Narrowing Down the Debugging Space of Slow Search Response Time

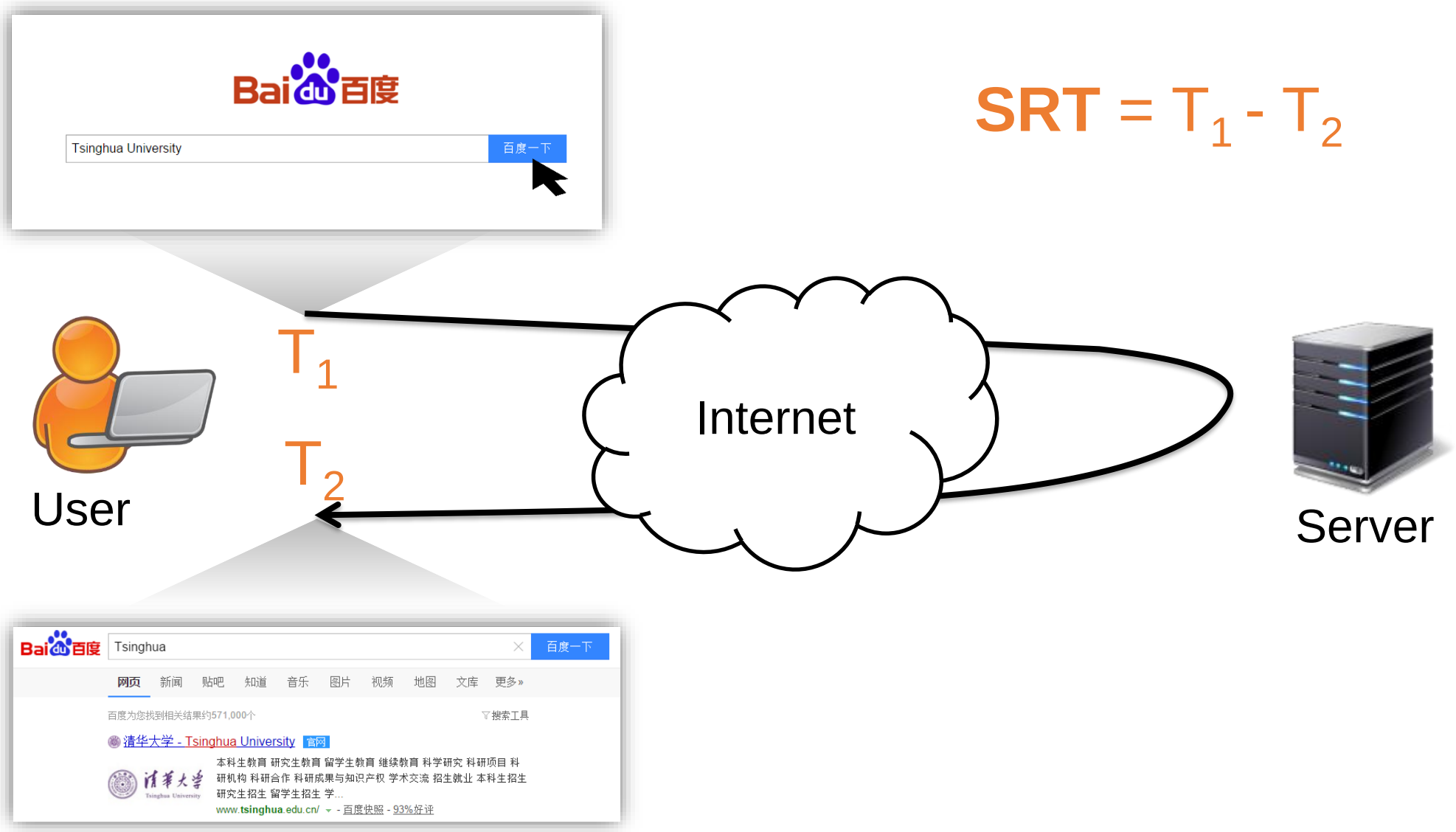
Dapeng Liu, Youjian Zhao, Dan Pei, Chengbin Quan,
Qingqian Tao, Pei Wang, Xiyang Chen, Dai Tan, Xiaowei Jing, Mei Feng

Tsinghua

Baidu 百度

PetroChina

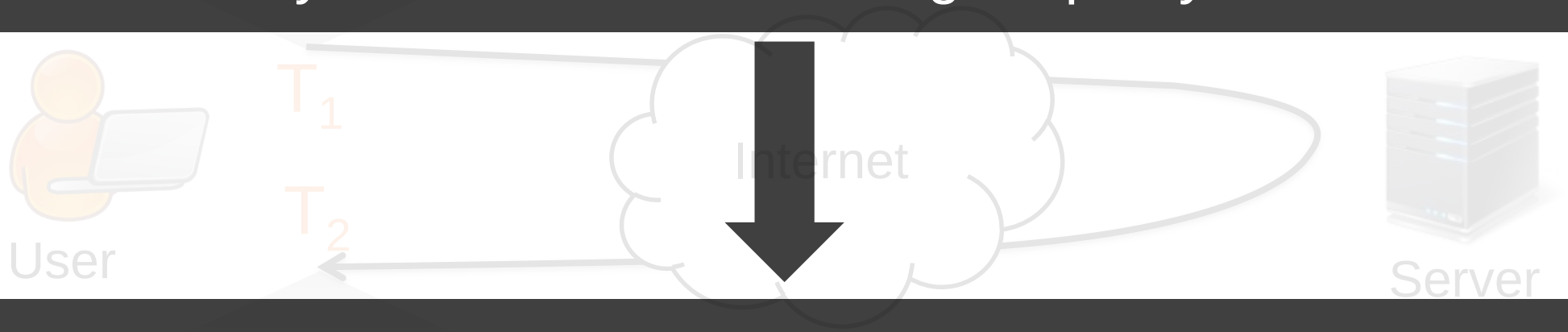
Search Response Time (SRT)



Search Response Time (SRT)

SRT matters!

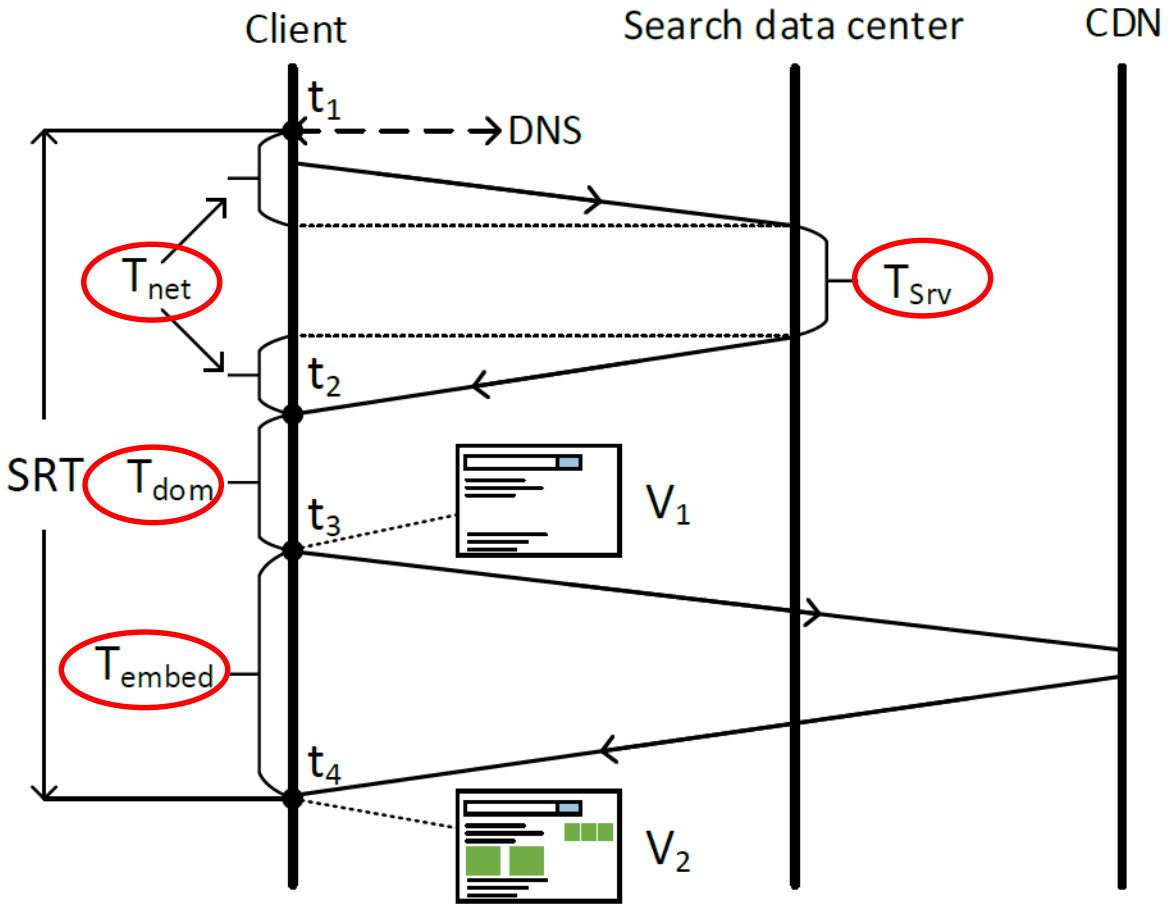
- Users are more likely to perform click on pages with short SRT[SIGIR'14]
- SRT increases by 1s, the revenue of Bing drops by 2.8%[Velocity'09]



