Stony Brook University

Deconstructing the Energy Consumption of the Mobile Page Load

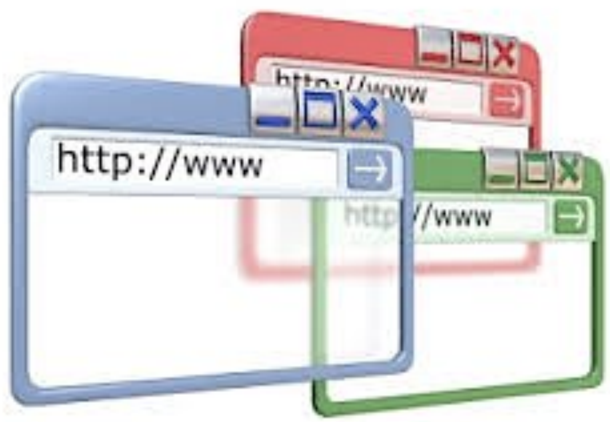
Yi Cao

Joint work with:
Javad Nejati, Muhammad Wajahat, Aruna Balasubramanian, Anshul Gandhi

Department of Computer Science, Stony Brook University

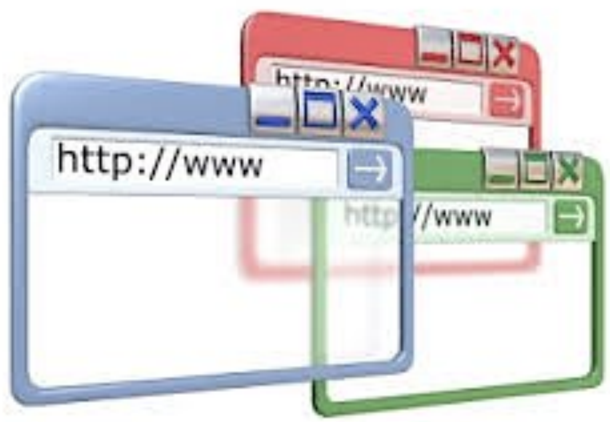
Overview

- Web browser — popular app on phones



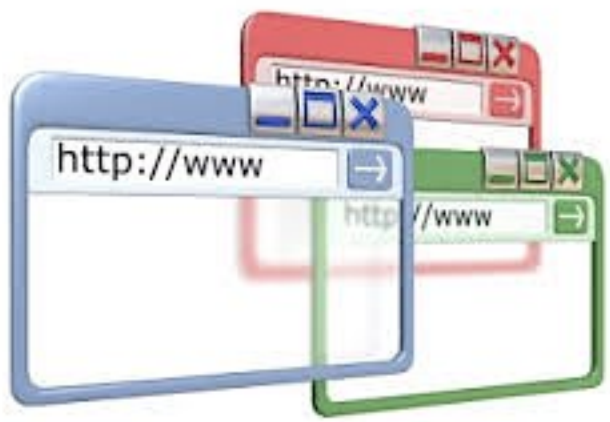
Overview

- Web browser — popular app on phones
    - Page speed is critical to users
    - Several Web optimizations to improve performance



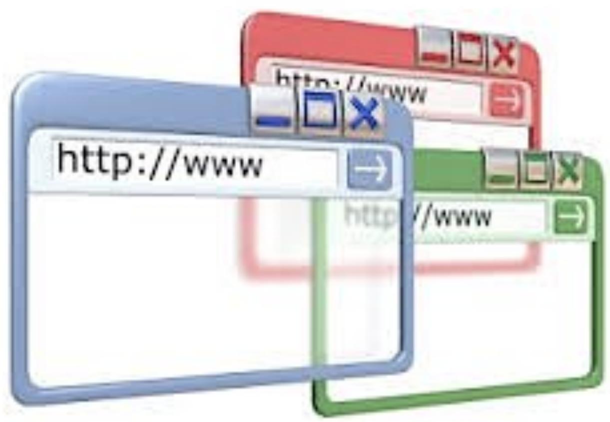
Overview

- Web browser — popular app on phones
  - Page speed is critical to users
  - Several Web optimizations to improve performance



Overview

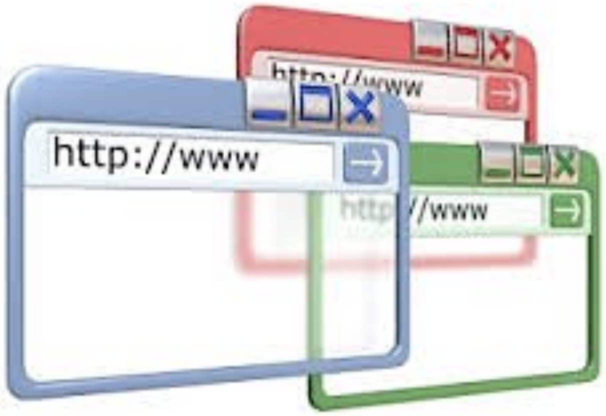
- Web browser — popular app on phones
  - Page speed is critical to users
  - Several Web optimizations to improve performance



- However, often ignore a crucial factor — **Energy**
  - Mobile devices are severely constrained by energy

Overview

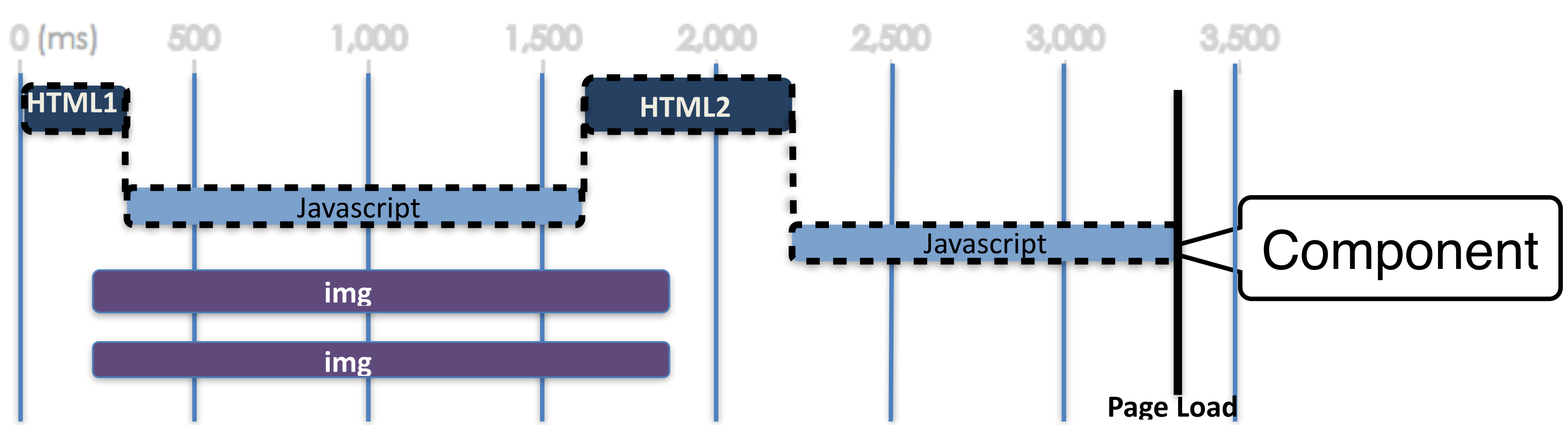
- Web browser — popular app on phones
  - Page speed is critical to users
  - Several Web optimizations to improve performance



- However, often ignore a crucial factor — **Energy**
  - Mobile devices are severely constrained by energy
  - **Reducing page load time may not imply energy savings**

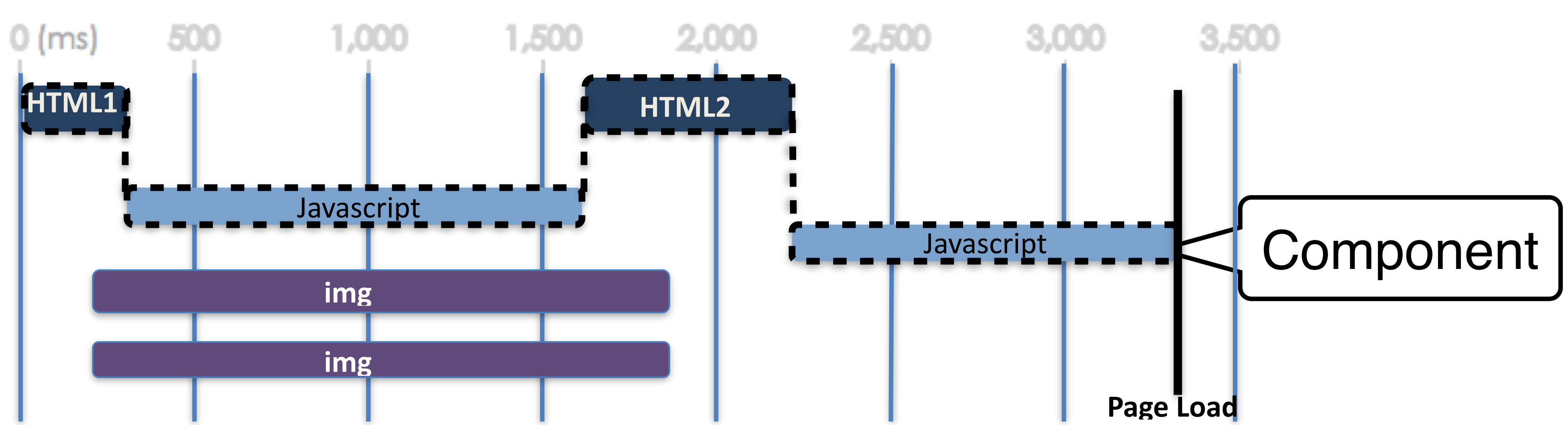
Page Load Process

- Page load activities (*Components*)
  - Computation: Evaluating HTML, Javascript, CSS.
  - Network: Downloads.



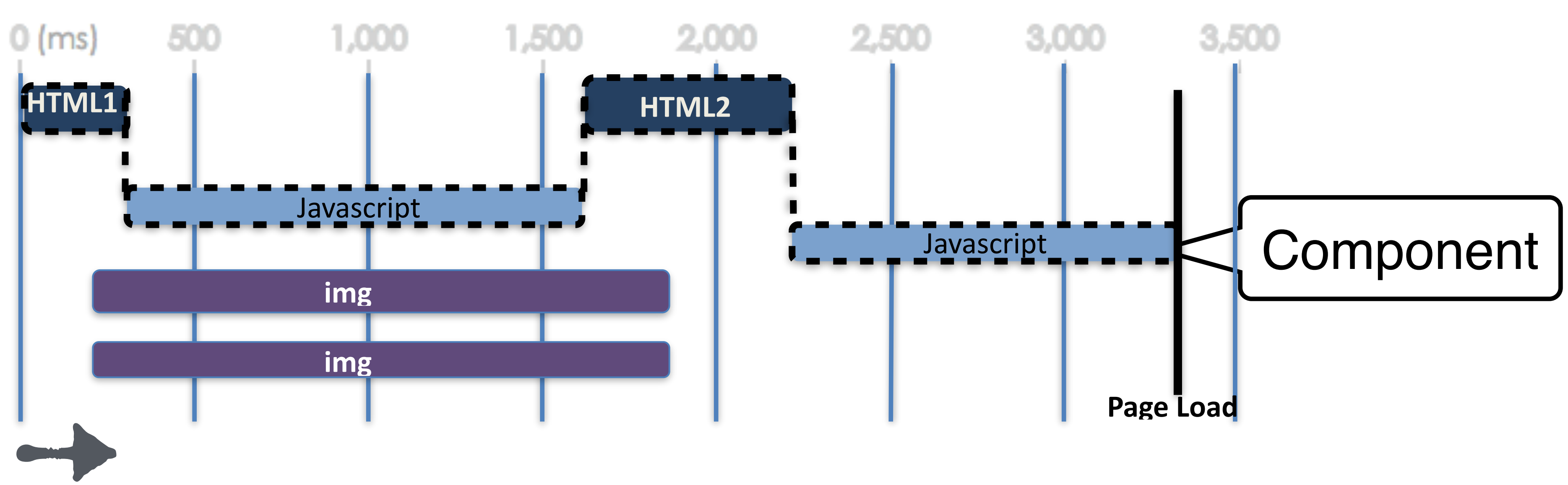
Page Load Process

- Page load activities (*Components*)
  - Computation: Evaluating HTML, Javascript, CSS.
  - Network: Downloads.
- In Browser Profiling Tool — WProf-M
  - Decomposes the page load into different components
  - Provides component type and time information



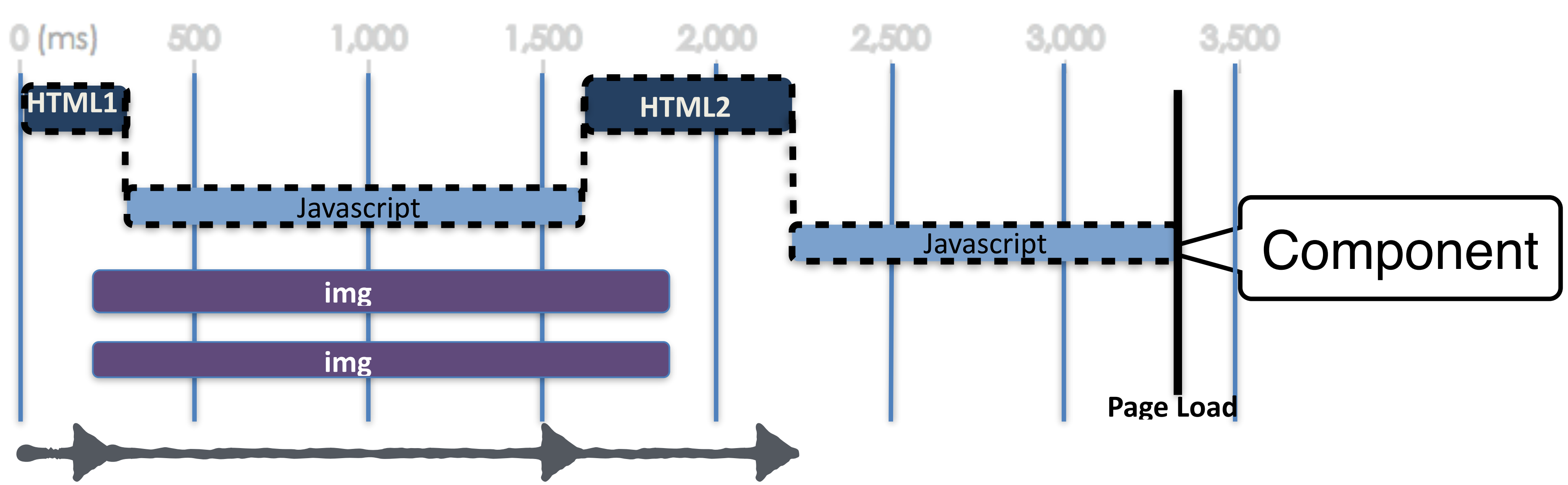
Page Load Process

- Page load activities (*Components*)
  - Computation: Evaluating HTML, Javascript, CSS.
  - Network: Downloads.
- In Browser Profiling Tool — WProf-M
  - Decomposes the page load into different components
  - Provides component type and time information



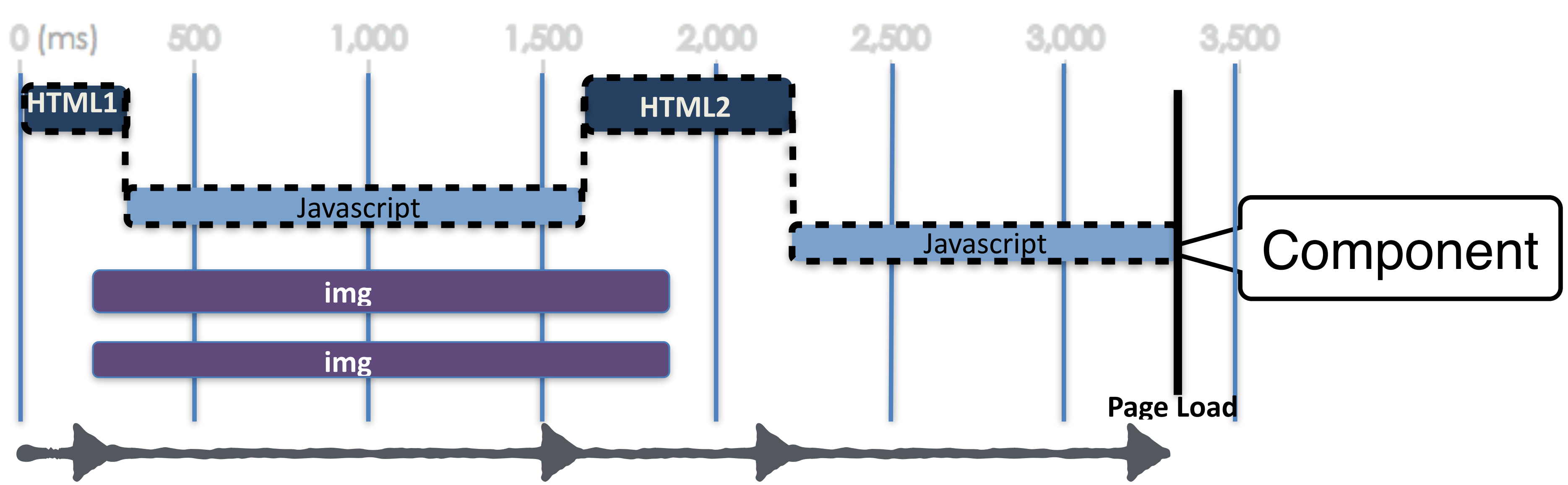
Page Load Process

- Page load activities (*Components*)
  - Computation: Evaluating HTML, Javascript, CSS.
  - Network: Downloads.
- In Browser Profiling Tool — WProf-M
  - Decomposes the page load into different components
  - Provides component type and time information



