
- Online learning

Conclusion



A New Hotspot Detection Framework

- New performance metric: runtime & performance trade-off
- Feature optimization based on mutual information
- Online learning

Future work

- Further improve the accuracy
- Hardware or parallel speedup of hotspot detector

Thank You

Hang Zhang (byu@cse.cuhk.edu.hk)

Bei Yu (byu@cse.cuhk.edu.hk)

Evangeline F. Y. Young (fyyoung@cse.cuhk.edu.hk)

香港中文大學
The Chinese University of Hong Kong