We need to understand Slow SRT

Monitoring SRT

- Logging SRT components



Monitoring SRT

- Logging potential impact factors

Browser Types

# of embedded images

Containing Ads or not

Network Status

Computing power of user devices

Monitoring SRT

- SRT logs

SRT (ms)	SRT components				Impact factors				
	T_{net}	T_{srv}	T_{dom}	T_{embed}	#Image	Browser	Ad	Computing power	Network status
3000									
2500									
400									
200									
…									

Slow SRT

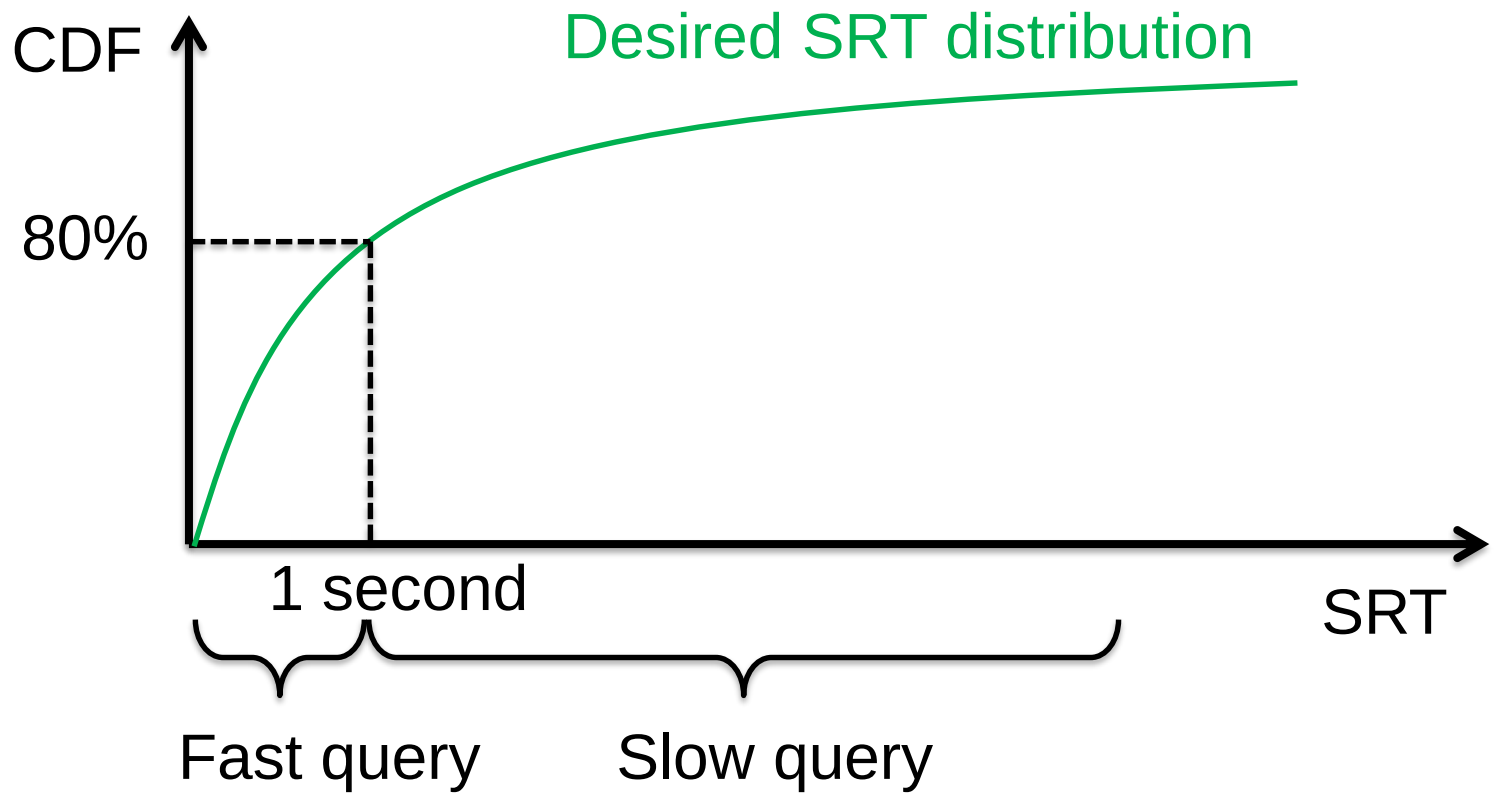
- **1s** can potentially interrupt users' flow of thought
  - http://www.nngroup.com/articles/response-times-3-important-limits/
- Search engines like Google and Baidu try to avoid this

SRT (ms)	SRT components				Impact factors				
	T_{net}	T_{srv}	T_{dom}	T_{embed}	#Image	Browser	Ad	Computing power	Network status
3000									
2500									
400									
200									
…									

Slow SRT: > 1000ms

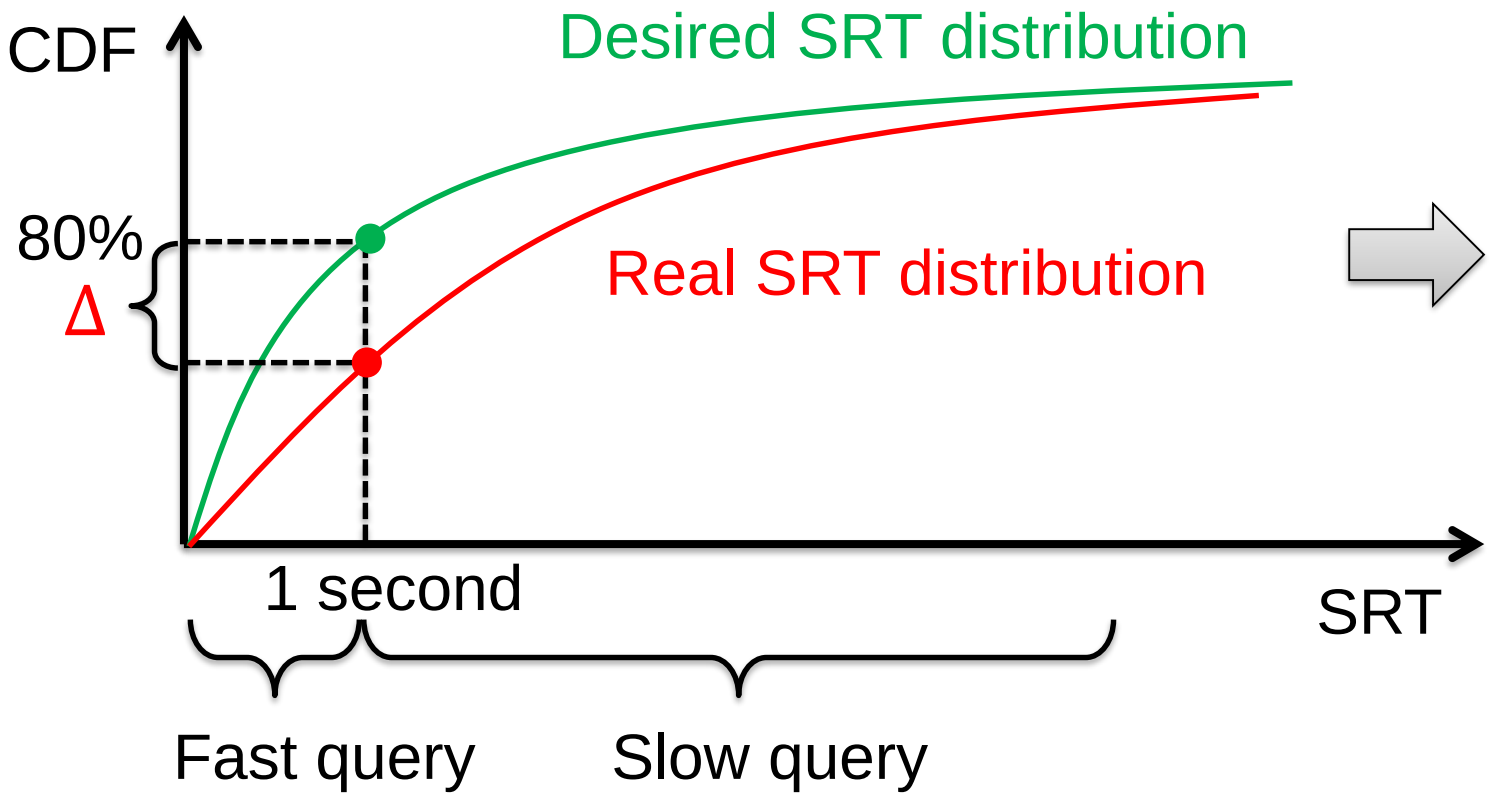
SRT Requirement

- Search engine often uses some **percentile** to measure SRT
  - 80th percentile SRT <1s → 80% of queries are served in <1s