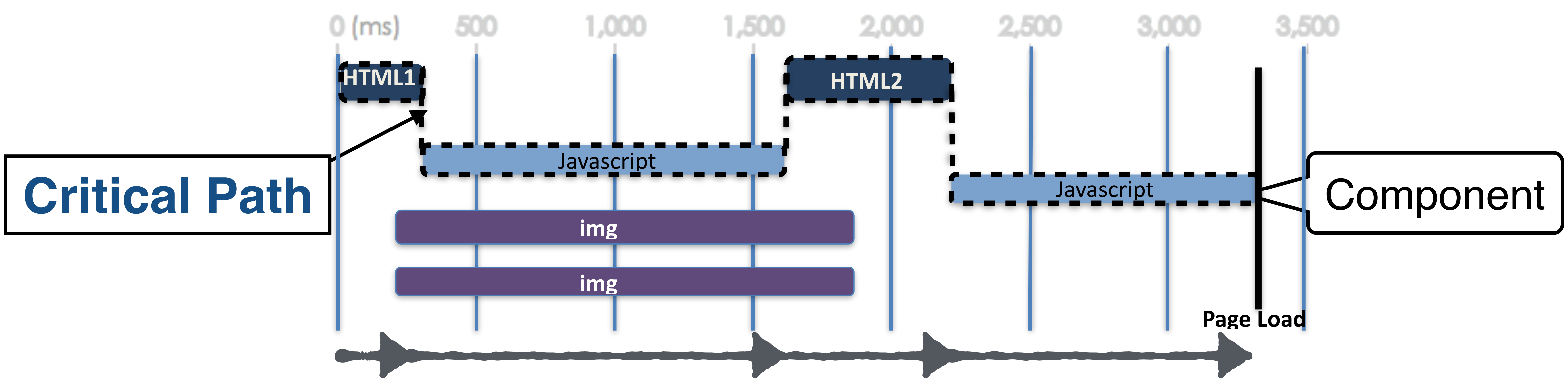
Page Load Process

- Page load activities (*Components*)
  - Computation: Evaluating HTML, Javascript, CSS.
  - Network: Downloads.
- In Browser Profiling Tool — WProf-M
  - Decomposes the page load into different components
  - Provides component type and time information



Page Load Process

- Page load activities (*Components*)
  - Computation: Evaluating HTML, Javascript, CSS.
  - Network: Downloads.
- In Browser Profiling Tool — WProf-M
  - Decomposes the page load into different components
  - Provides component type and time information
  - **Page load time (PLT) is determined by the critical path**



Energy of the Page Load

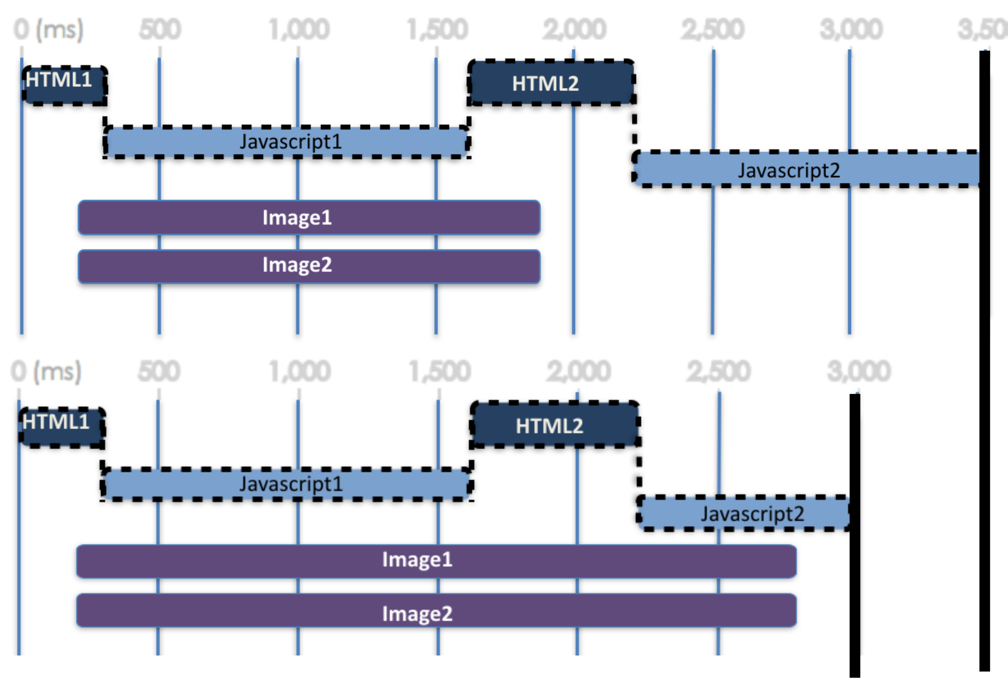
- Reducing PLT may not imply reducing energy
  - While PLT depends on the critical path
  - Energy depends on all page load activities

Energy of the Page Load

- Reducing PLT may not imply reducing energy
  - While PLT depends on the critical path
  - Energy depends on all page load activities

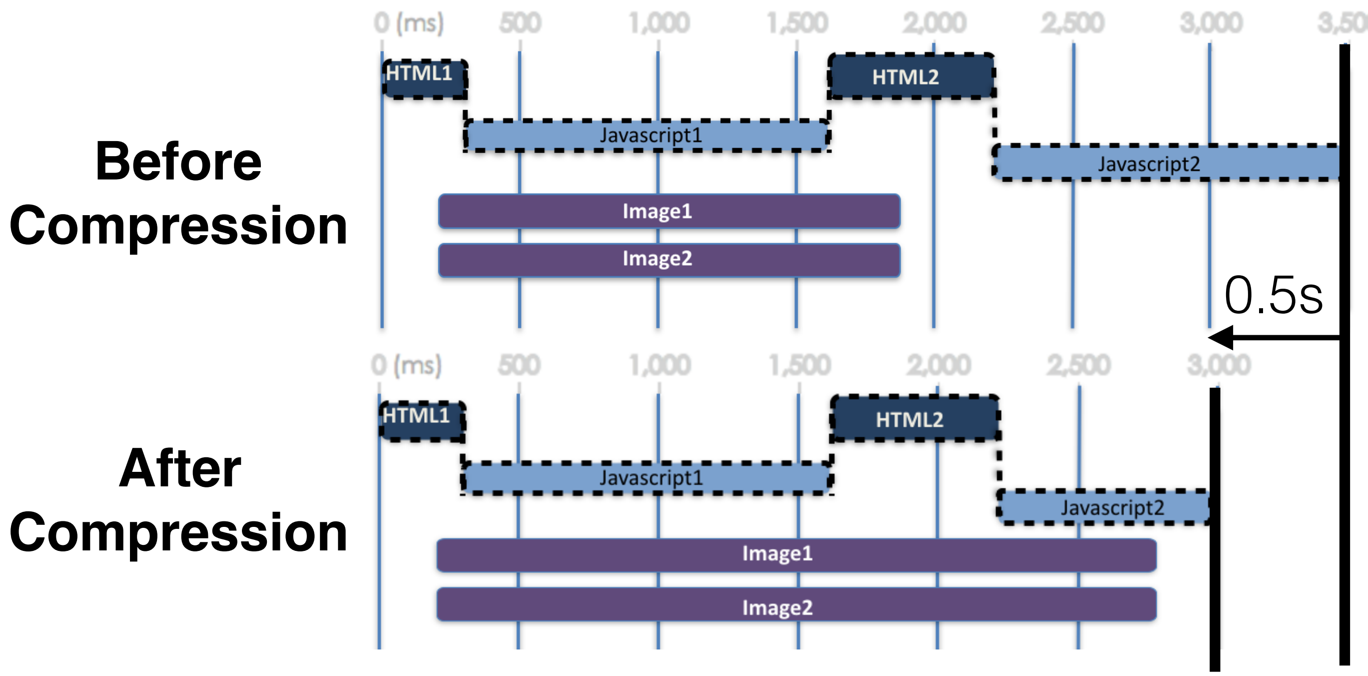
Before Compression

After Compression



Energy of the Page Load

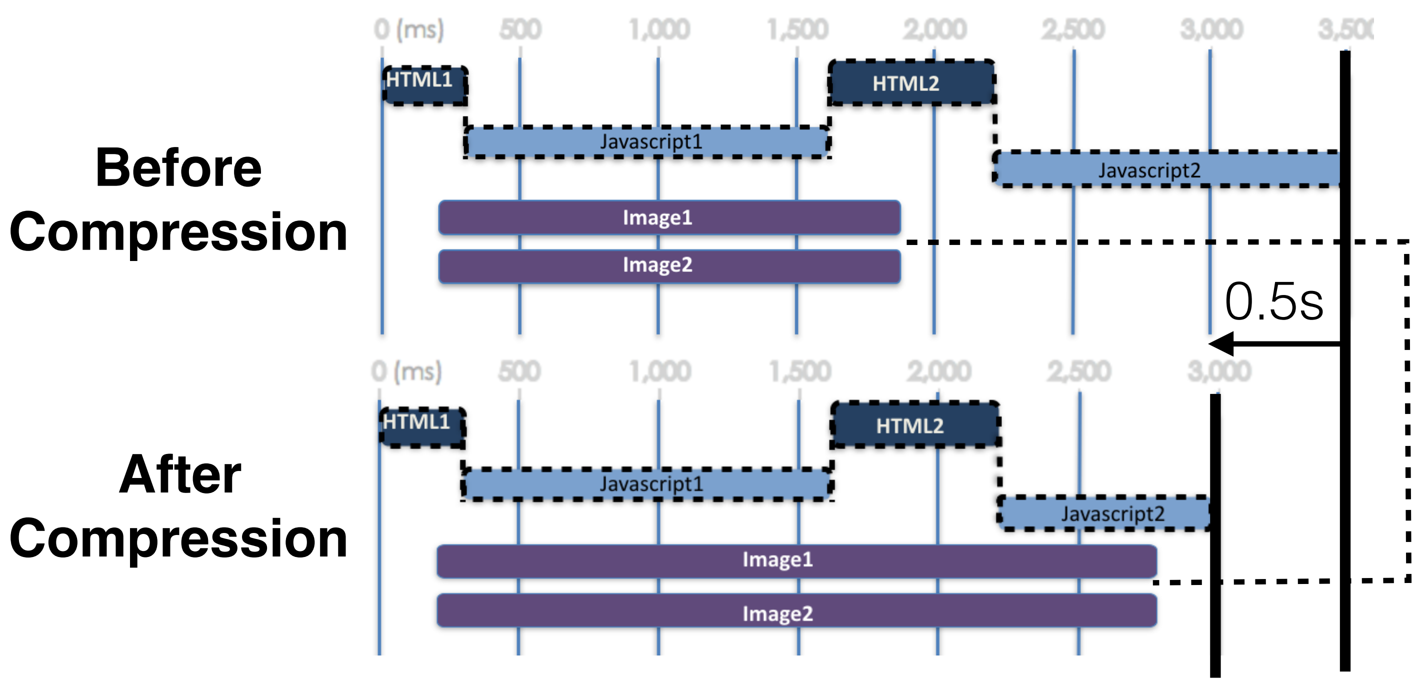
- Reducing PLT may not imply reducing energy
  - While PLT depends on the critical path
  - Energy depends on all page load activities



PLT ↓, However…

Energy of the Page Load

- Reducing PLT may not imply reducing energy
  - While PLT depends on the critical path
  - Energy depends on all page load activities

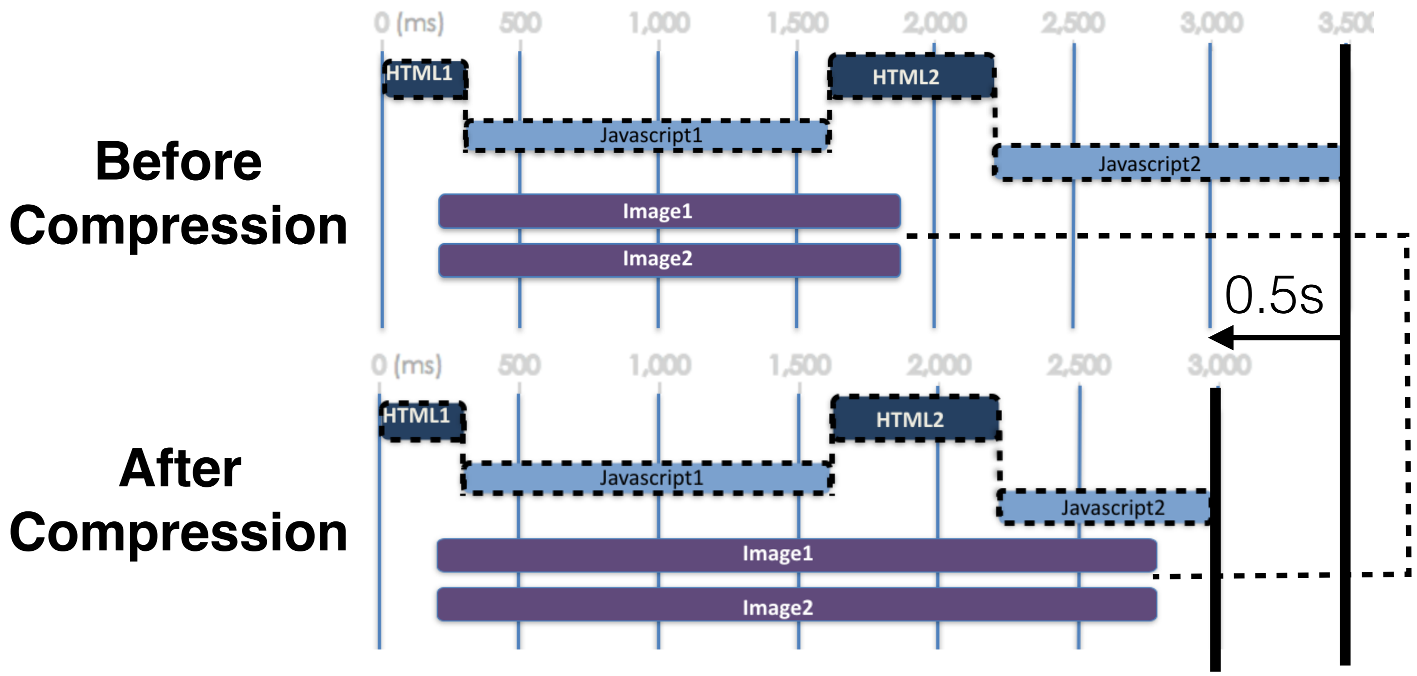


PLT ↓, However…

IMG processing time ⇧ due to decompression

Energy of the Page Load

- Reducing PLT may not imply reducing energy
  - While PLT depends on the critical path
  - Energy depends on all page load activities



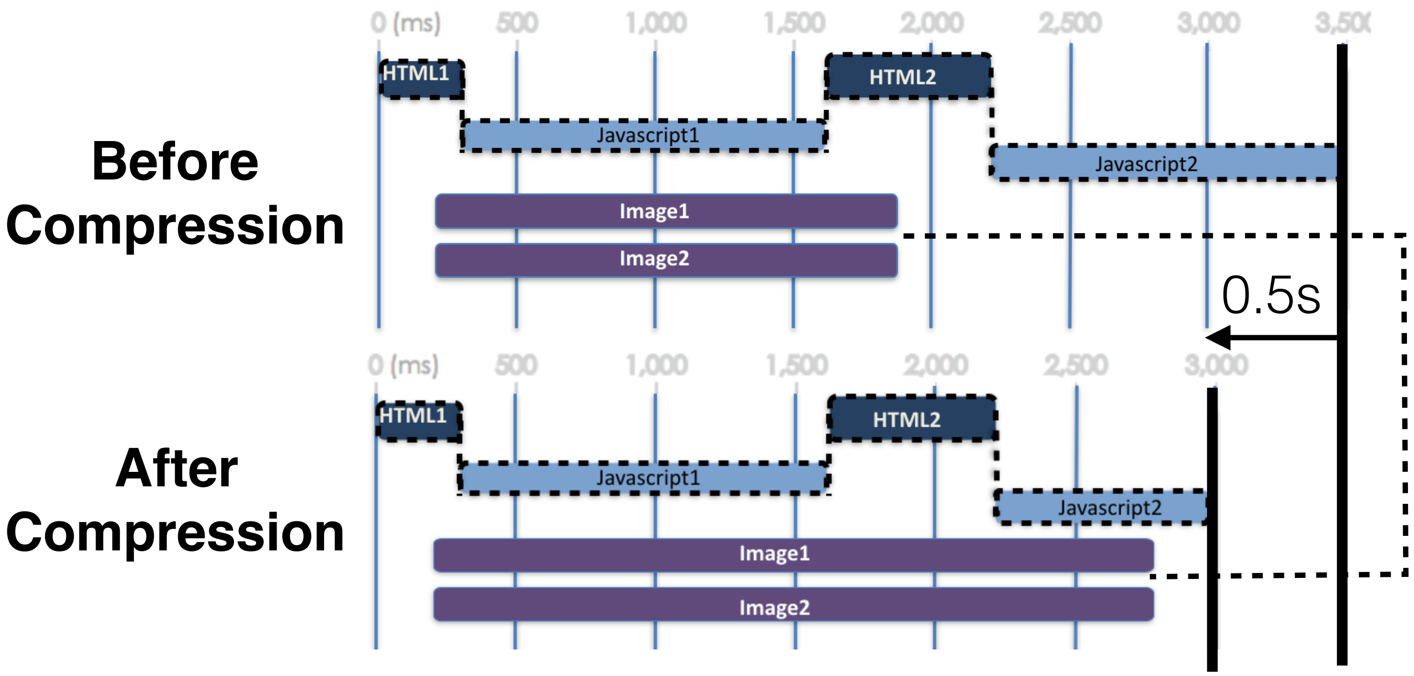
After compression, energy might ↑ although PLT ↓

PLT ↓, However…

IMG processing time ⇧ due to decompression

Energy of the Page Load

- Reducing PLT may not imply reducing energy
  - While PLT depends on the critical path
  - Energy depends on all page load activities



After compression, energy might ↑ although PLT ↓

PLT ↓, However…

IMG processing time ⇧ due to decompression

- To estimate the Web energy, we need to:
  - evaluate the energy of entire page load
  - analyze the energy for *each individual component*