SRT Requirement

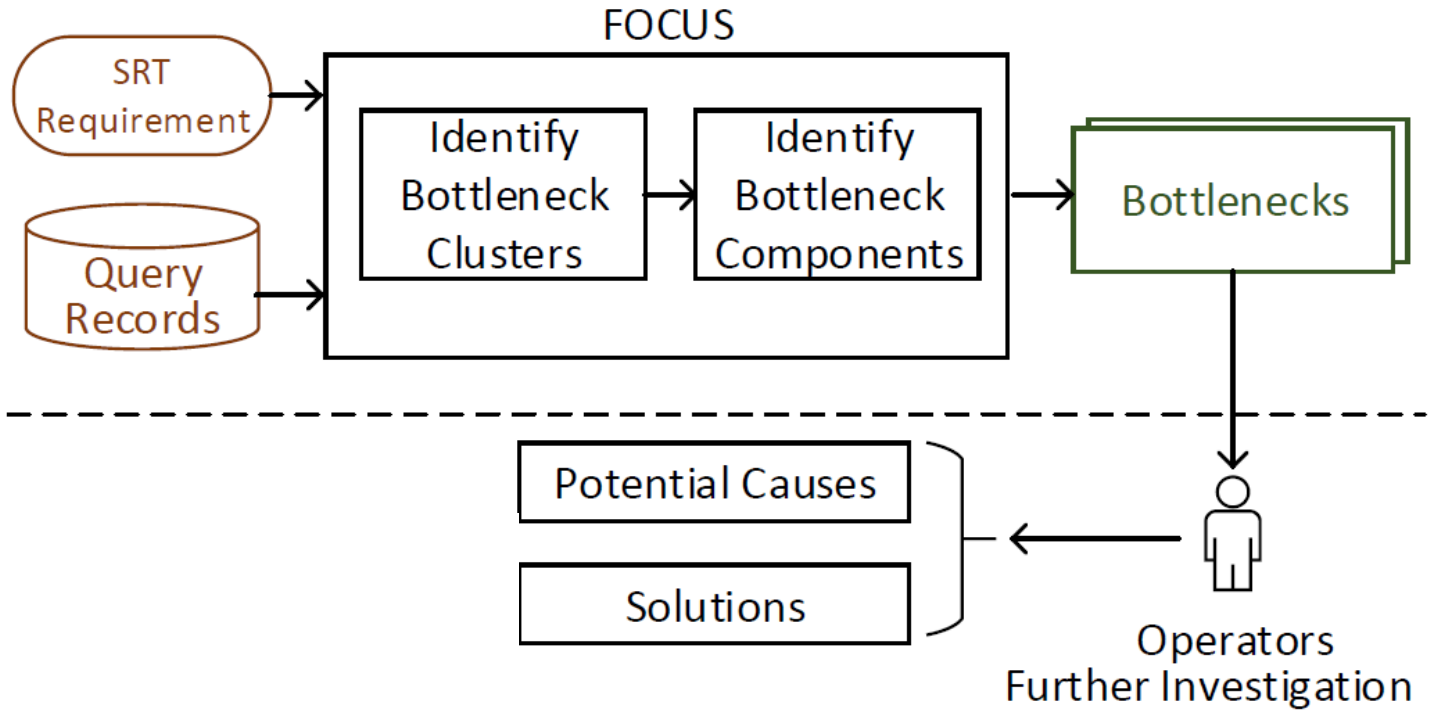
- **1s** can potentially interrupt users' flow of thought
- Search engines like Google and Baidu try to avoid this



Δ slow queries need to be improved to fast ones

Key Idea

- We first need to find what conditions contain at least Δ slow queries
  - **Bottleneck clusters**
- In each cluster, we want to explain the slow SRT with SRT components
  - **Bottleneck components**



Identifying Bottleneck Clusters

- Cluster: a combination of impact factors

Please read the paper for how to categorize continuous values, e.g., #Image

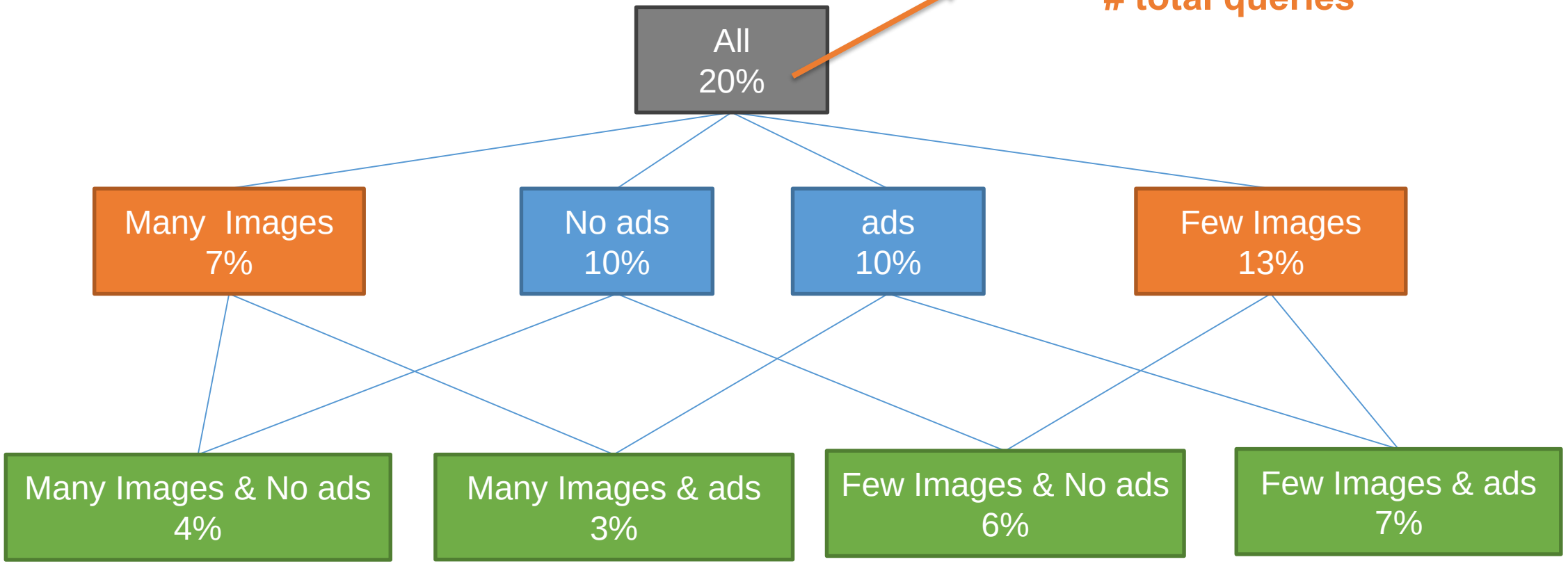
	#Image	Browser	Ad	Computing power	Network status
Cluster #1	Many	*	Yes	*	*
Cluster #2	*	*	*	*	Fast
…					

- **They can overlap with each other**
- **These clusters are organized into a multi-dimensional hierarchy**

Identifying Bottleneck Clusters

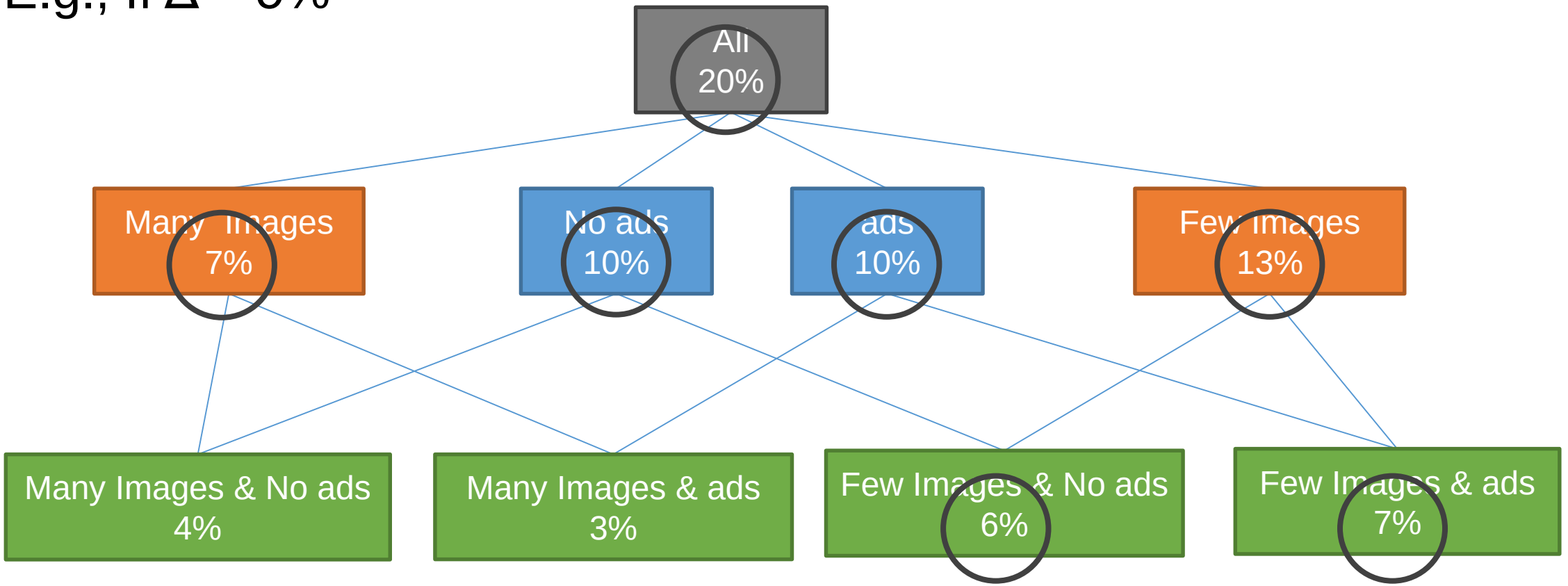
- Take two factors as an example

 # Slow queries in the cluster / # total queries



Identifying Bottleneck Clusters

- Naïve bottleneck clusters are too **redundant** or **general**
- E.g., If Δ = 5%

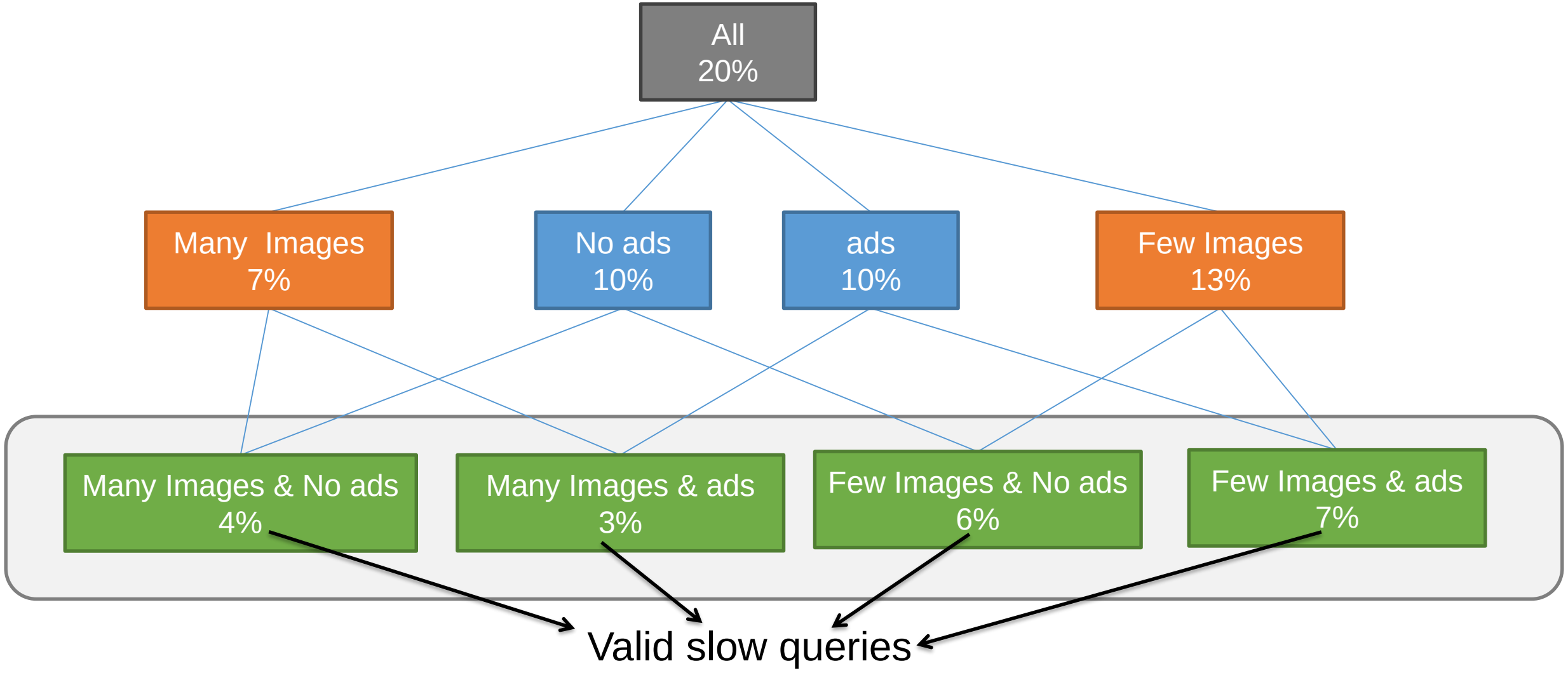


Identifying Bottleneck Clusters

- Idea:
  - Attributing slow queries to **low-level clusters** as much as possible since they are more specific
  - **Valid slow queries**: slow queries that haven't been counted in low-level clusters
  - **Bottleneck cluster: ~~slow queries > Δ~~ valid slow queries > Δ**

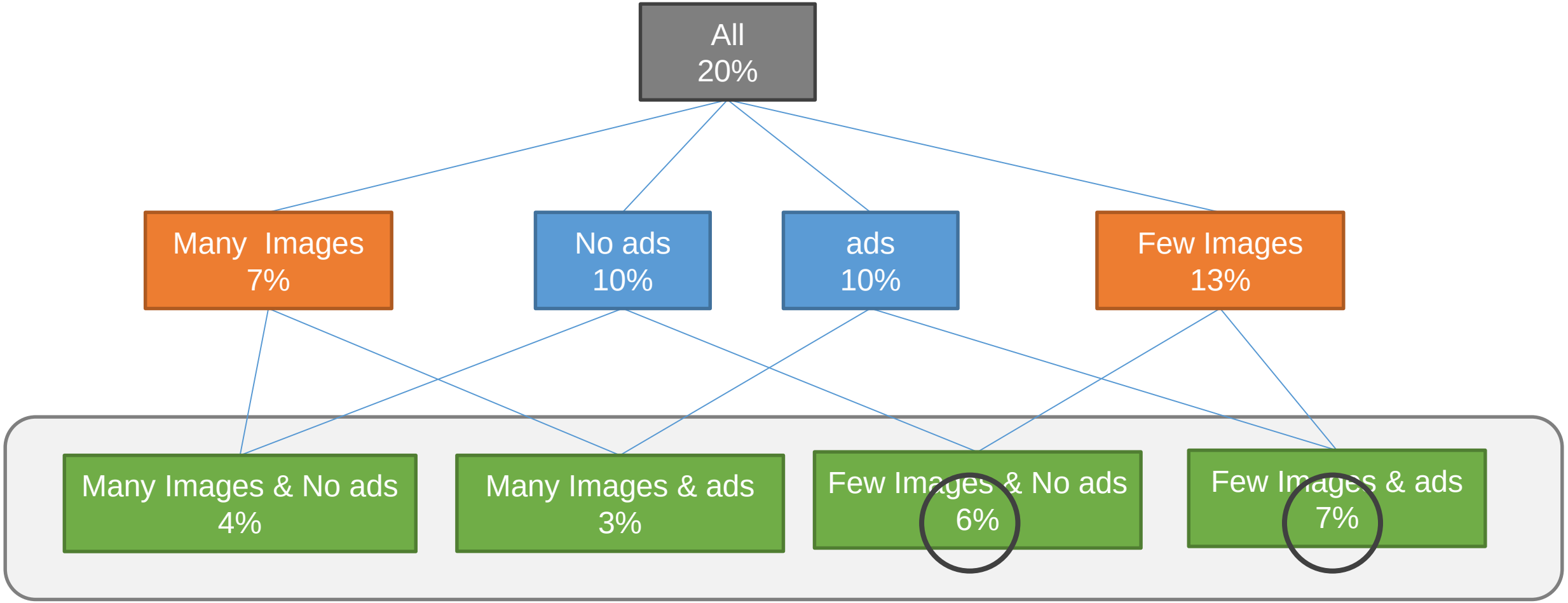
Identifying Bottleneck Clusters

- Results (Δ = 5%)