Problem Statement

Problem Statement

1. Can we get quick, accurate power and energy estimations for mobile page loads?

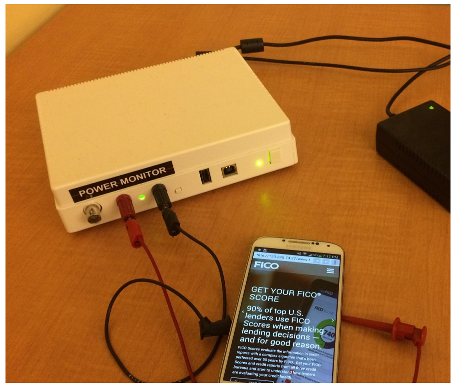
Problem Statement

1. Can we get quick, accurate power and energy estimations for mobile page loads?

2. Is it possible to provide visibility into both how and why Web page enhancements affect energy consumption?

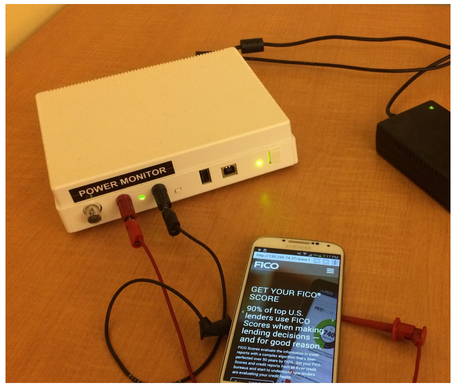
Existing Solutions

- Power Monitors:
  - Measures power consumption accurately



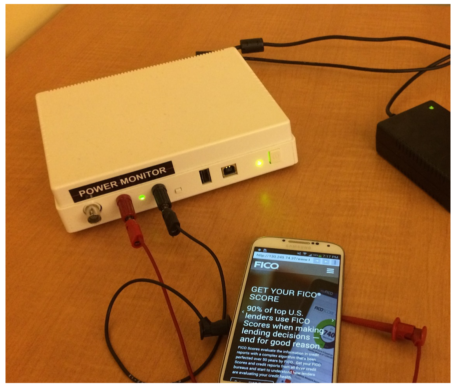
Existing Solutions

- Power Monitors:
  - Measures power consumption accurately
  - But only report aggregate power
  - The energy bottlenecks remain hidden



Existing Solutions

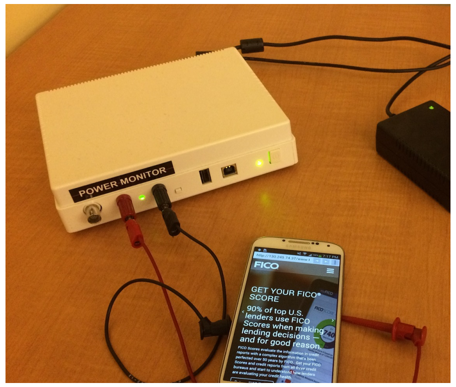
- Power Monitors:
  - Measures power consumption accurately
  - But only report aggregate power
  - The energy bottlenecks remain hidden



- Power Modeling
  - Infers relationship between power and system stats

Existing Solutions

- Power Monitors:
  - Measures power consumption accurately
  - But only report aggregate power
  - The energy bottlenecks remain hidden



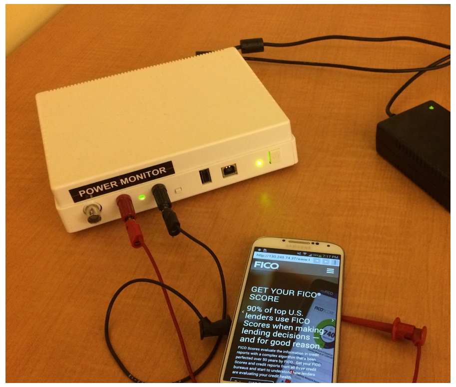
- Power Modeling
  - Infers relationship between power and system stats

$$P(CPU) = \beta \times CPU\_util$$

Stony Brook University

Existing Solutions

- Power Monitors:
  - Measures power consumption accurately
  - But only report aggregate power
  - The energy bottlenecks remain hidden



- Power Modeling
  - Infers relationship between power and system stats

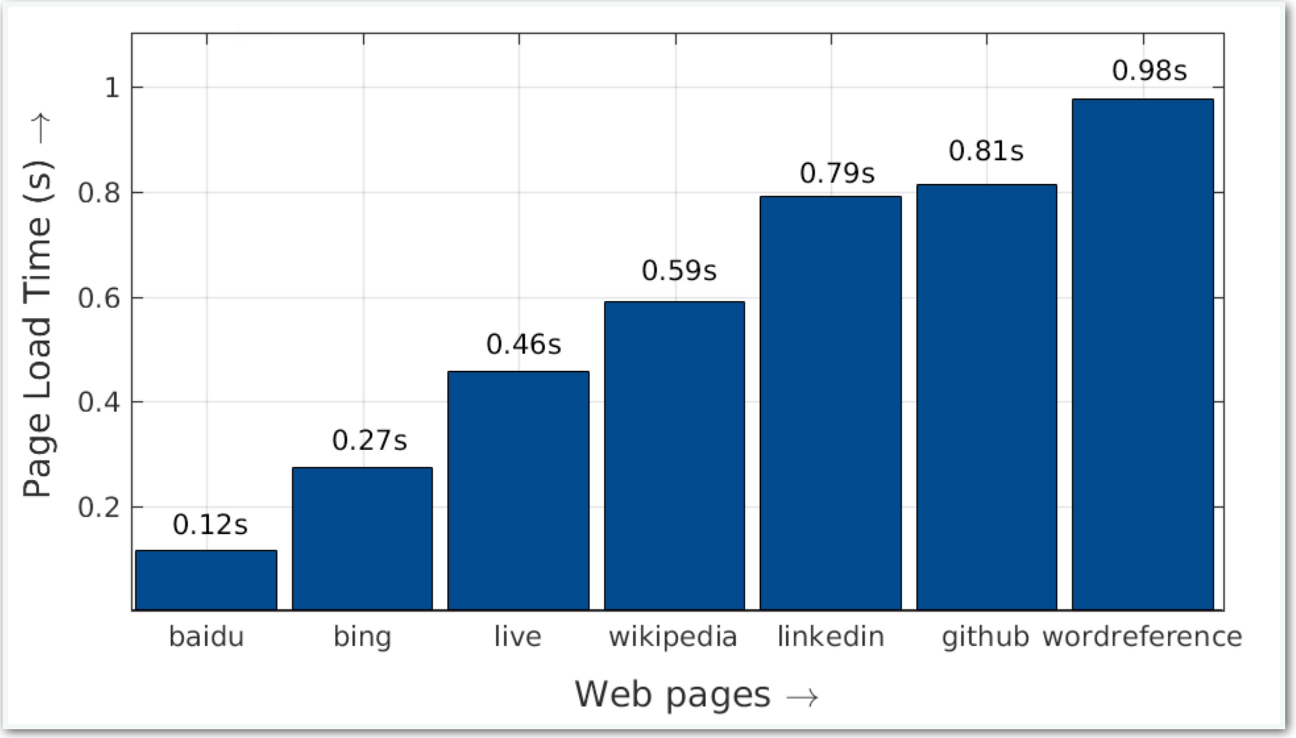
$$P(CPU) = \beta \times CPU\_util$$

  - However, they are not sufficient for mobile Web browsing…

Challenges (1/3)

1. Transcience

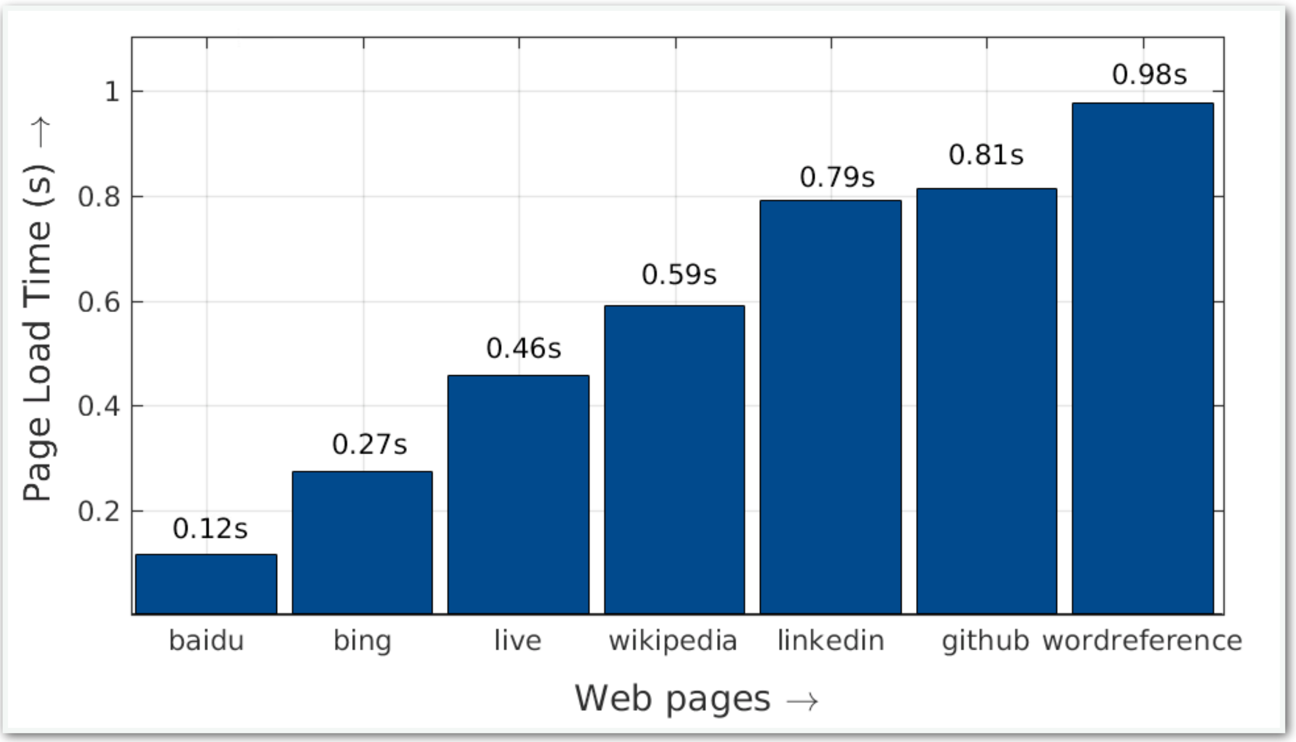
- The page load process is short-lived



Challenges (1/3)

1. Transcience

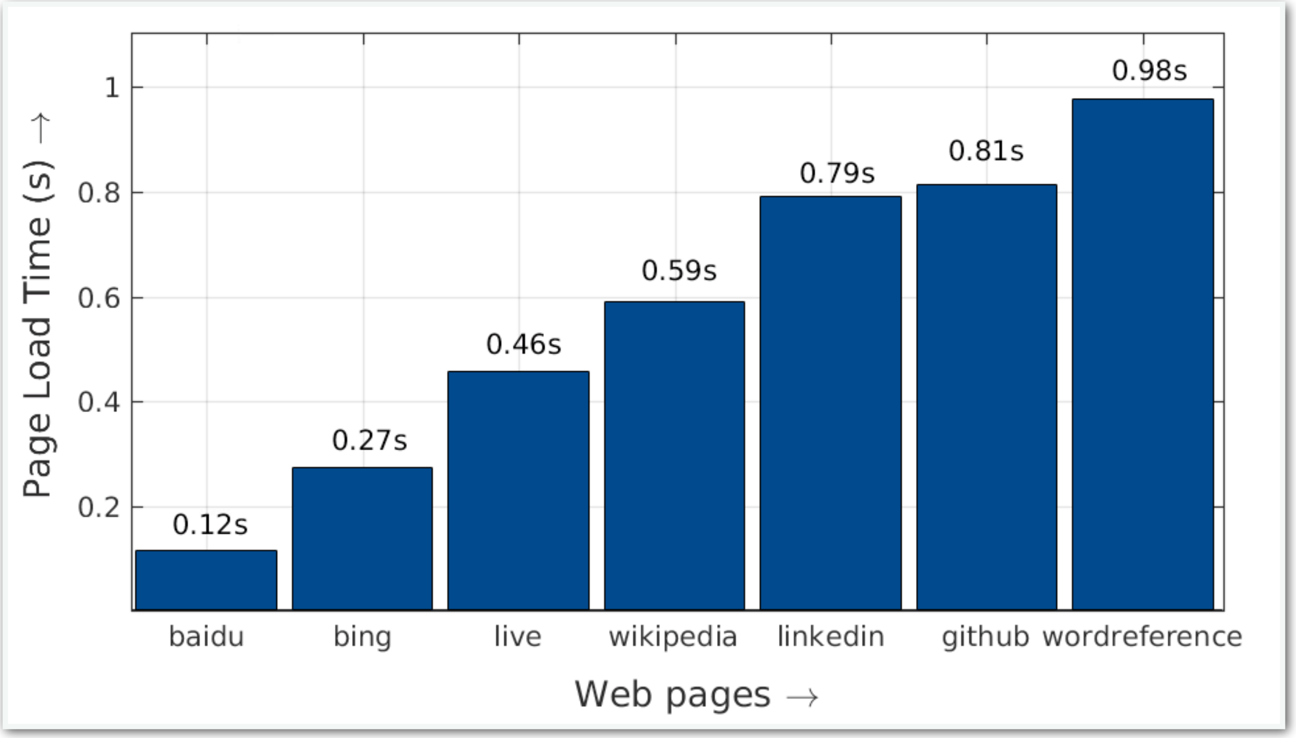
- The page load process is short-lived
- For resource-based power models
  - Need **extremely fine-grained** resource logging to get enough data



Challenges (1/3)

1. Transcience

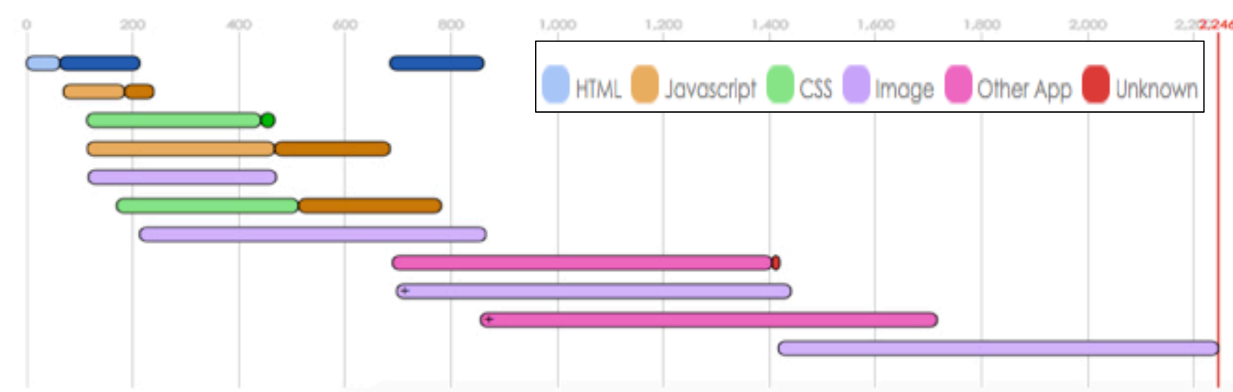
- The page load process is short-lived
- For resource-based power models
  - Need **extremely fine-grained** resource logging to get enough data
  - Frequent resource logging incurs **huge** overhead
    - CPU overhead 30% at 100Hz logging



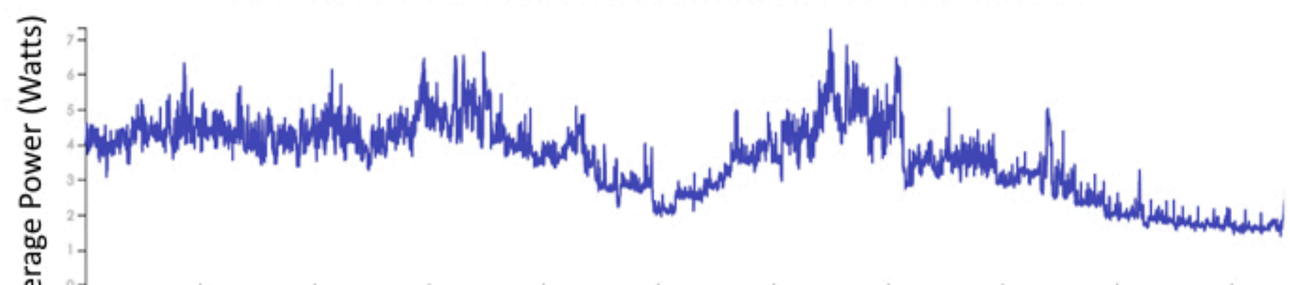
Challenges (2/3)

2. Complexity

- A web page consists of many components



(a) Component level decomposition of loading instagram.com

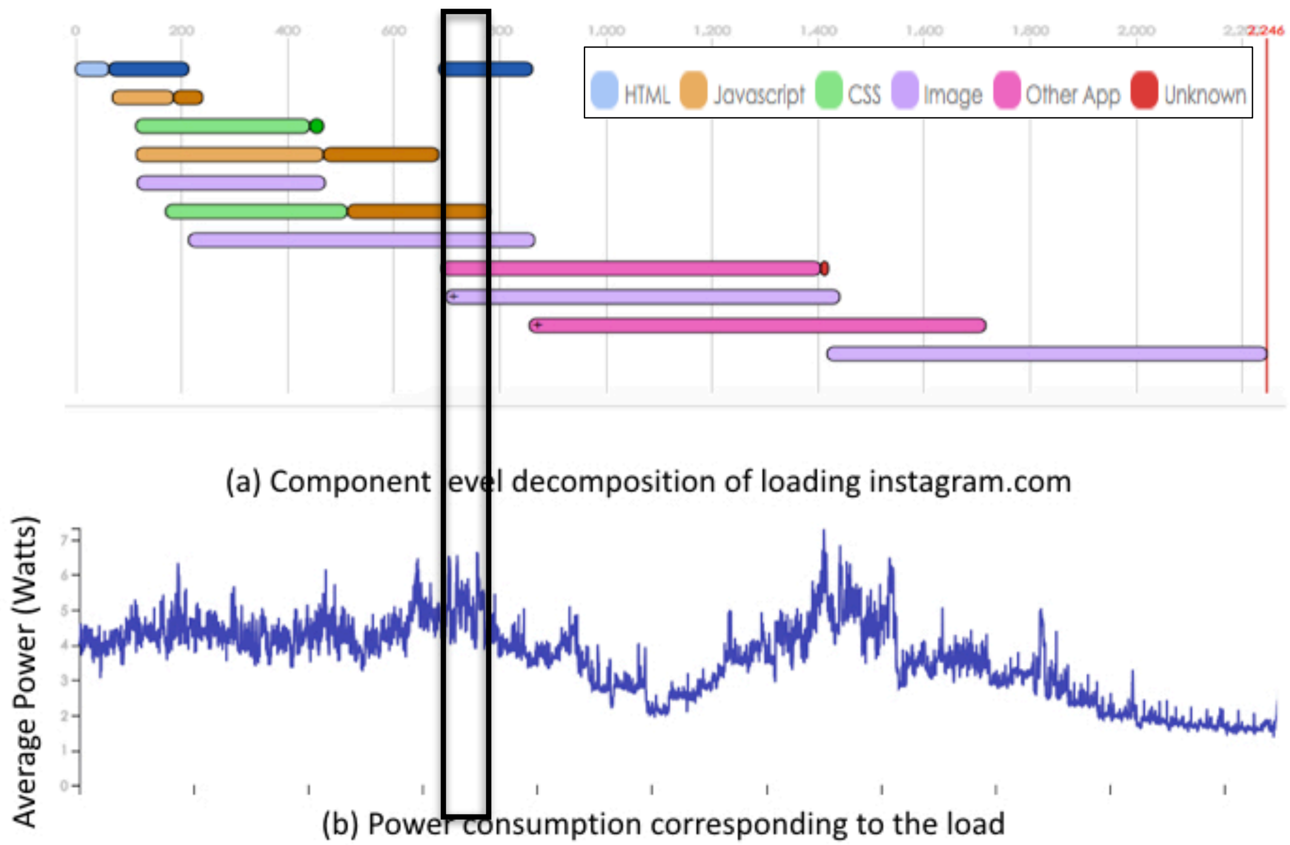


(b) Power consumption corresponding to the load

Challenges (2/3)