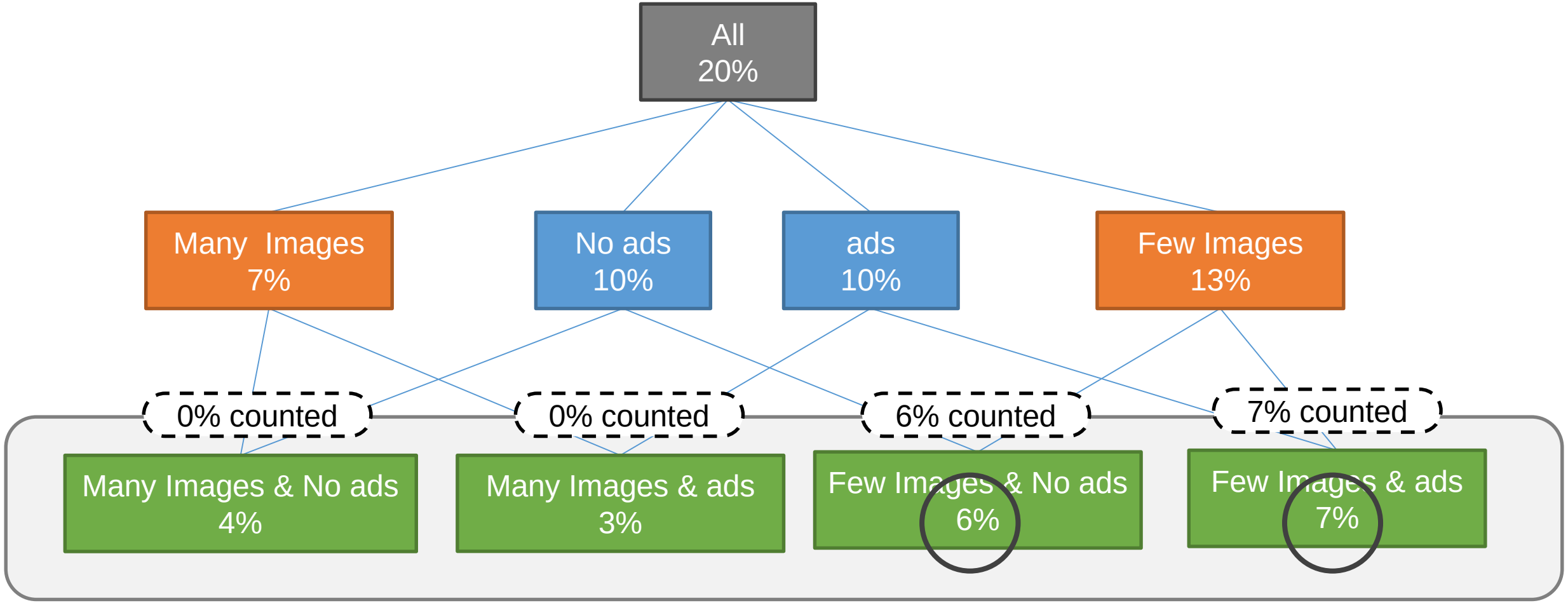
Identifying Bottleneck Clusters

- Results (Δ = 5%)



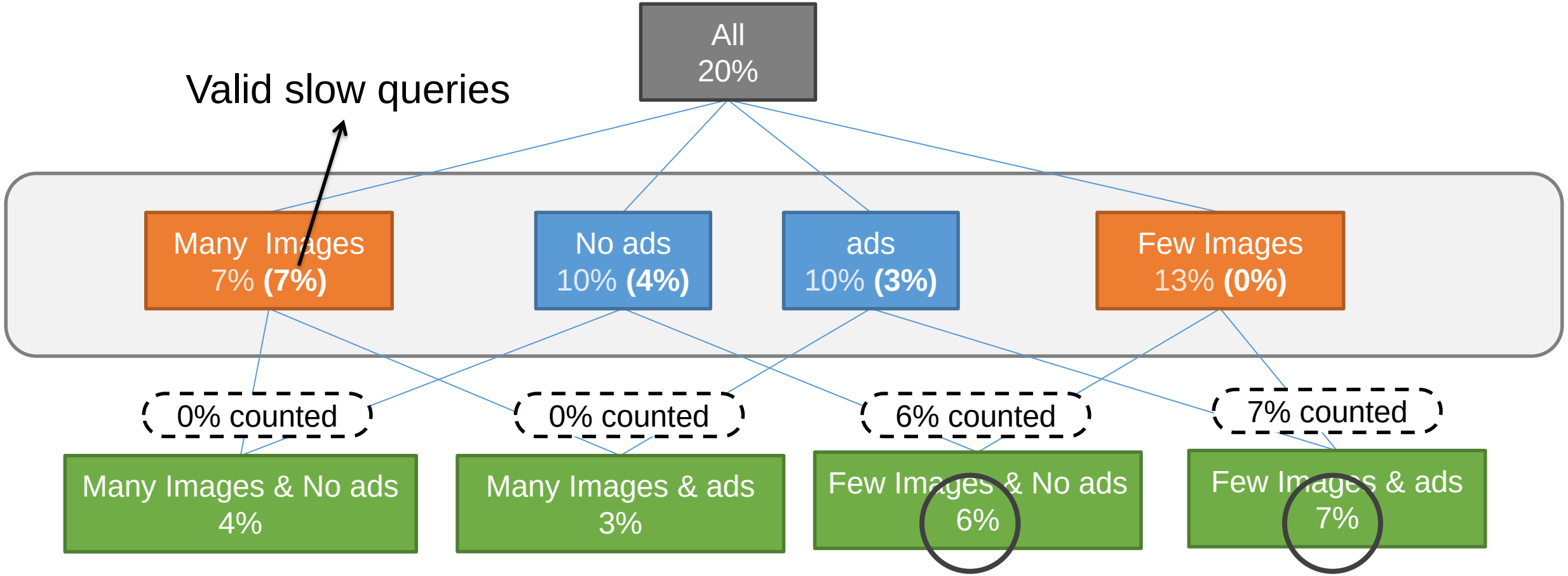
Identifying Bottleneck Clusters

- Results (Δ = 5%)



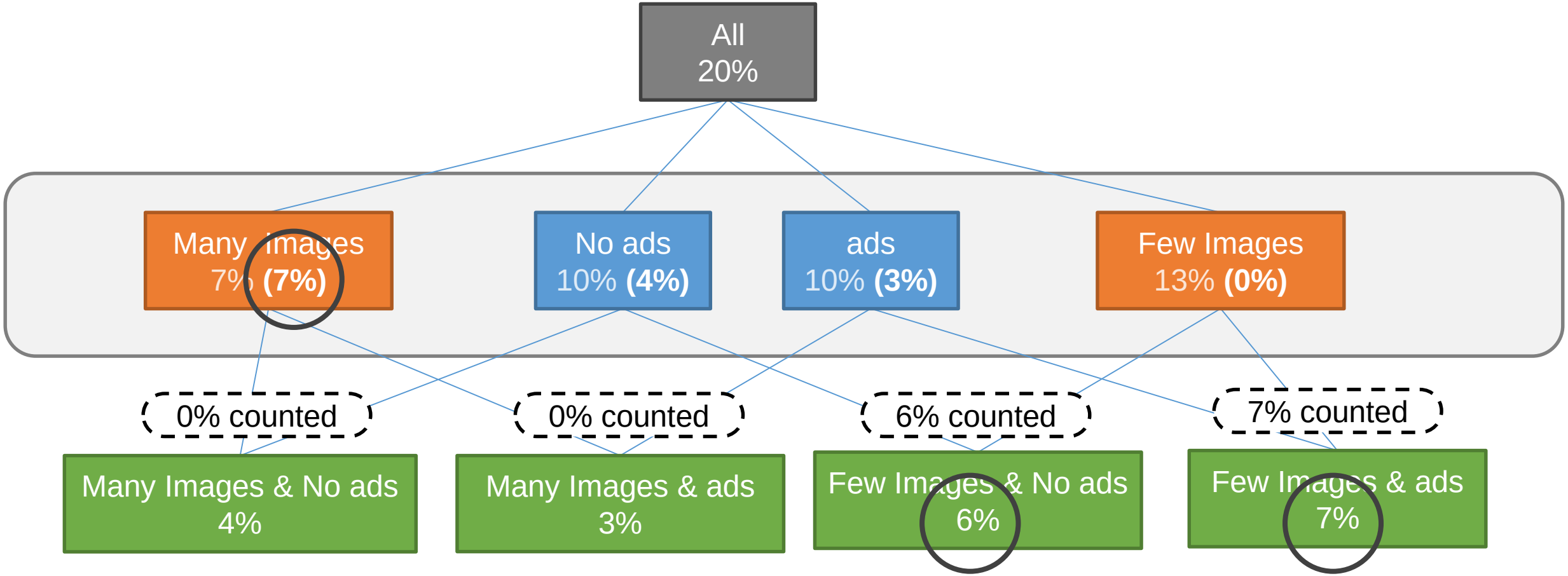
Identifying Bottleneck Clusters

- Results (Δ = 5%)