2. Complexity

- A web page consists of many components
- Difficult to tease out the energy effects of
  - Specific page load activities



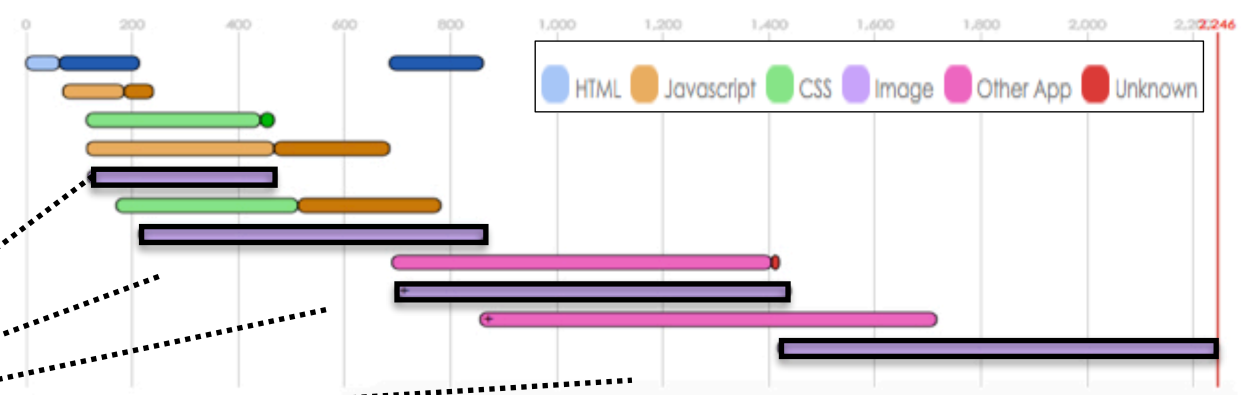
(a) Component level decomposition of loading instagram.com

(b) Power consumption corresponding to the load

Challenges (2/3)

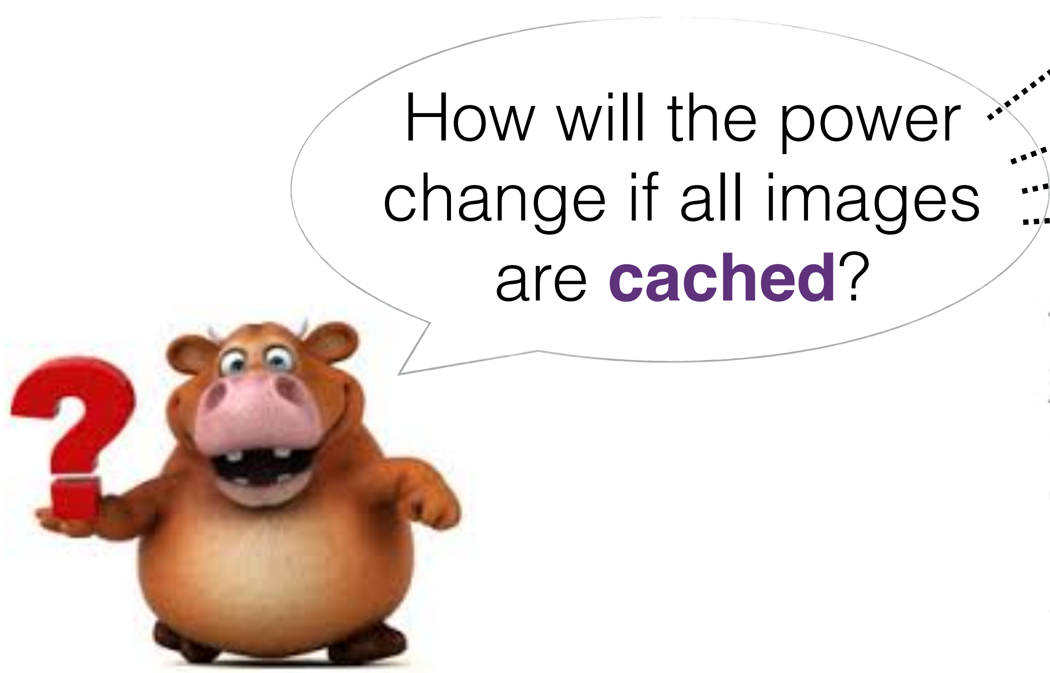
2. Complexity

- A web page consists of many components
- Difficult to tease out the energy effects of
  - Specific page load activities
  - Web optimizations



(a) Component level decomposition of loading instagram.com

(b) Power consumption corresponding to the load



Challenges (3/3)

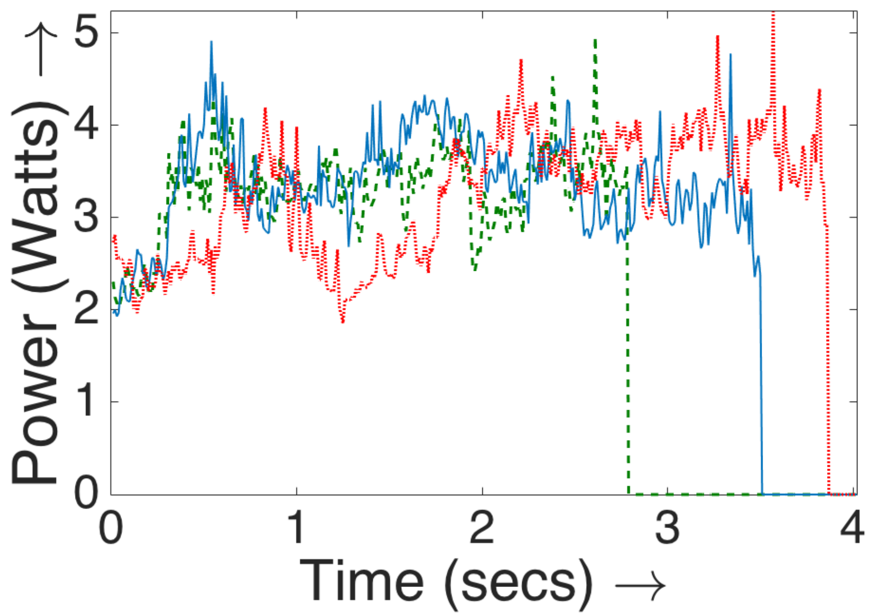
3. Variance
    - Energy and PLT can vary significantly when loaded under the same conditions repeatedly.

Challenges (3/3)

3. Variance

- Energy and PLT can vary significantly when loaded under the same conditions repeatedly.
    - Example: Three runs of answers.yahoo.com

	Red	Blue	Green
PLT(s)	3.9	3.5	2.8
Energy(J)	12.5	11.8	9.2

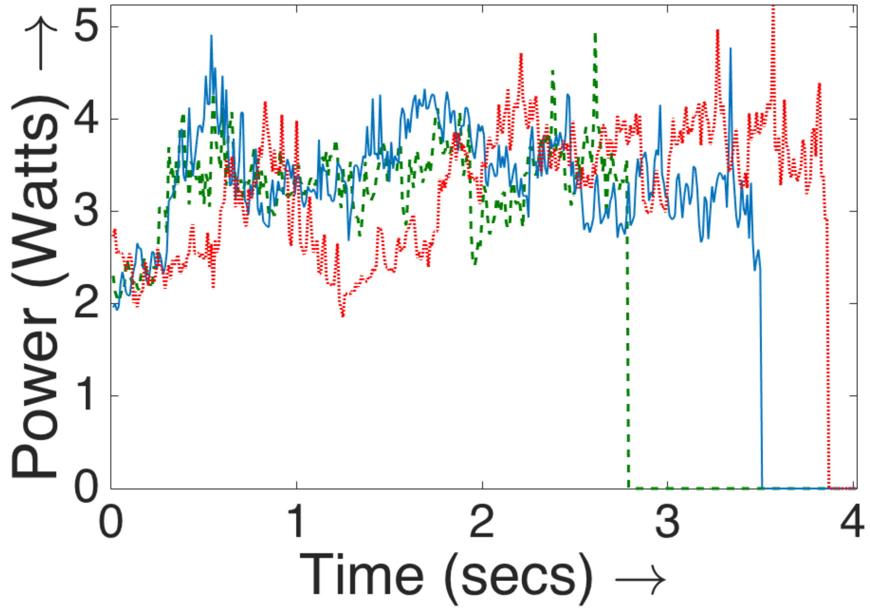


Challenges (3/3)

3. Variance

- Energy and PLT can vary significantly when loaded under the same conditions repeatedly.
  - Example: Three runs of answers.yahoo.com

	Red	Blue	Green
PLT(s)	3.9	3.5	2.8
Energy(J)	12.5	11.8	9.2



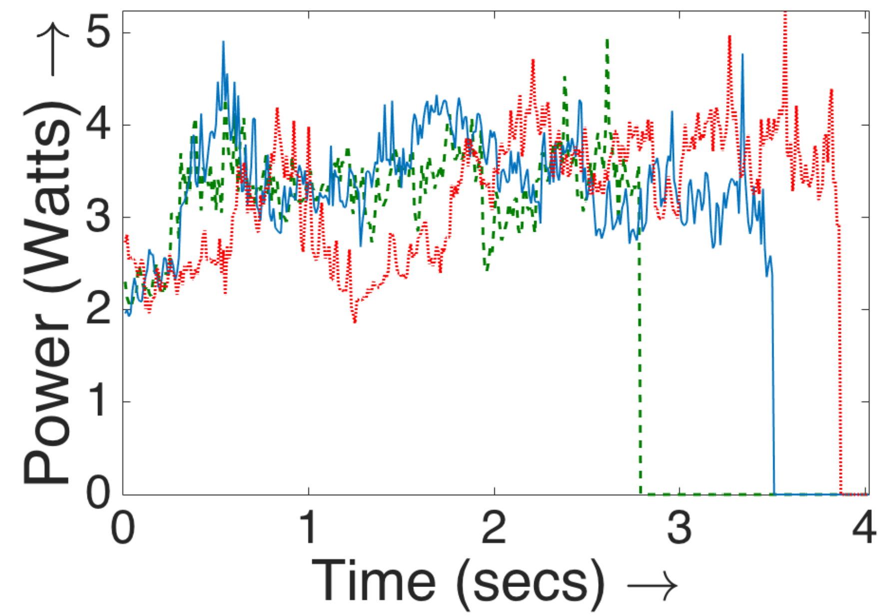
- Difficult to estimate the power consumption of a Web page load simply by referring to previous page loads.

Challenges (3/3)

3. Variance

- Energy and PLT can vary significantly when loaded under the same conditions repeatedly.
  - Example: Three runs of answers.yahoo.com

	Red	Blue	Green
PLT(s)	3.9	3.5	2.8
Energy(J)	12.5	11.8	9.2



- Difficult to estimate the power consumption of a Web page load simply by referring to previous page loads.
- Thus, we focus on power per page load instantiation.

Outline

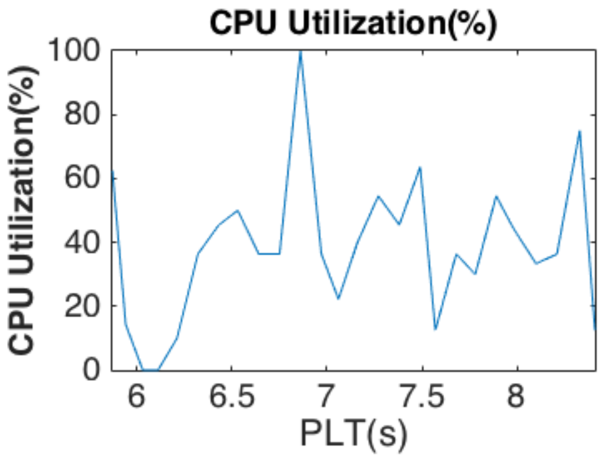
- **RECON**
  - **Idea**
  - **Power Model**
  - **Training & Testing**
- Evaluation & Results
- Application
- Conclusion

High-Level Idea

- Idea: Resource Monitoring + App Semantics

High-Level Idea

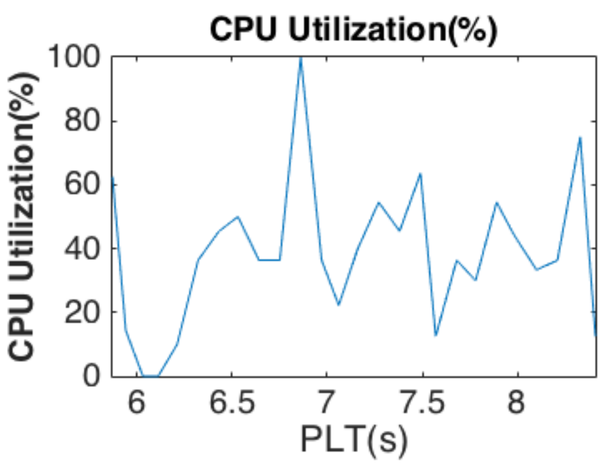
- Idea: Resource Monitoring + App Semantics
  - Coarse-grained resource monitoring (10/sec; 2% overhead)

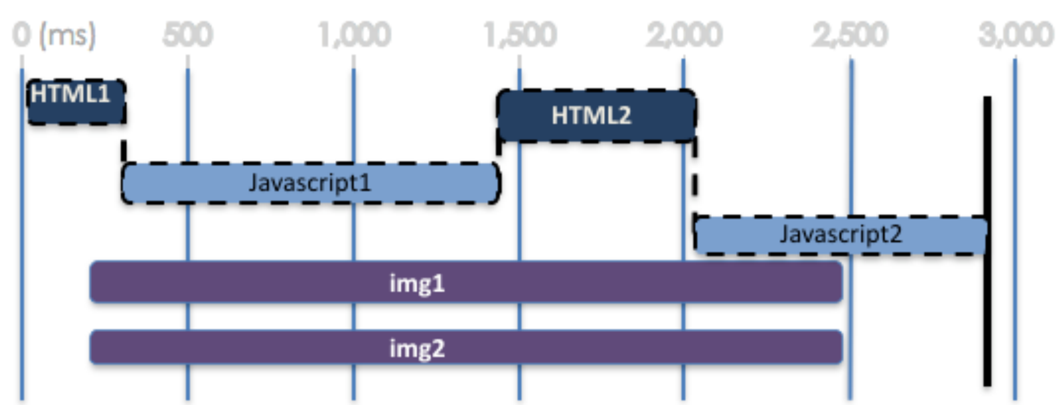


Resource Data
- CPU util/freq
- Bytes sent/recv

High-Level Idea

- Idea: Resource Monitoring + App Semantics
  - Coarse-grained resource monitoring (10/sec; 2% overhead)
  - Augmented by low-level page load semantics from **WProf-M**



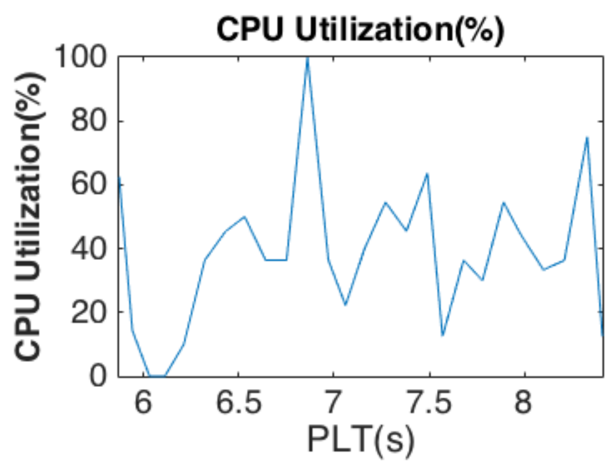


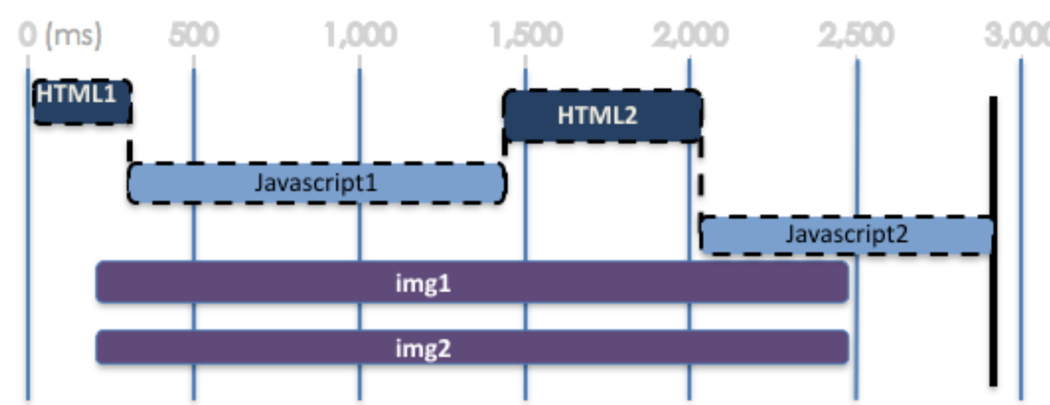
Resource Data
- CPU util/freq
- Bytes sent/recv

Component Data
- Component type
- Component time

High-Level Idea

- Idea: Resource Monitoring + App Semantics
  - Coarse-grained resource monitoring (10/sec; 2% overhead)
  - Augmented by low-level page load semantics from **WProf-M**





Resource Data
- CPU util/freq
- Bytes sent/recv

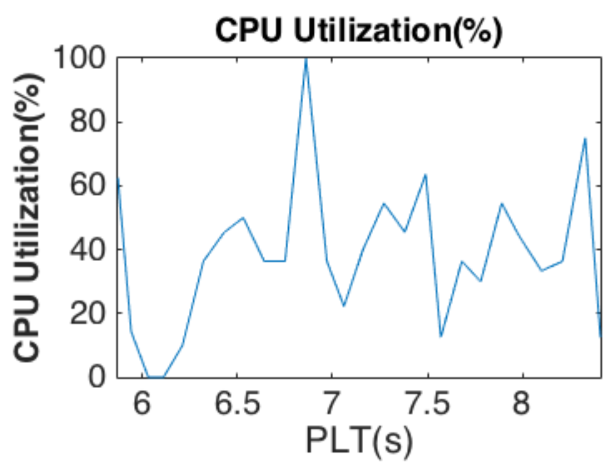
Component Data
- Component type
- Component time

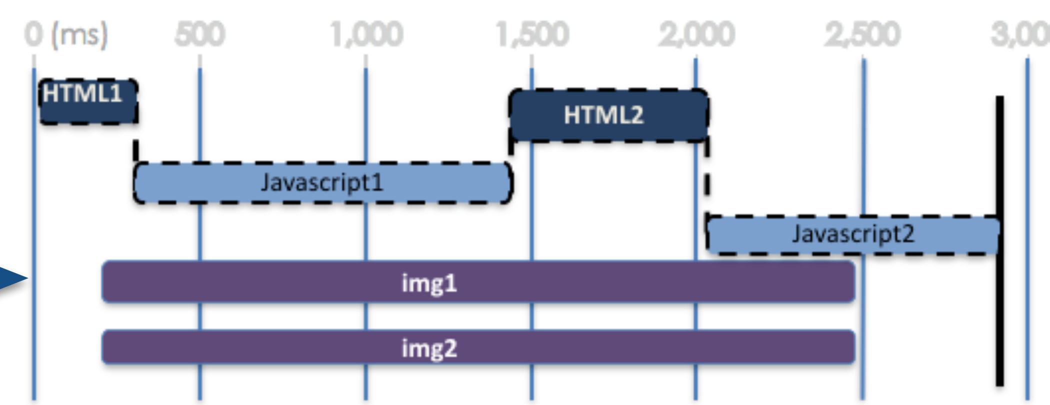
RECON

REsource- and **CO**mpo**N**ent-based modeling

High-Level Idea

- Idea: Resource Monitoring + App Semantics
  - Coarse-grained resource monitoring (10/sec; 2% overhead)
  - Augmented by low-level page load semantics from **WProf-M**





Resource Data
- CPU util/freq
- Bytes sent/recv

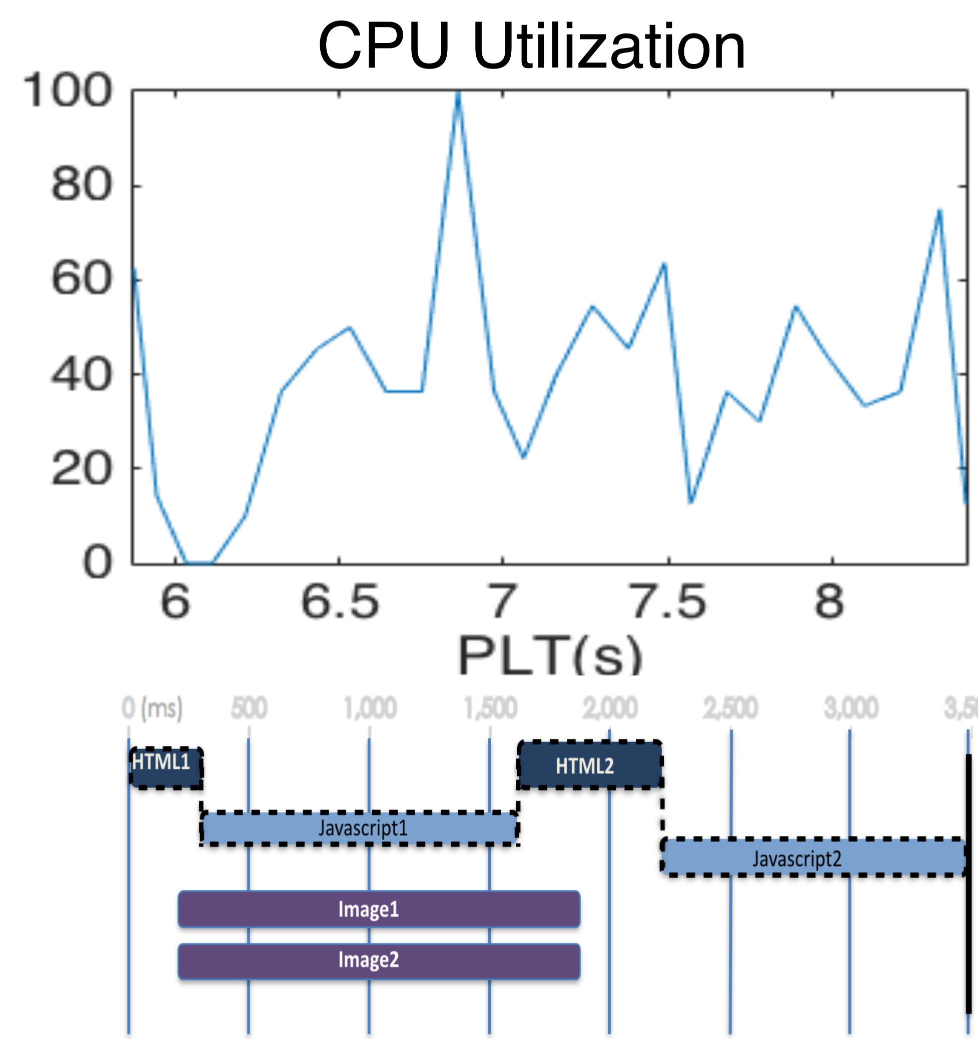
Component Data
- Component type
- Component time

RECON

REsource- and **CO**mpo**N**ent-based modeling

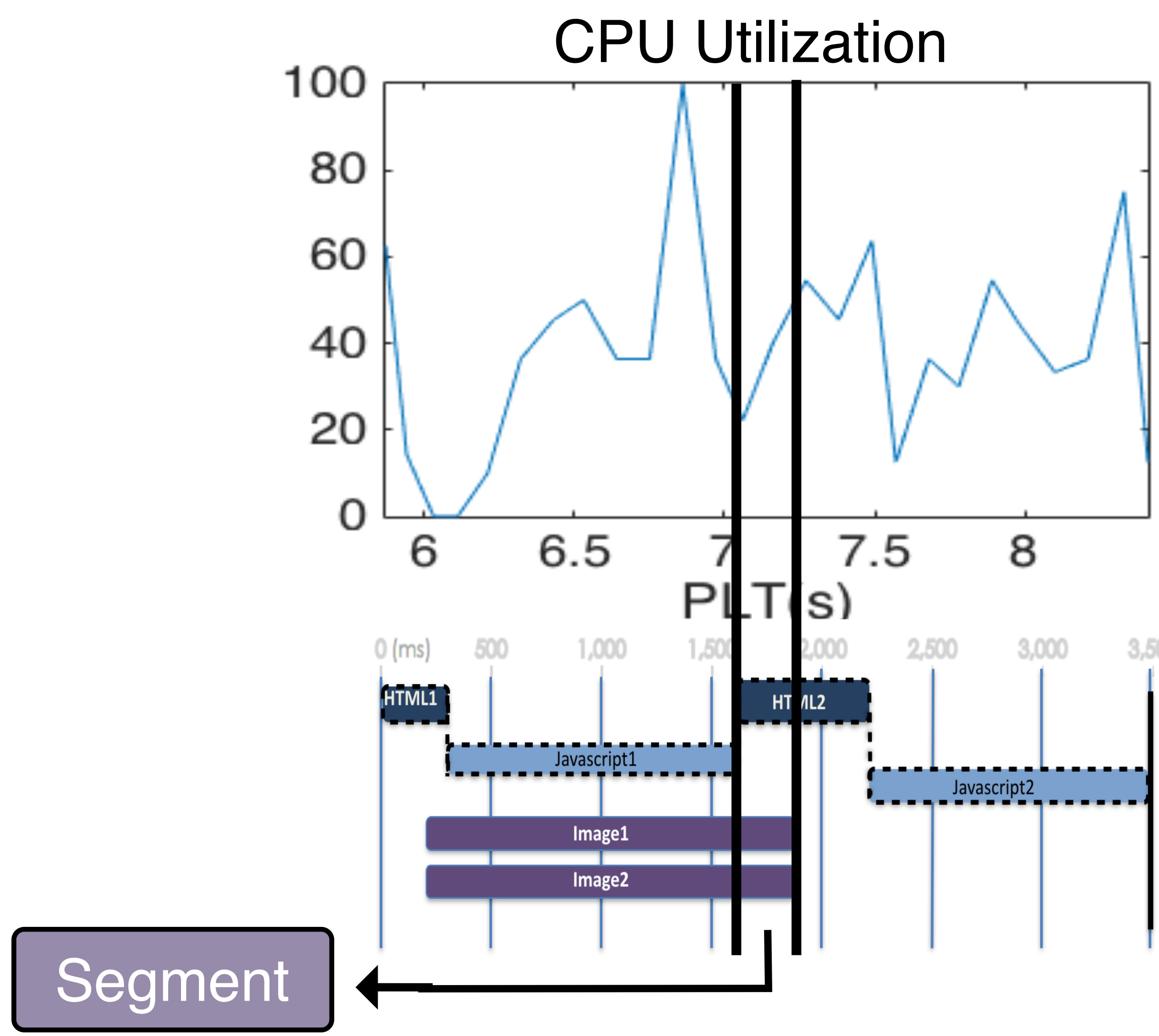
Segmentation

- How to match resource with component information



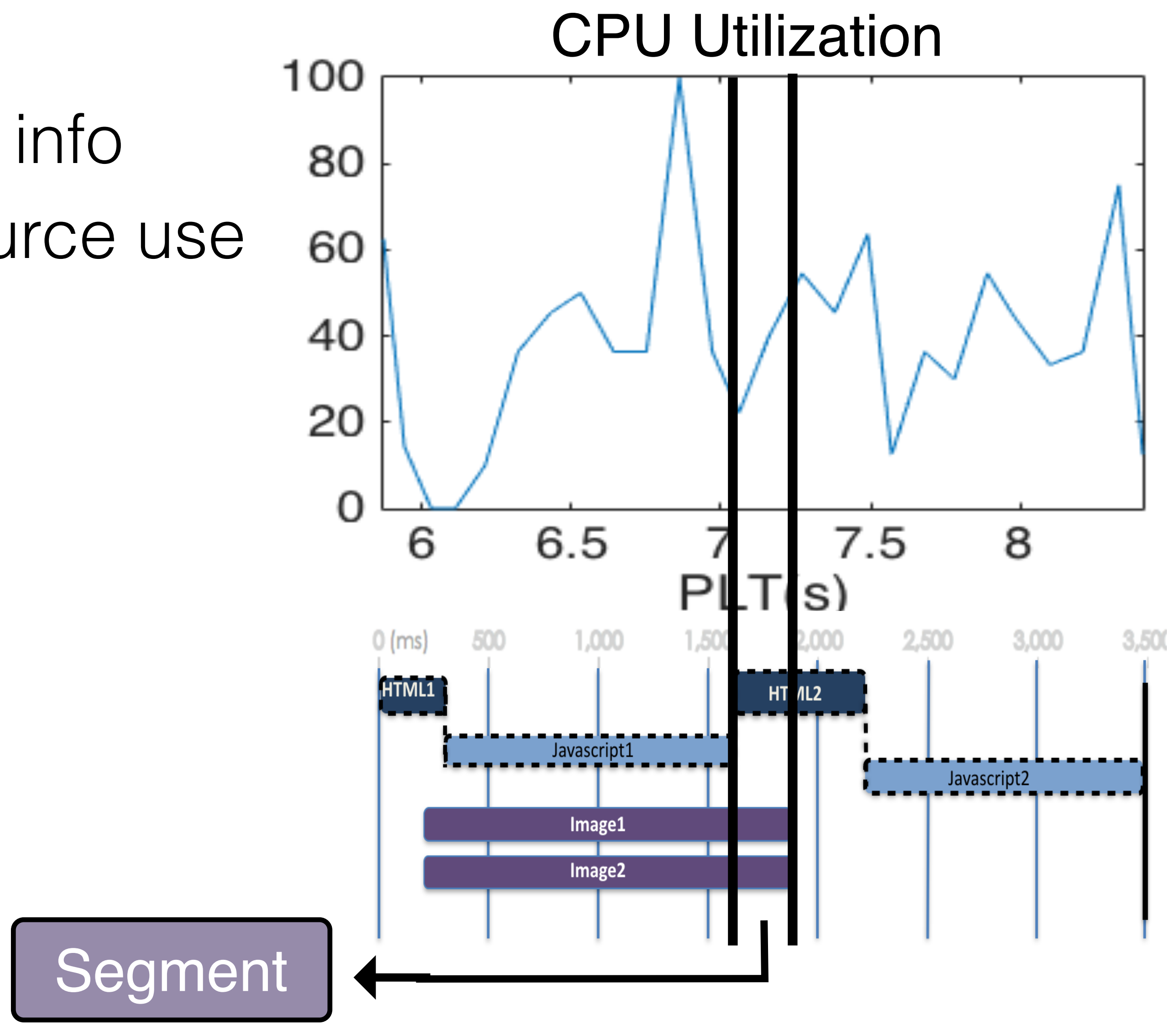
Segmentation

- How to match resource with component information
  - Breakdown the page load process into segments



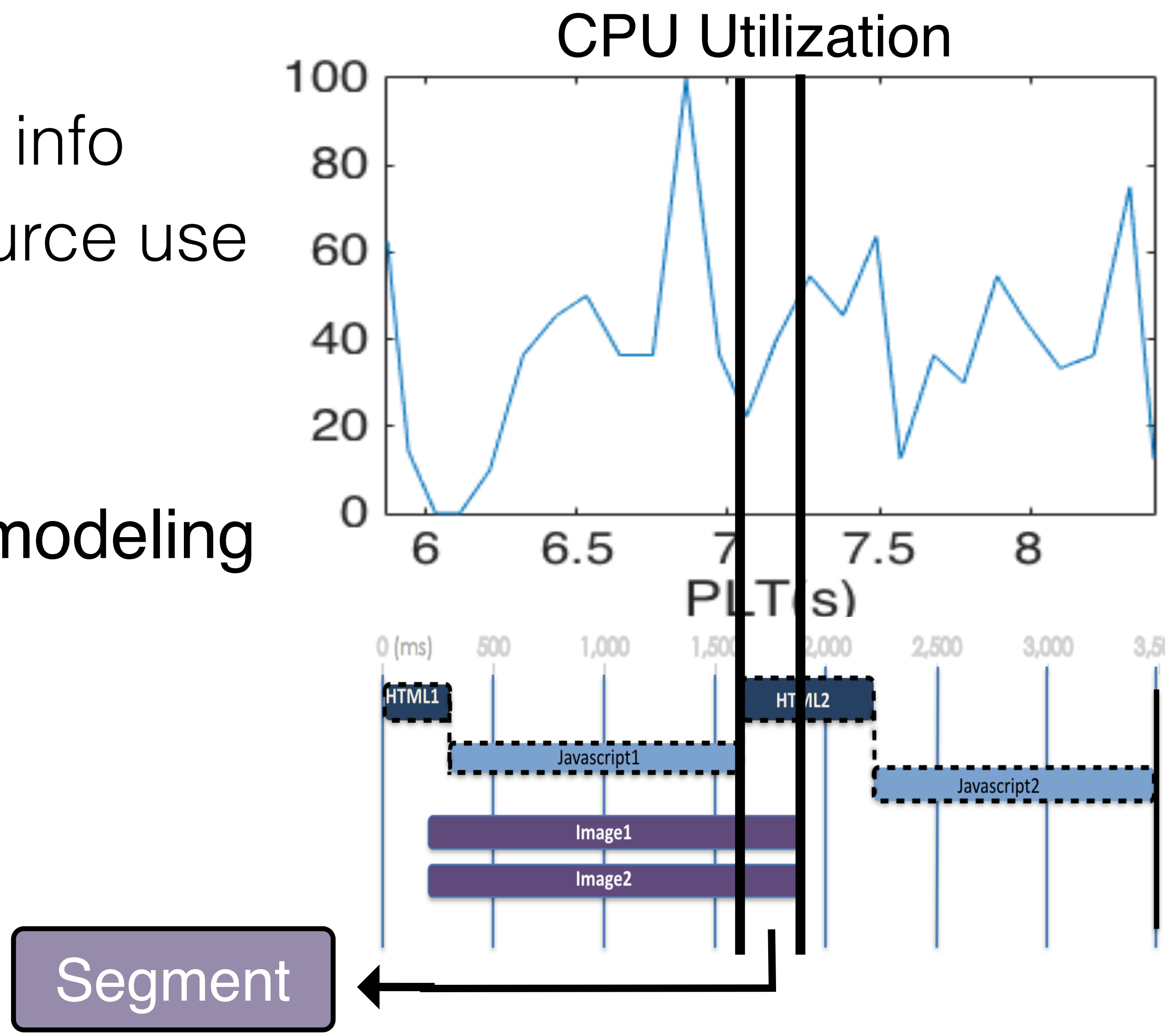
Segmentation

- How to match resource with component information
  - Breakdown the page load process into segments
  - Within each segment..
    - Collect component info
    - Compute avg resource use



Segmentation

- How to match resource with component information
  - Breakdown the page load process into segments
  - Within each segment..
    - Collect component info
    - Compute avg resource use
- *RECON*
  - Segment level power modeling



Linear Regression Model

- Weighted Linear combination

$$P_s \quad = \quad \alpha + \sum_{i \in Resources} \beta_i R_i + \sum_{j \in C_s} \gamma_j F_j,$$

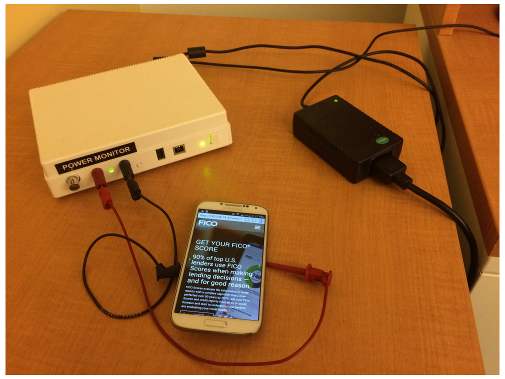
  - Specifically, for each segment

Linear Regression Model

- Weighted Linear combination

$$P_s = \alpha + \sum_{i \in Resources} \beta_i R_i + \sum_{j \in C_s} \gamma_j F_j,$$