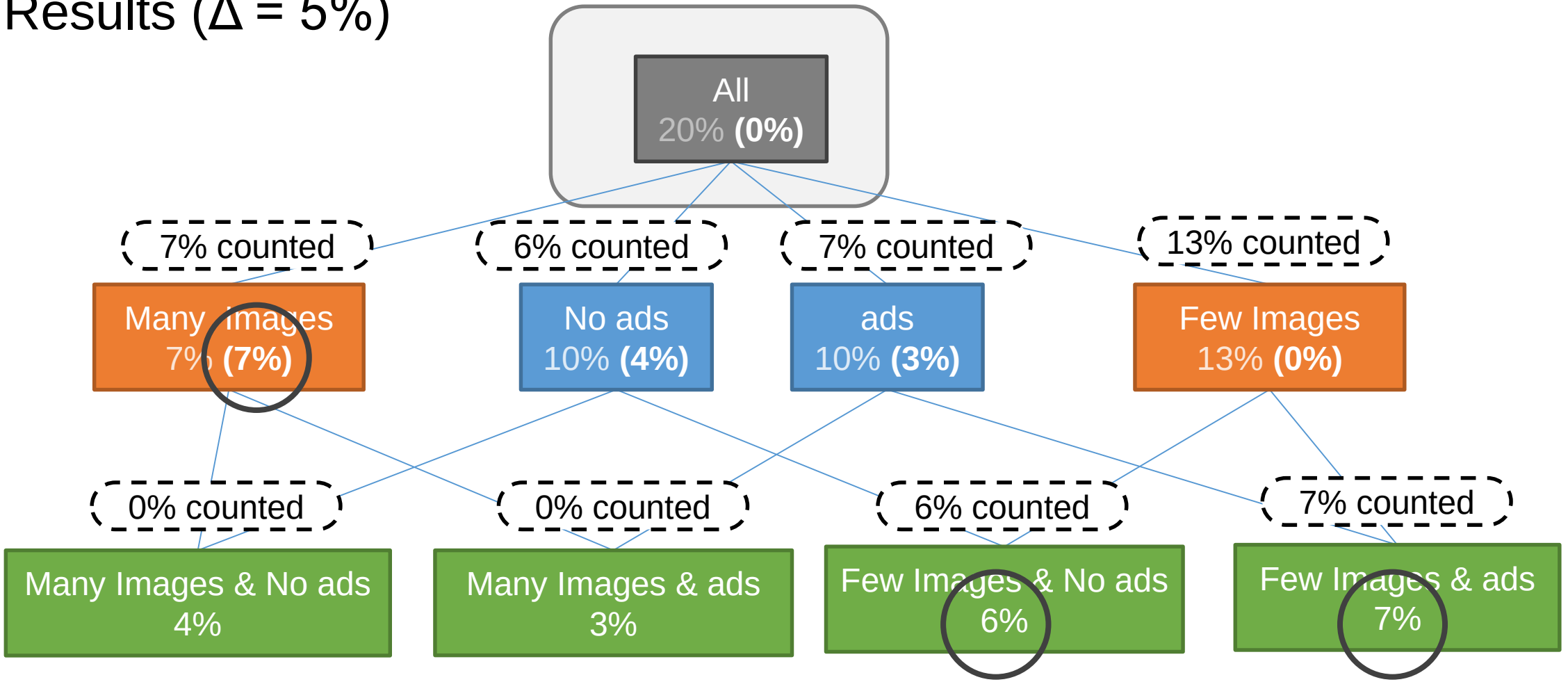
Identifying Bottleneck Clusters

- Results (Δ = 5%)



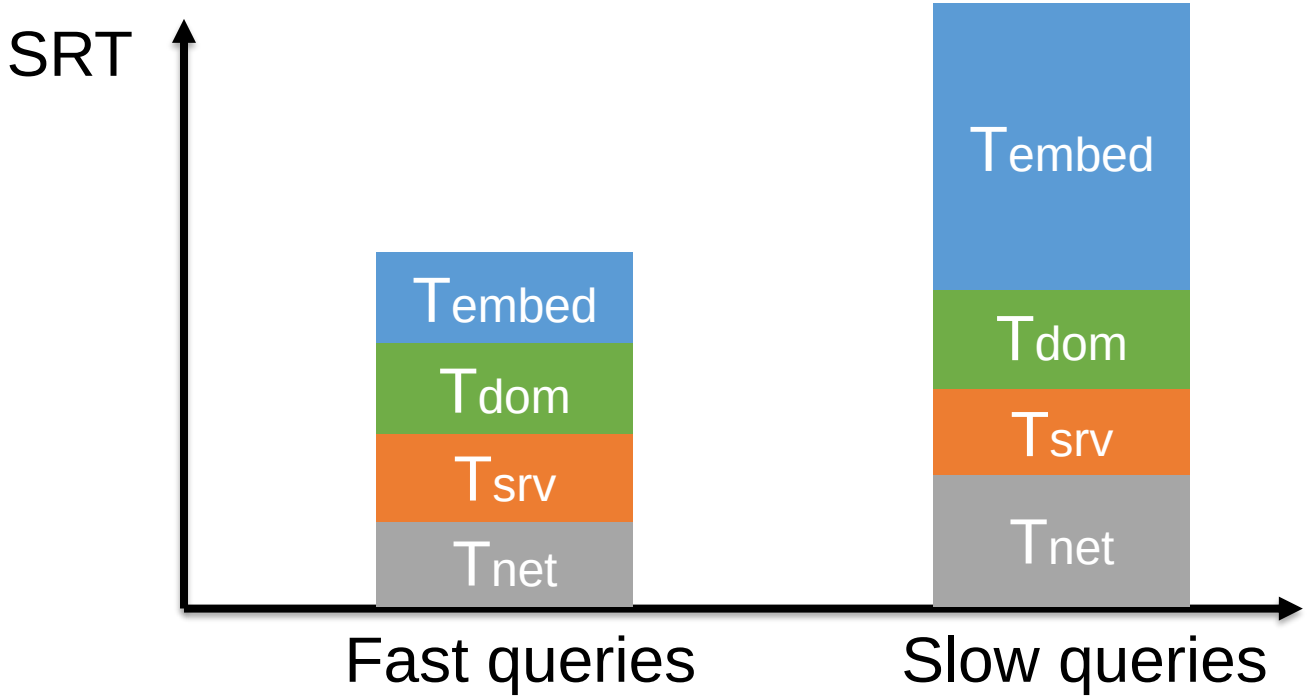
Identifying Bottleneck Clusters

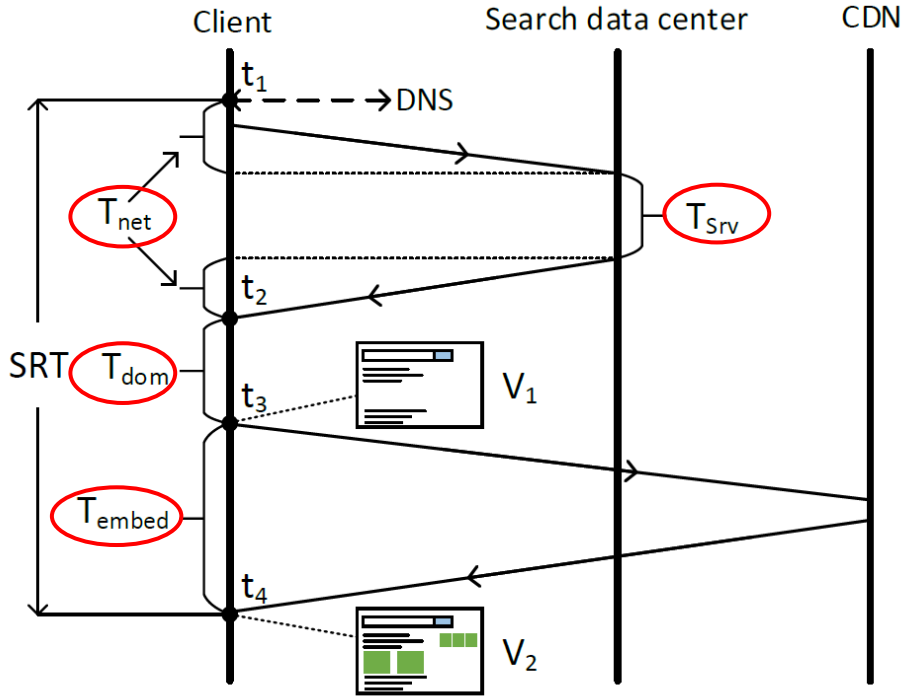
- Results (Δ = 5%)