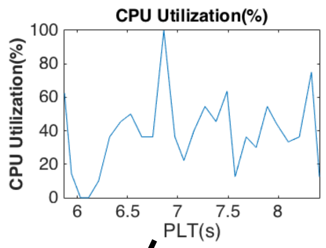
  - Specifically, for each segment
    - P_s (Average power consumption of segment *s*)



Using a power monitor to get P_s just for building the model

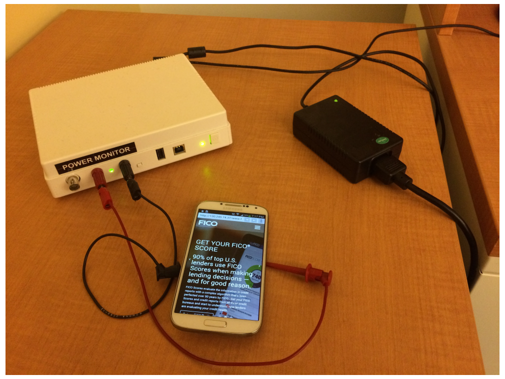
Linear Regression Model

- Weighted Linear combination



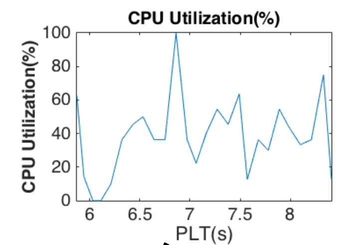
$$P_s = \alpha + \sum_{i \in Resources} \beta_i R_i + \sum_{j \in C_s} \gamma_j F_j,$$

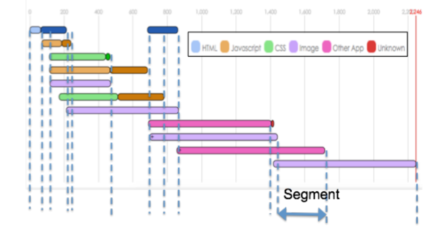
- Specifically, for each segment
  - P_s (Average power consumption of segment *s*)
  - R_i (Resource Usage: CPU %, bytes rx/tx, …)



Using a power monitor to get P_s just for building the model

Linear Regression Model

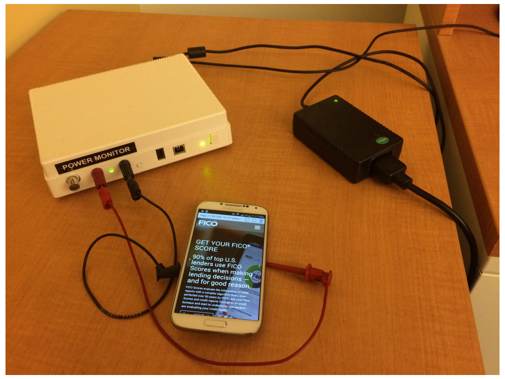




- Weighted Linear combination

$$P_s = \alpha + \sum_{i \in Resources} \beta_i R_i + \sum_{j \in C_s} \gamma_j F_j,$$

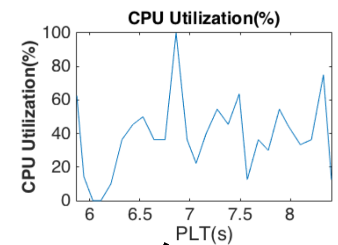
  - Specifically, for each segment
    - P_s (Average power consumption of segment *s*)
    - R_i (Resource Usage: CPU %, bytes rx/tx, …)
    - F_j (Frequency of Component: EvalHtml, …)

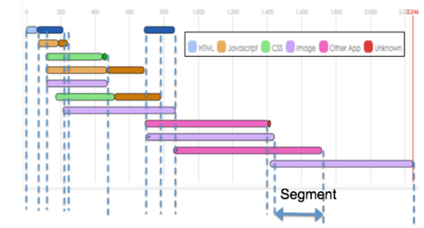


Using a power monitor to get P_s just for building the model

Linear Regression Model

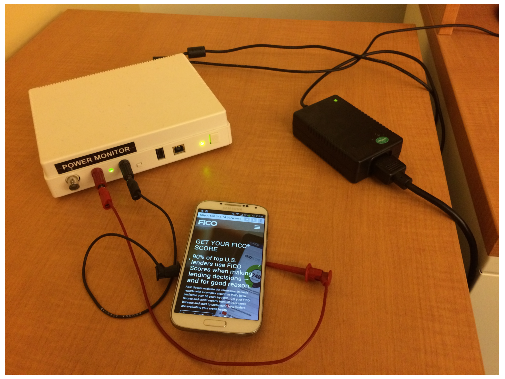
- Weighted Linear combination





$$P_s = \alpha + \sum_{i \in Resources} \beta_i R_i + \sum_{j \in C_s} \gamma_j F_j ,$$

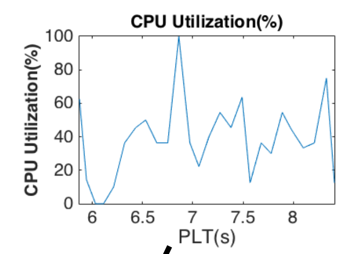
- Specifically, for each segment
  - P_s (Average power consumption of segment *s*)
  - R_i (Resource Usage: CPU %, bytes rx/tx, …)
  - F_j (Frequency of Component: EvalHtml, …)
  - $\alpha, \beta_i, \gamma_j$ (Weights)

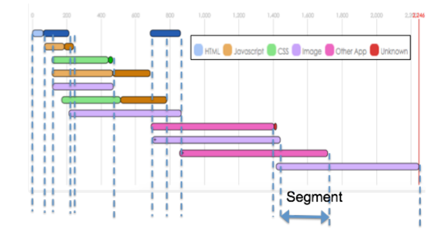


Using a power monitor to get P_s just for building the model

Linear Regression Model

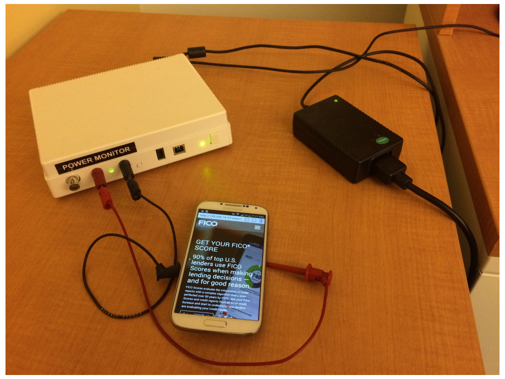
- Weighted Linear combination





$$P_s = \alpha + \sum_{i\in Resources} \beta_i R_i + \sum_{j\in C_s} \gamma_j F_j,$$

- Specifically, for each segment
  - P_s (Average power consumption of segment *s*)
  - R_i (Resource Usage: CPU %, bytes rx/tx, …)
  - F_j (Frequency of Component: EvalHtml, …)
  - $\alpha, \beta_i, \gamma_j$ (Weights)
- Measure: P_s, R_i, F_j
- To Derive *unknown* $\alpha, \beta_i, \gamma_j$:
  - Use multiple linear regression



Using a power monitor to get P_s just for building the model

Neural Network Model

- Detect non-linear relationships:

$$P_s = y_0 + \sum_{k=1}^{m} y_k \left(1 + \exp(-(x_k + \sum_{i \in Res} \theta_{k,i} R_i + \sum_{j \in C_s} \phi_{k,j} F_j)) \right)^{-1}$$

Neural Network Model

- Detect non-linear relationships:

$$P_s = y_0 + \sum_{k=1}^{m} y_k \left(1 + \exp(-(x_k + \sum_{i \in Res} \theta_{k,i} R_i + \sum_{j \in C_s} \phi_{k,j} F_j))\right)^{-1}$$

- Trade-off
  - LR: fast | simple — 2 seconds for 4-CV
  - NN: powerful | complicated, slow — 20 minutes for 1-CV
  -

Model Building — LR

- Training
  - Randomly select **80** pages, pick **60** for training
    - For each Web page, we run 10 times

$$P_s = \alpha + \sum_{i \in Resources} \beta_i R_i + \sum_{j \in C_s} \gamma_j F_j,$$

Model Building — LR

- Training
  - Randomly select **80** pages, pick **60** for training
    - For each Web page, we run 10 times
  - Monitor P_s , R_i , F_j ; derive $\alpha, \beta_i, \gamma_j$

$$P_s = \alpha + \sum_{i \in Resources} \beta_i R_i + \sum_{j \in C_s} \gamma_j F_j,$$

Model Building — LR

- Training
  - Randomly select **80** pages, pick **60** for training
    - For each Web page, we run 10 times
  - Monitor P_s , R_i , F_j ; derive $\alpha, \beta_i, \gamma_j$

$$P_s = \alpha + \sum_{i \in Resources} \beta_i R_i + \sum_{j \in C_s} \gamma_j F_j,$$

- Testing
  - Test on the remaining 20 pages
    - 10 runs per page

Model Building — LR

- Training
  - Randomly select **80** pages, pick **60** for training
    - For each Web page, we run 10 times
  - Monitor P_s , R_i , F_j ; derive $\alpha, \beta_i, \gamma_j$

$$P_s = \alpha + \sum_{i \in Resources} \beta_i R_i + \sum_{j \in C_s} \gamma_j F_j,$$

- Testing
  - Test on the remaining 20 pages
    - 10 runs per page
  - Monitor R_i, F_j; estimate $\hat{P}_s$ using weighted linear summation

Model Building — LR

- Training
  - Randomly select **80** pages, pick **60** for training
    - For each Web page, we run 10 times
  - Monitor P_s , R_i , F_j ; derive $\alpha, \beta_i, \gamma_j$

$$P_s = \alpha + \sum_{i \in Resources} \beta_i R_i + \sum_{j \in C_s} \gamma_j F_j,$$

- Testing
  - Test on the remaining 20 pages
    - 10 runs per page
  - Monitor R_i, F_j; estimate $\hat{P}_s$ using weighted linear summation

- Experiment on 3 devices:
  - Samsung Galaxy S4, S5, Nexus
  - Device-specific weights

Outline

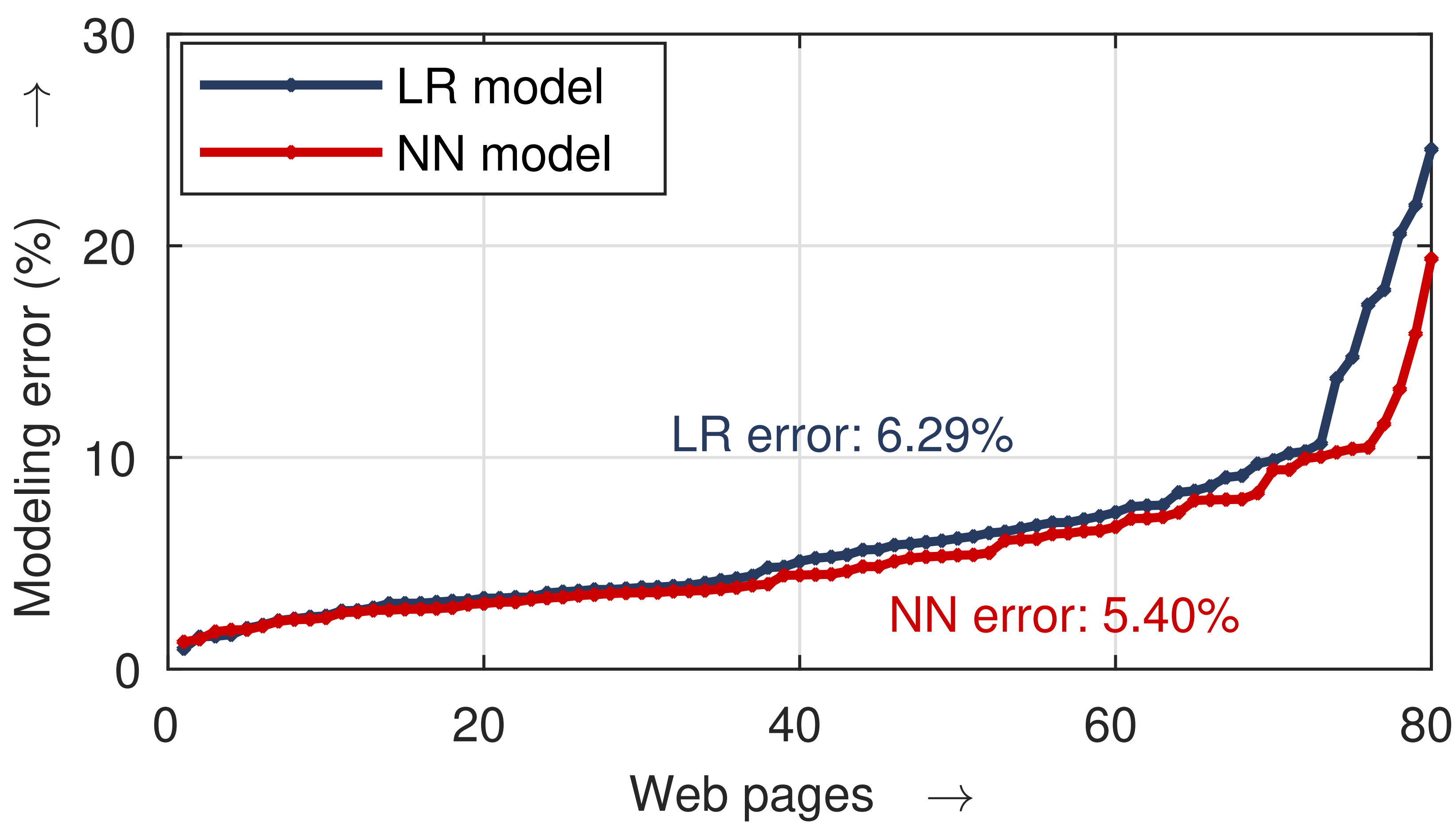
- RECON
- **Evaluation & Results**
  - **Mean Error**
  - **RECON Error CDF & Different devices**
- Application
- Conclusion

Mean Error < 7%

- Webpage-level Estimation (Galaxy S4)

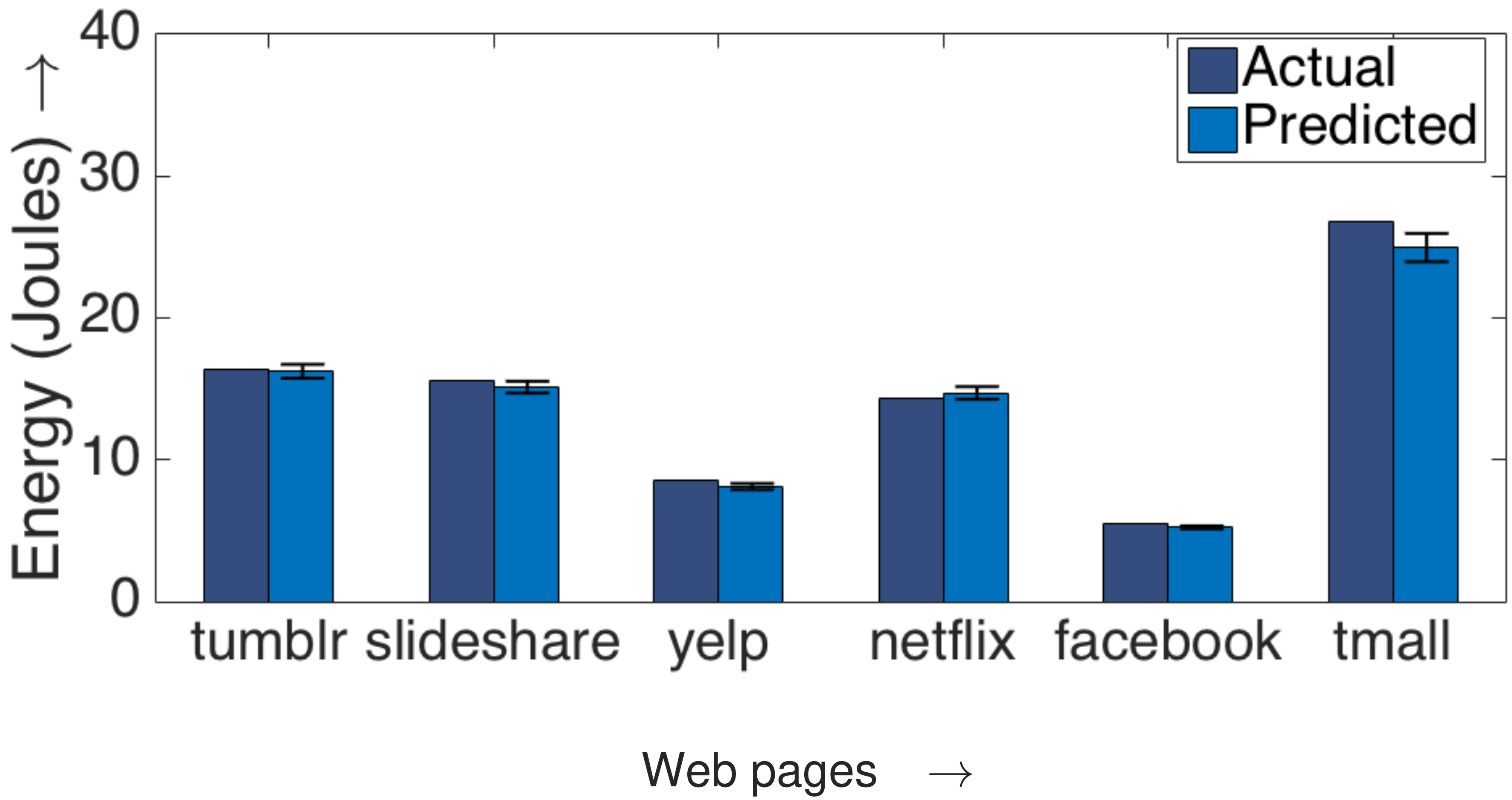
Mean Error < 7%

- Webpage-level Estimation (Galaxy S4)
  - Average estimation error 6.3% across 80 Web pages (4-fold CV)
    - NN reduces the error to 5.4%.