Identifying Bottleneck SRT Components

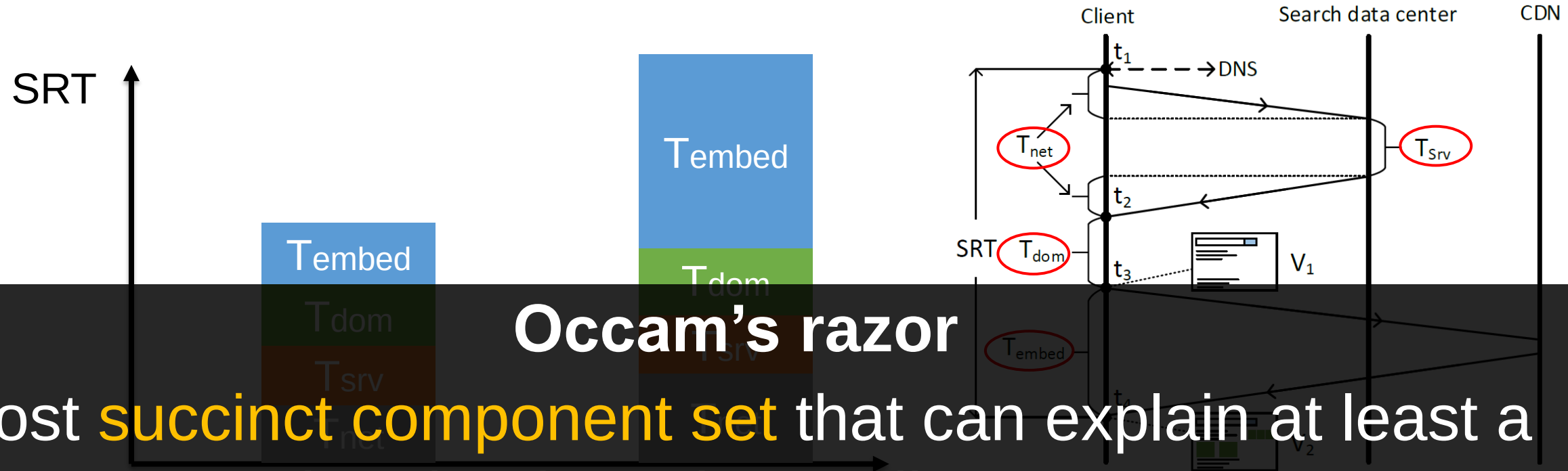
- In a bottleneck cluster, which SRT component(s) explain slow queries better?





Identifying Bottleneck SRT Components

- In a bottleneck cluster, which SRT component(s) explain slow queries better?

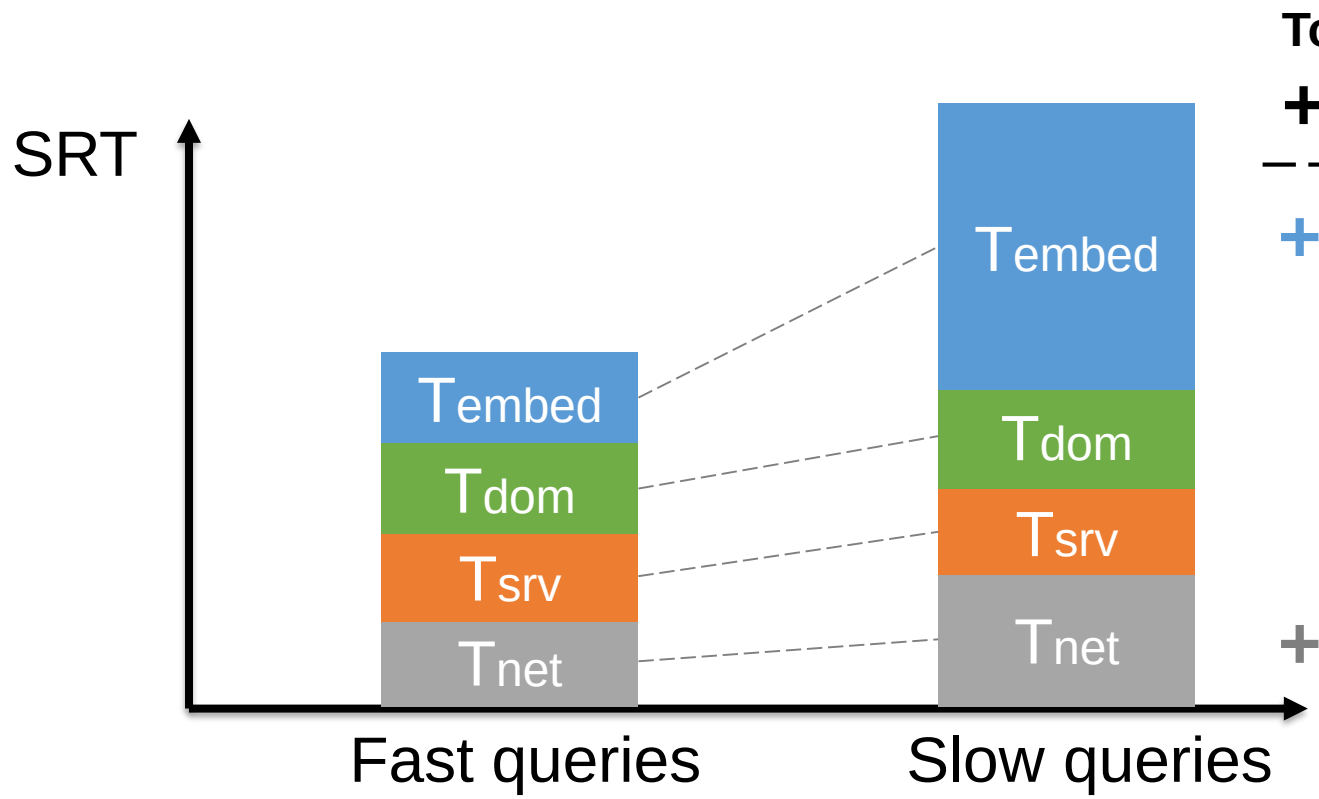


Occam's razor

Most succinct component set that can explain at least a threshold fraction of differences

Identifying Bottleneck SRT Components

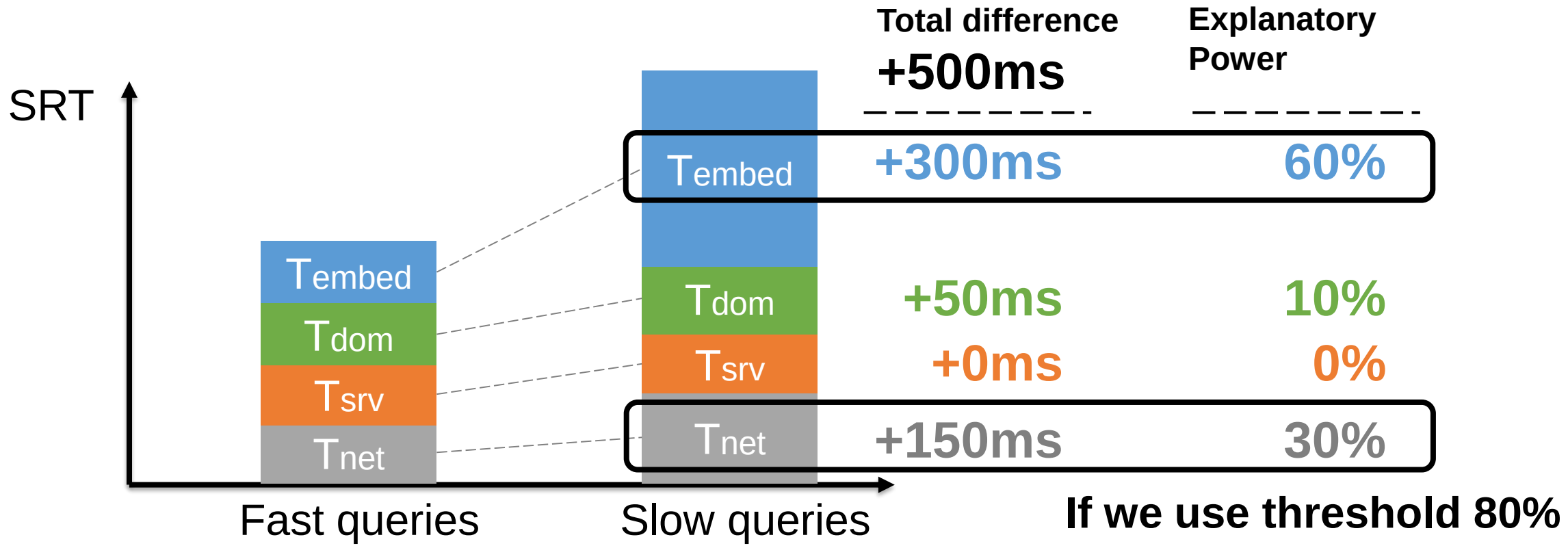
- In a bottleneck cluster, which SRT component(s) explain slow queries better?