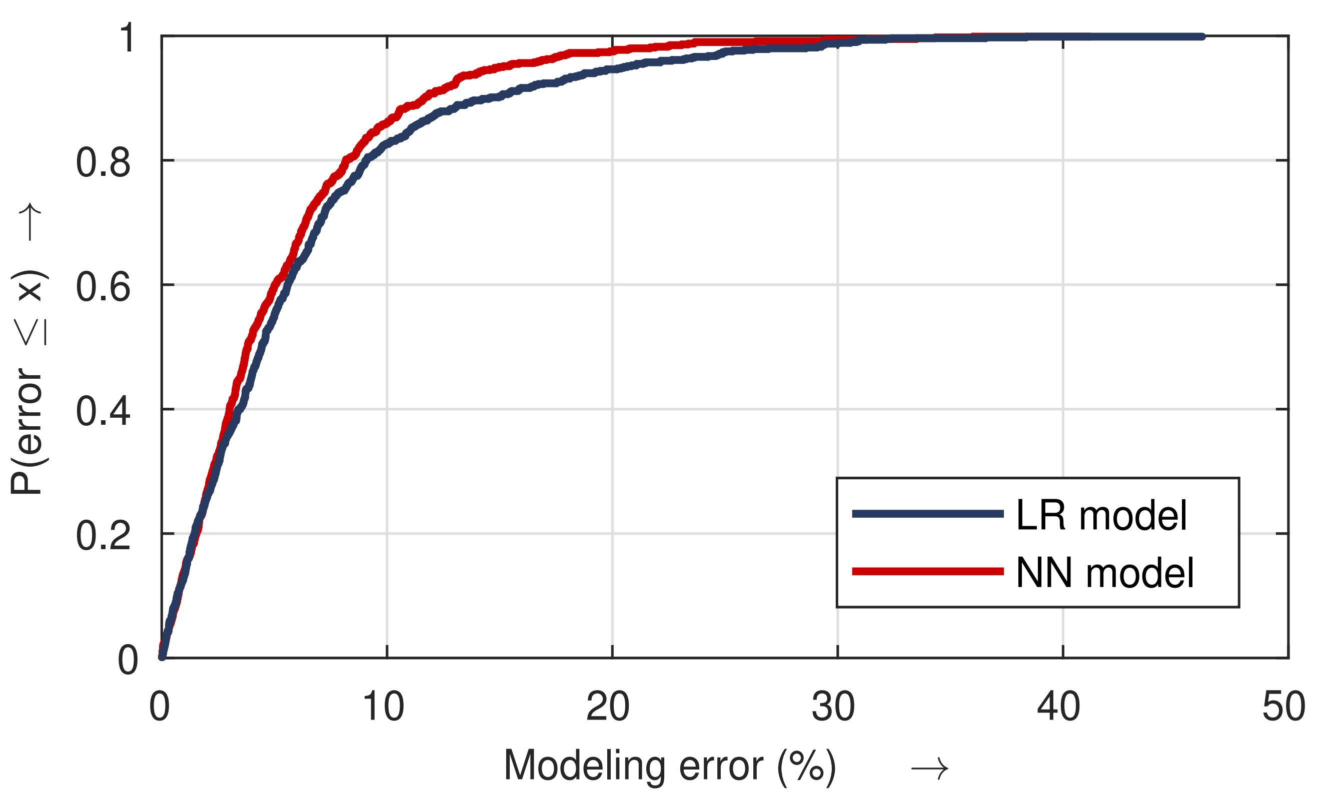
Mean Error < 7%

- Webpage-level Estimation (Galaxy S4)
  - Average estimation error 6.3% across 80 Web pages (4-fold CV)
    - NN reduces the error to 5.4%.



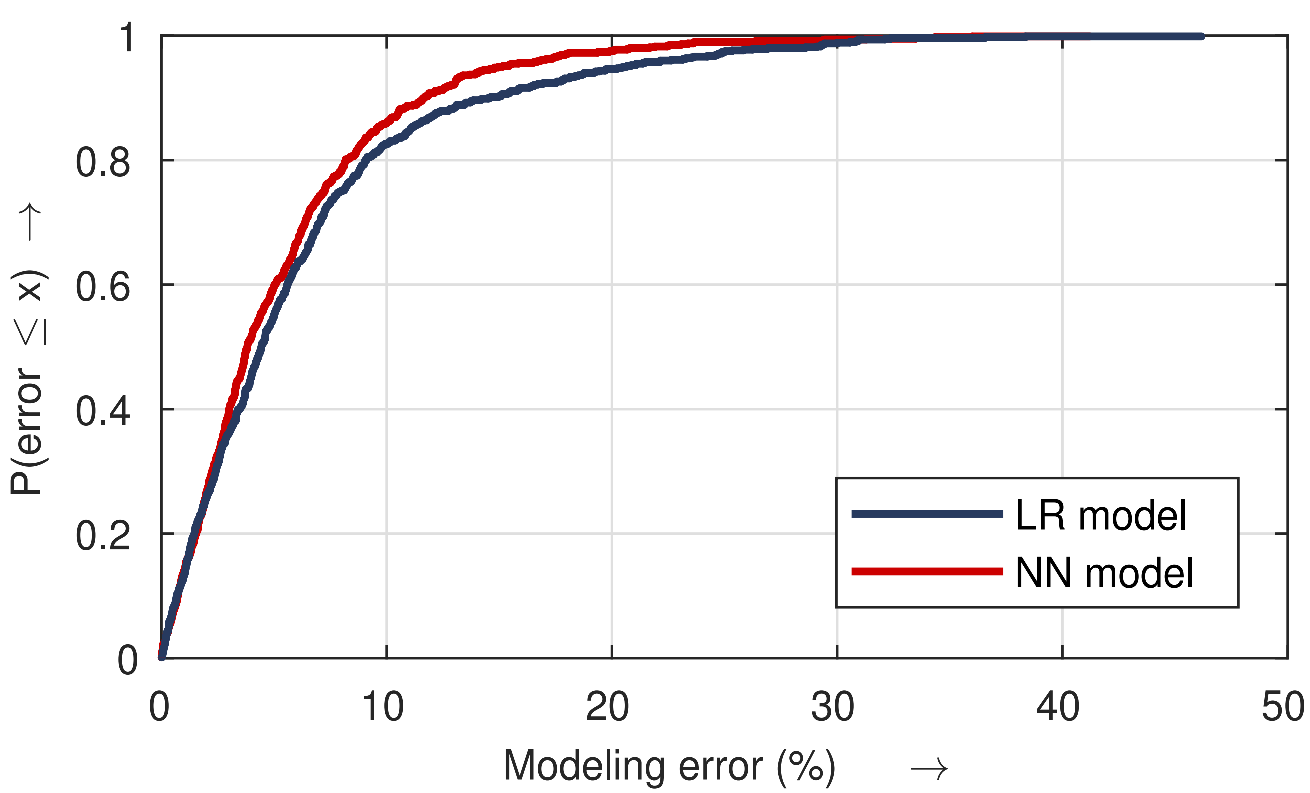
Error CDF

- RECON Error CDF



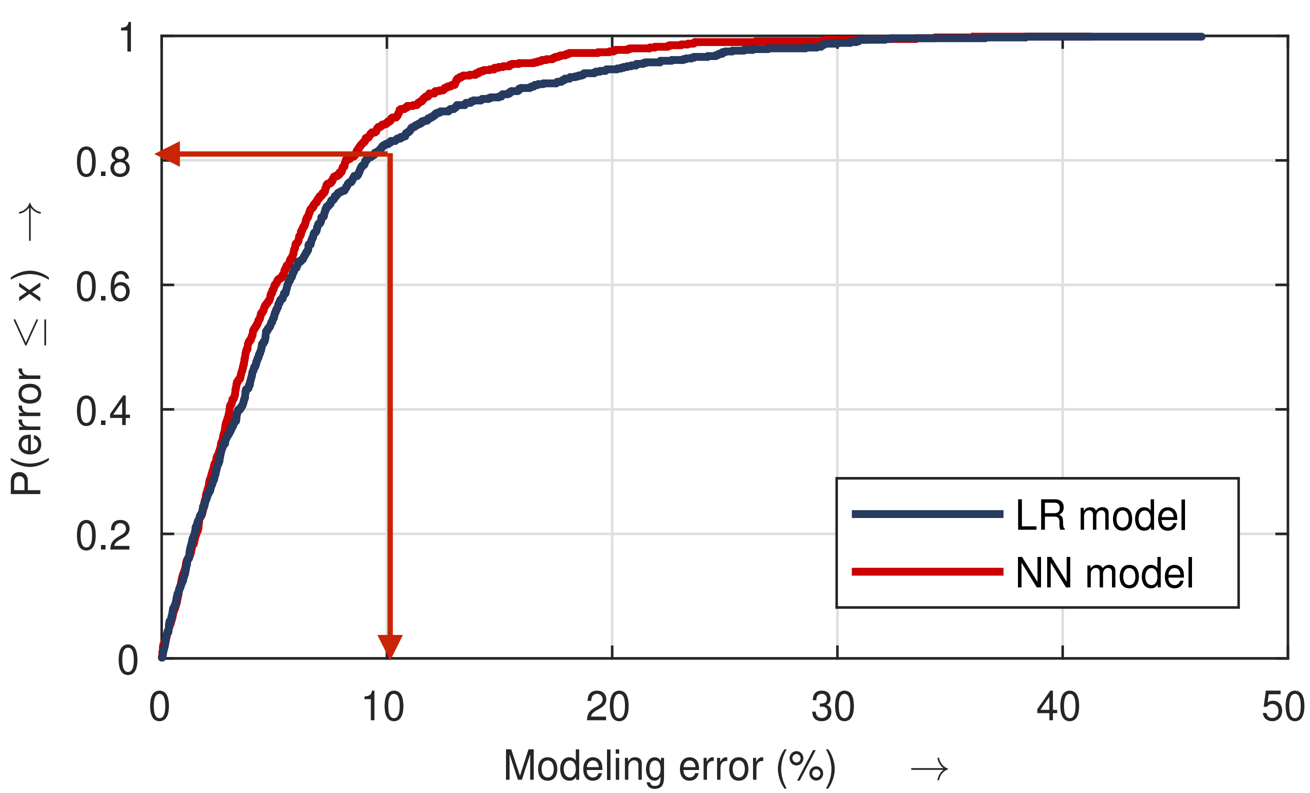
Error CDF

- RECON Error CDF
  - The CDF shows the energy estimation errors across all runs of all 80 Web pages. We see that 80% of the errors are below 10%.



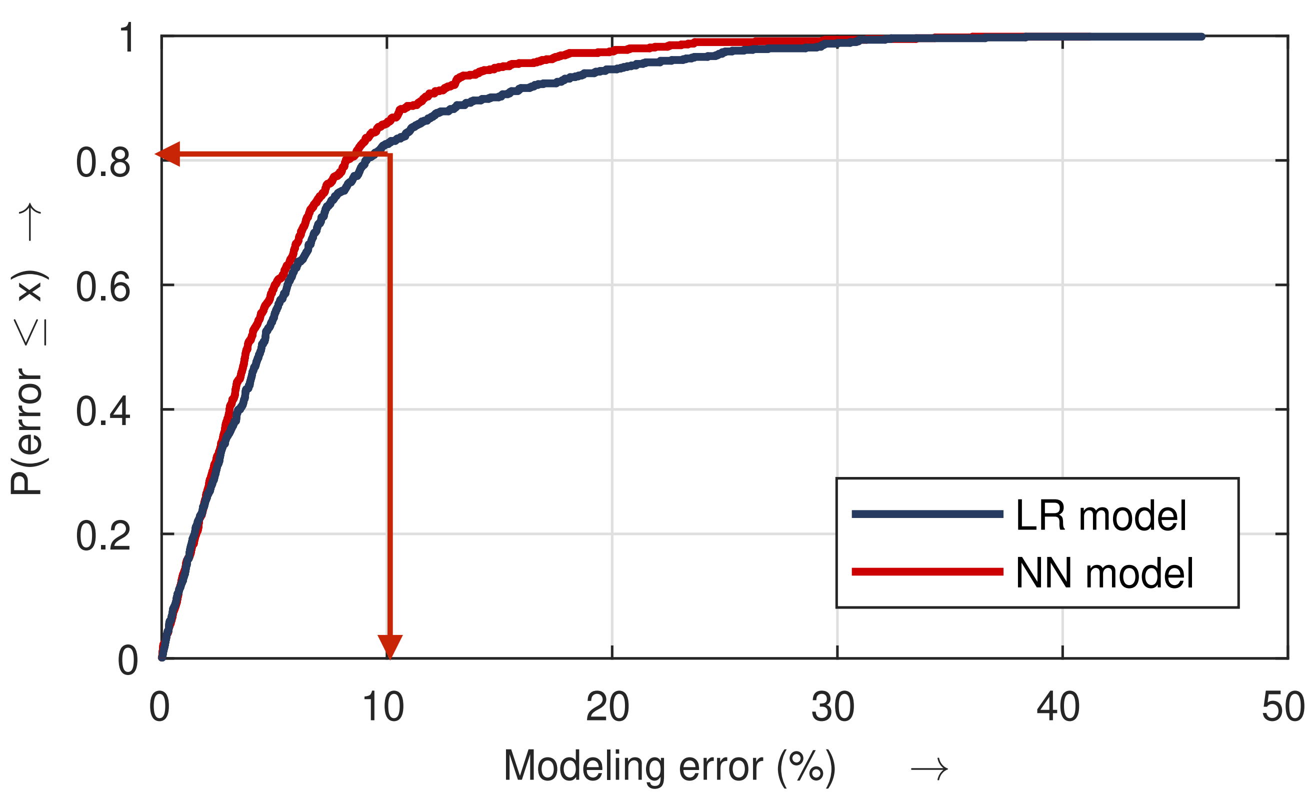
Error CDF

- RECON Error CDF
  - The CDF shows the energy estimation errors across all runs of all 80 Web pages. We see that 80% of the errors are below 10%.



Error CDF

- RECON Error CDF
  - The CDF shows the energy estimation errors across all runs of all 80 Web pages. We see that 80% of the errors are below 10%.

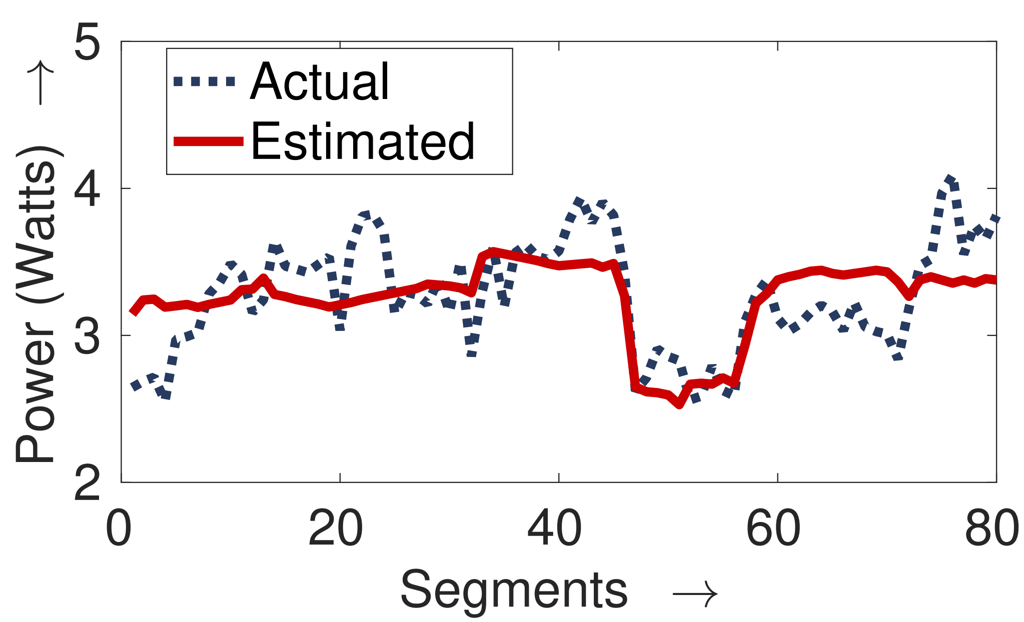


Different Devices

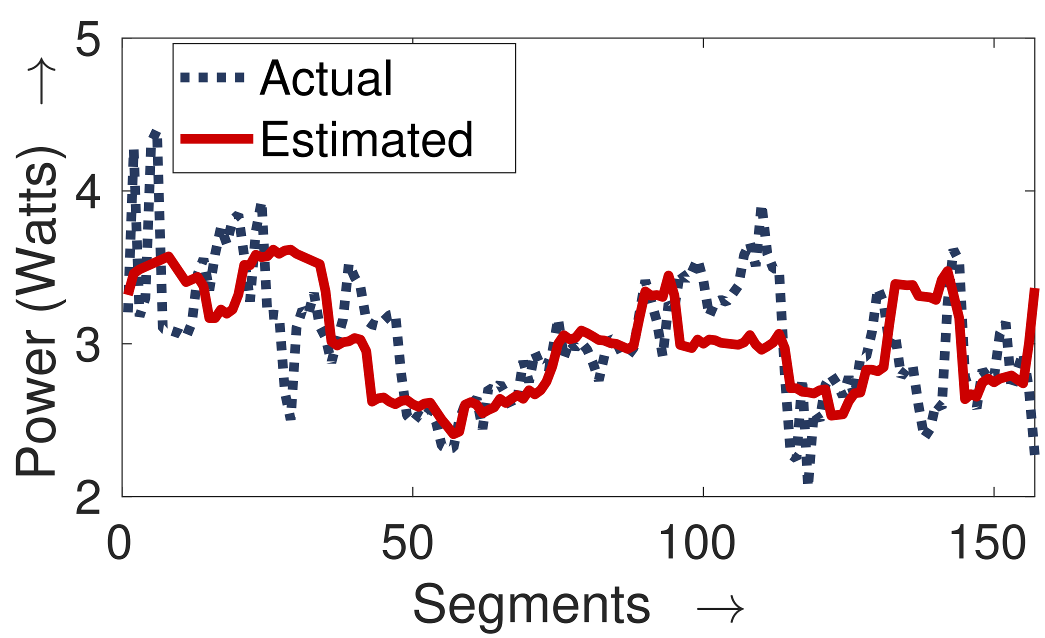
Error	S4	S5	Nexus
Webpage	6.3%	7.1%	9.1%