	Total difference **+500ms**	Explanatory Power (fraction)
Tembed	+300ms	60%
Tdom	+50ms	10%
Tsrv	+0ms	0%
Tnet	+150ms	30%

Identifying Bottleneck SRT Components

- In a bottleneck cluster, which SRT component(s) explain slow queries better?



Results

- Bottlenecks

ID	Bottleneck Clusters					Bottleneck Components				%Slow query
	CP	NS	$\#image$	$browser$	ad	T_{net}	T_{srv}	T_{dom}	T_{embed}	
B1	*	$worse$	*	*	yes	✓		✓	✓	68.8%
B2	*	$worse$	$many$	*	*	✓			✓	67.1%
B3	*	$worse$	*	$U4$	*	✓			✓	61.6%
B4	*	$worse$	*	*	no	✓			✓	61.0%
B5	$good$	$worse$	*	*	*	✓			✓	59.1%
B6	$worse$	*	*	*	*	✓		✓	✓	53.9%
B7	*	*	$many$	*	yes	✓		✓	✓	34.5%
B8	*	*	*	$U1$	*	✓		✓	✓	26.9%
B9	*	*	$many$	*	no	✓			✓	25.1%
B10	*	*	$many$	$U4$	*	✓			✓	23.2%
B11	$good$	*	$many$	*	*	✓			✓	21.1%
B12	*	*	few	*	no	✓	✓		✓	20.5%
B13	$good$	*	*	$U4$	no	✓			✓	16.5%
B14	*	$better$	*	*	*		✓		✓	10.0%

Results

- Bottlenecks

Network status and #Images are the two major factors

ID	Bottleneck Clusters					Bottleneck Components				%Slow query
	CP	NS	$\#image$	$browser$	ad	T_{net}	T_{srv}	T_{dom}	T_{embed}	
B1	*	$worse$	*	*	yes	✓		✓	✓	68.8%
B2	*	$worse$	$many$	*	*	✓			✓	67.1%
B3	*	$worse$	*	$U4$	*	✓			✓	61.6%
B4	*	$worse$	*	*	no	✓			✓	61.0%
B5	$good$	$worse$	*	*	*	✓			✓	59.1%
B6	$worse$	*	*	*	*	✓		✓	✓	53.9%
B7	*	*	$many$	*	yes	✓		✓	✓	34.5%
B8	*	*	*	$U1$	*	✓		✓	✓	26.9%
B9	*	*	$many$	*	no	✓			✓	25.1%
B10	*	*	$many$	$U4$	*	✓			✓	23.2%
B11	$good$	*	$many$	*	*	✓			✓	21.1%
B12	*	*	few	*	no	✓	✓		✓	20.5%
B13	$good$	*	*	$U4$	no	✓			✓	16.5%
B14	*	$better$	*	*	*		✓		✓	10.0%

Results

Good conditions can still lead to a lot of slow queries though their percentage is low (tail part)

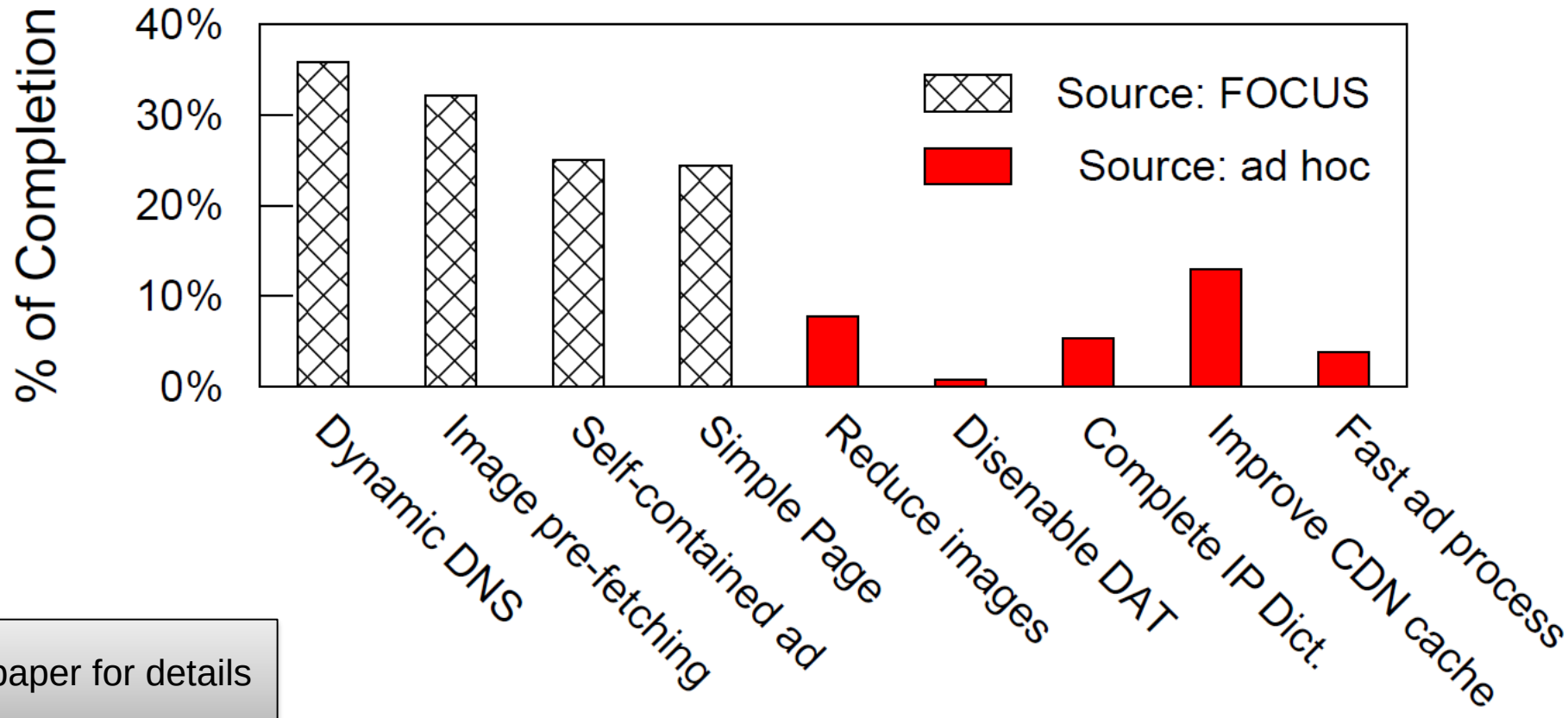
Some impact factors are missing! (e.g., Server-side load)

- Bottlenecks

ID	Bottleneck Clusters					Bottleneck Components				%Slow query
	CP	NS	$\#image$	$browser$	ad	T_{net}	T_{srv}	T_{dom}	T_{embed}	
B1	*	$worse$	*	*	yes	✓		✓	✓	68.8%
B2	*	$worse$	$many$	*	*	✓			✓	67.1%
B3	*	$worse$	*	$U4$	*	✓			✓	61.6%
B4	*	$worse$	*	*	no	✓			✓	61.0%
B5	$good$	$worse$	*	*	*	✓			✓	59.1%
B6	$worse$	*	*	*	*	✓		✓	✓	53.9%
B7	*	*	$many$	*	yes	✓		✓	✓	34.5%
B8	*	*	*	$U1$	*	✓		✓	✓	26.9%
B9	*	*	$many$	*	no	✓			✓	25.1%
B10	*	*	$many$	$U4$	*	✓			✓	23.2%
B11	$good$	*	$many$	*	*	✓			✓	21.1%
B12	*	*	few	*	no	✓	✓		✓	20.5%
B13	$good$	*	*	$U4$	no	✓			✓	16.5%
B14	*	$better$	*	*	*		✓		✓	10.0%

What-if Simulation for Improvement Solutions

- Solutions focusing on the bottlenecks output by FOCUS are more effective



Please read the paper for details

Conclusion

- Collect SRT related data
  - Potential impact factors
  - SRT components

- Using these data, FOCUS is the **first step** to narrow down the debugging space of slow SRT to some specific directions

- For further analysis, we need
  - Domain knowledge
  - More detailed data

Thank you