Segment Error

- **Fine-grained** power estimation
  - Based on segments



Segment error 7.8% for yelp.com



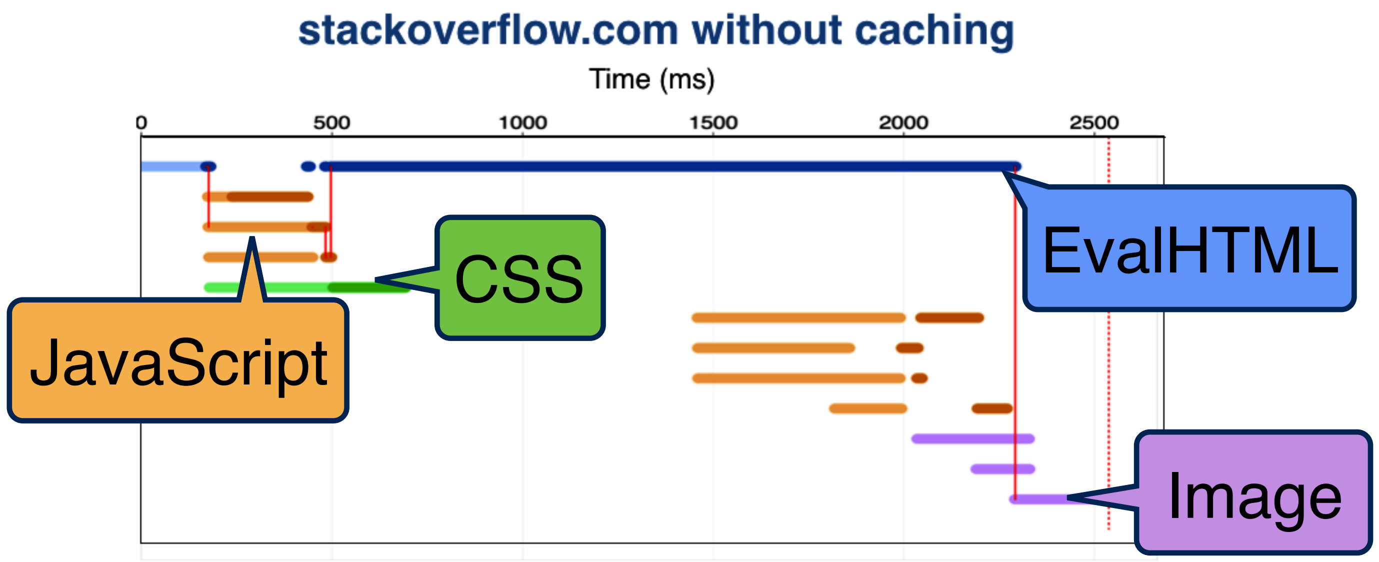
Segment error 9.7% for sfr.fr

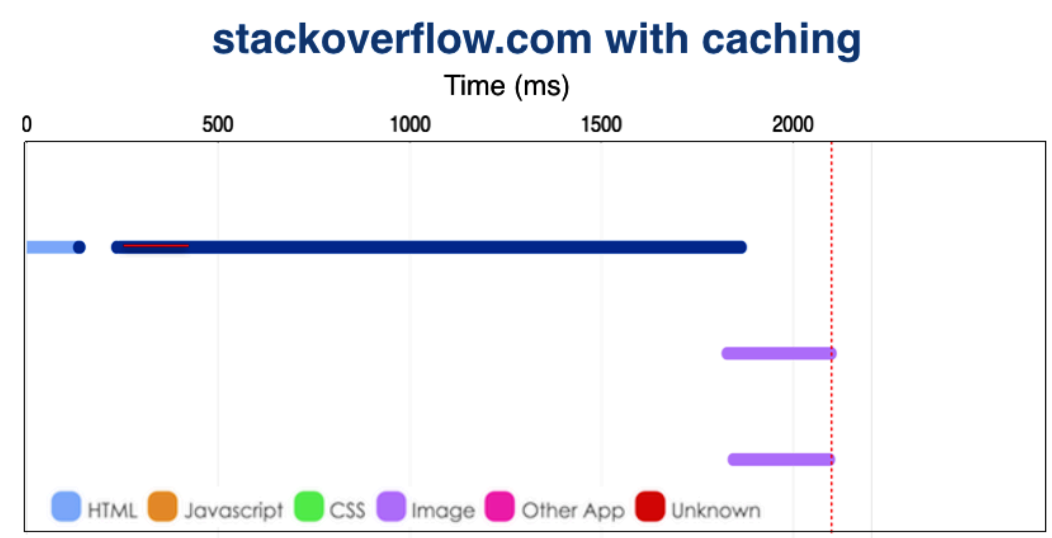
Outline

- RECON
- Evaluation & Results
- **Application**
  - Analyze Web enhancements' non-intuitive energy behaviors
  - Two case studies
    - Caching
    - Compression
- Conclusion

Case 1: Caching

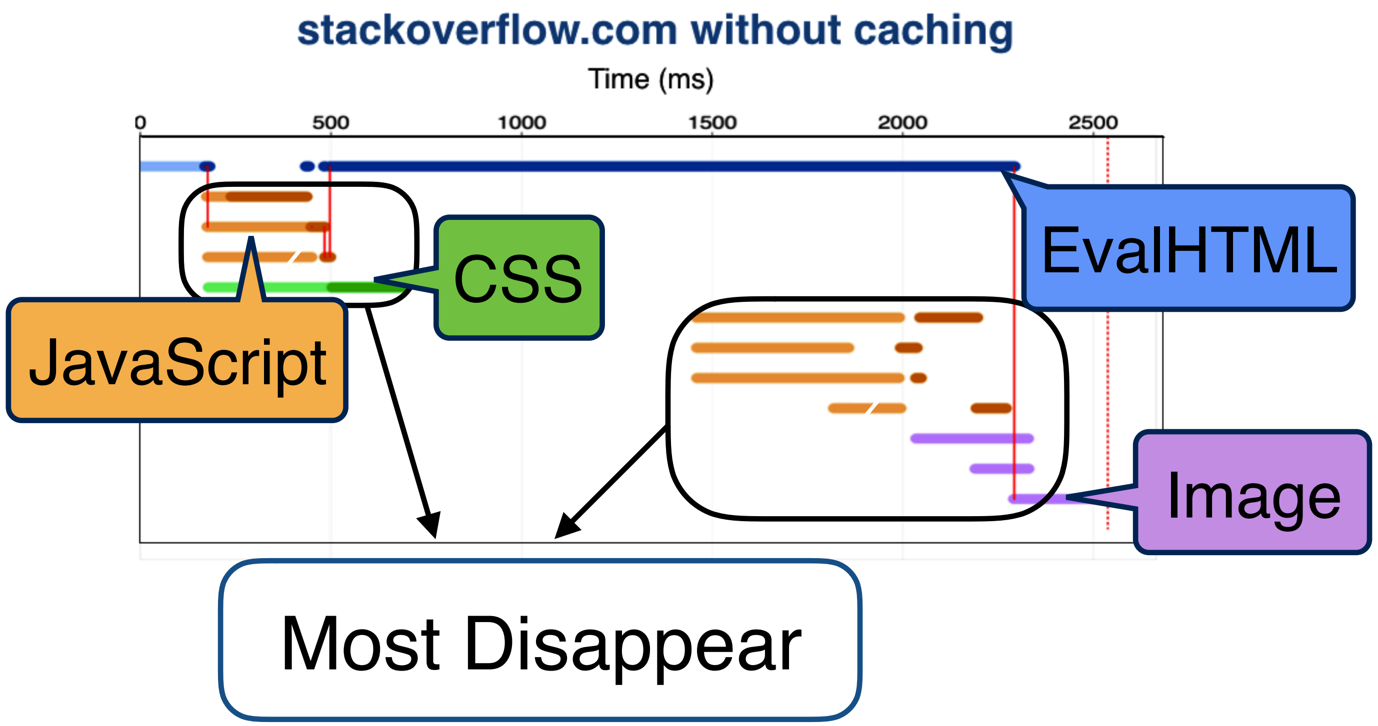
- How will PLT and Energy change due to caching?

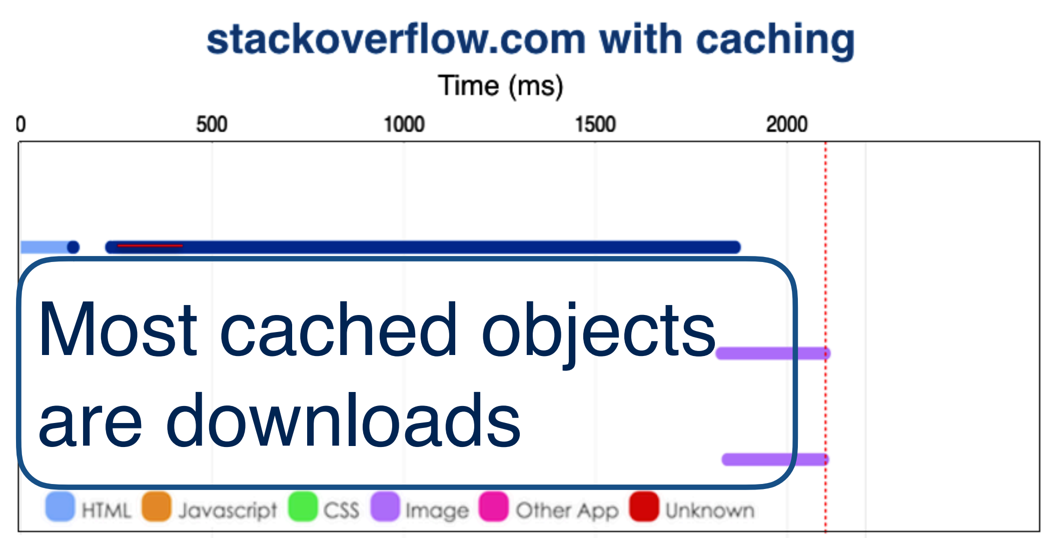




Case 1: Caching

- How will PLT and Energy change due to caching?



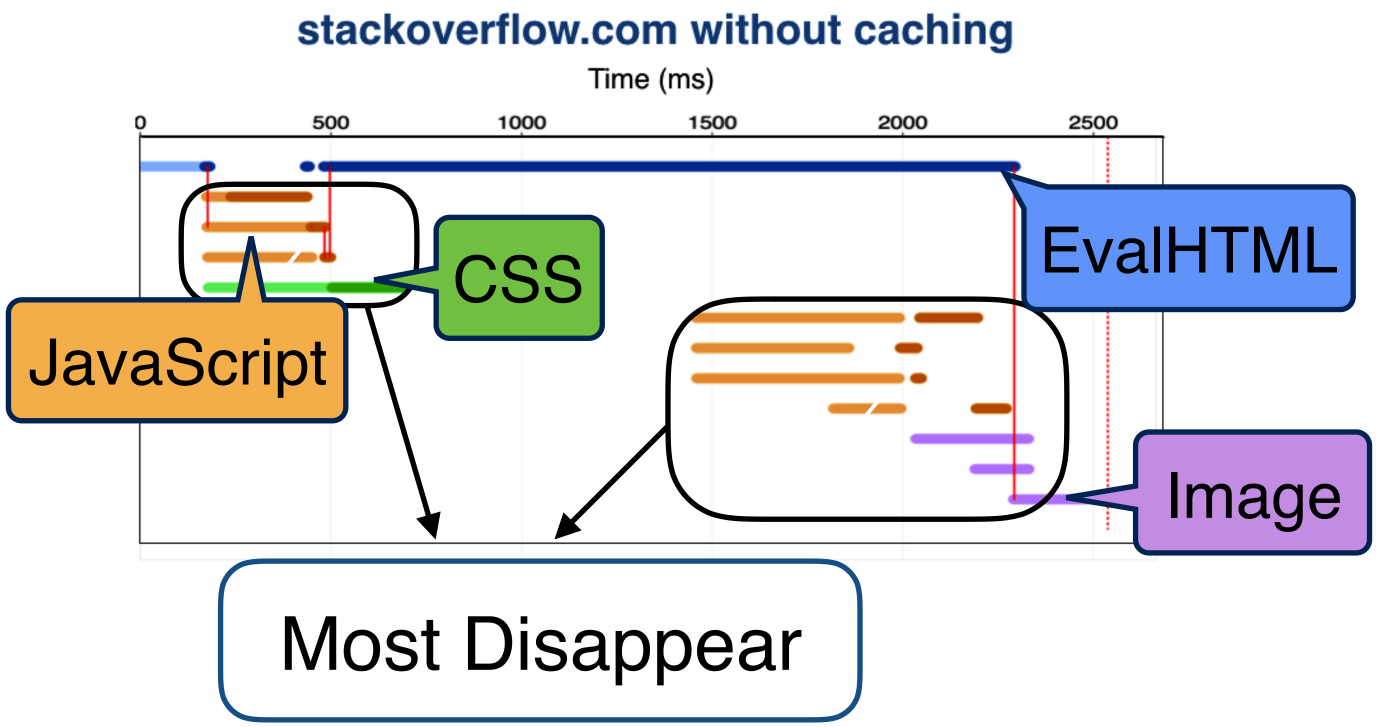


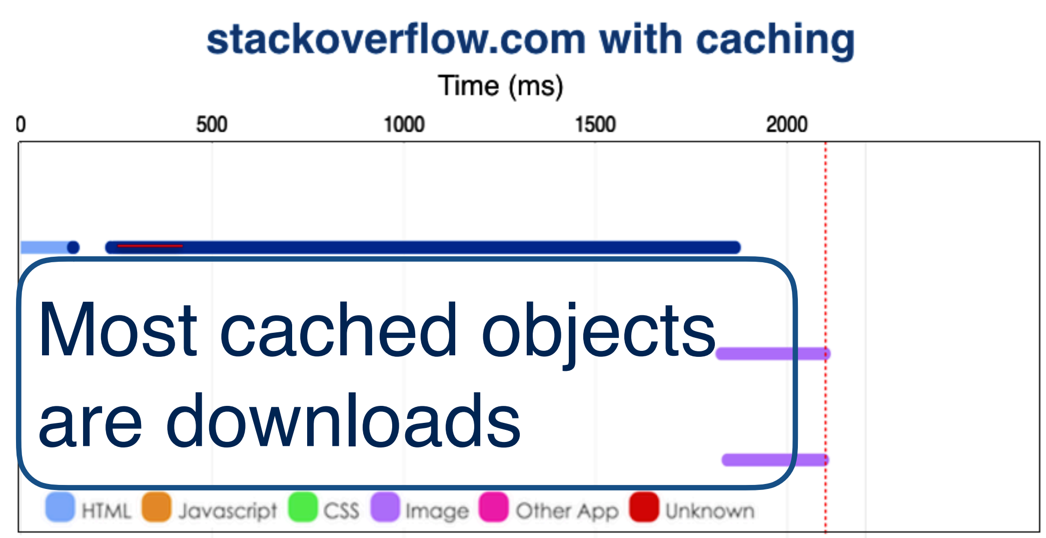
Case 1: Caching

- How will PLT and Energy change due to caching?

	PLT(s)	Energy(J)
Original	2.5	8.2
Cached	2.1	5.7
Reduce%	16%	30%

Energy Reduction ~= 2X PLT Reduction



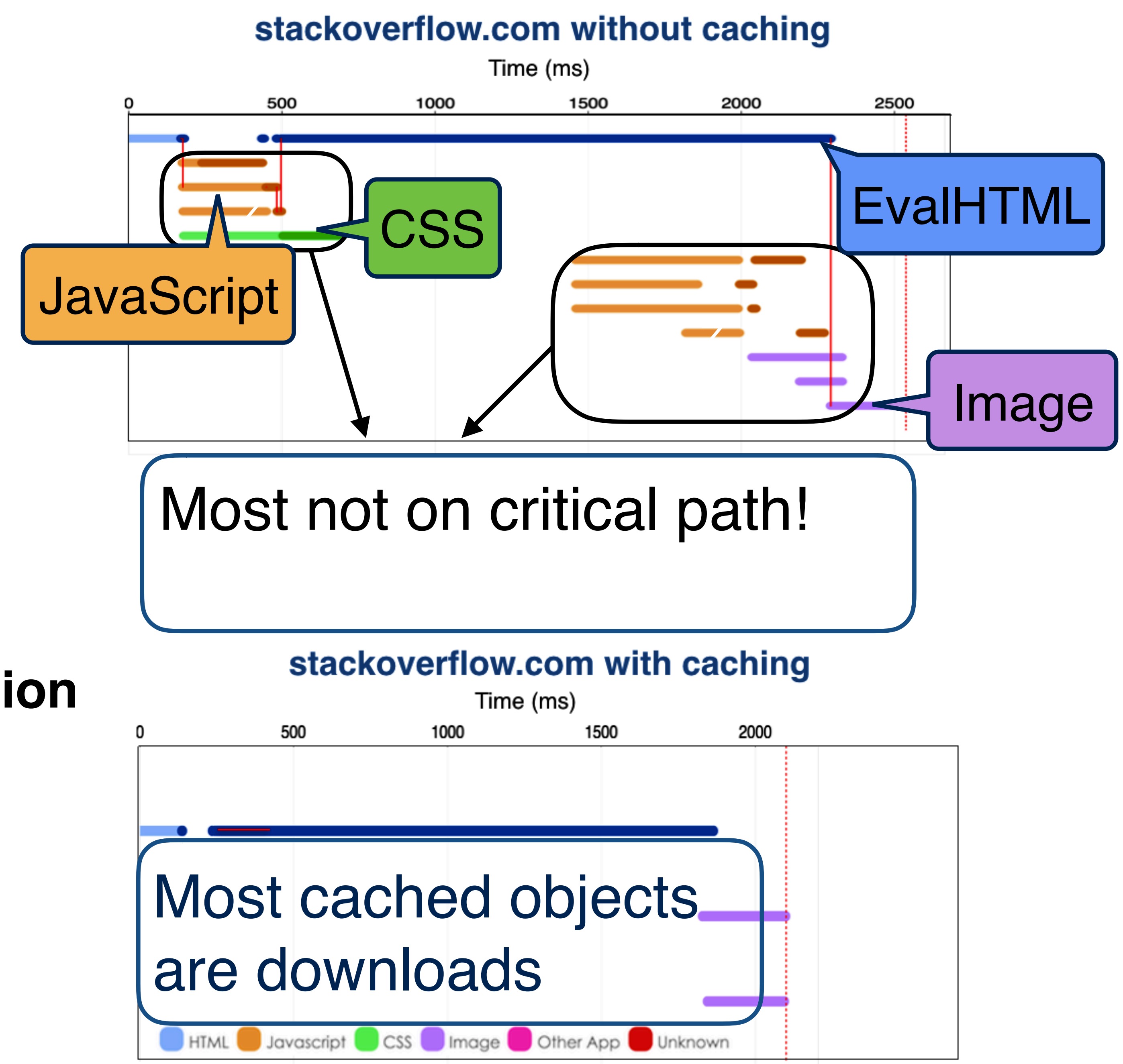


Case 1: Caching

- How will PLT and Energy change due to caching?

	PLT(s)	Energy(J)
Original	2.5	8.2
Cached	2.1	5.7
Reduce%	16%	30%

Energy Reduction ~= 2X PLT Reduction

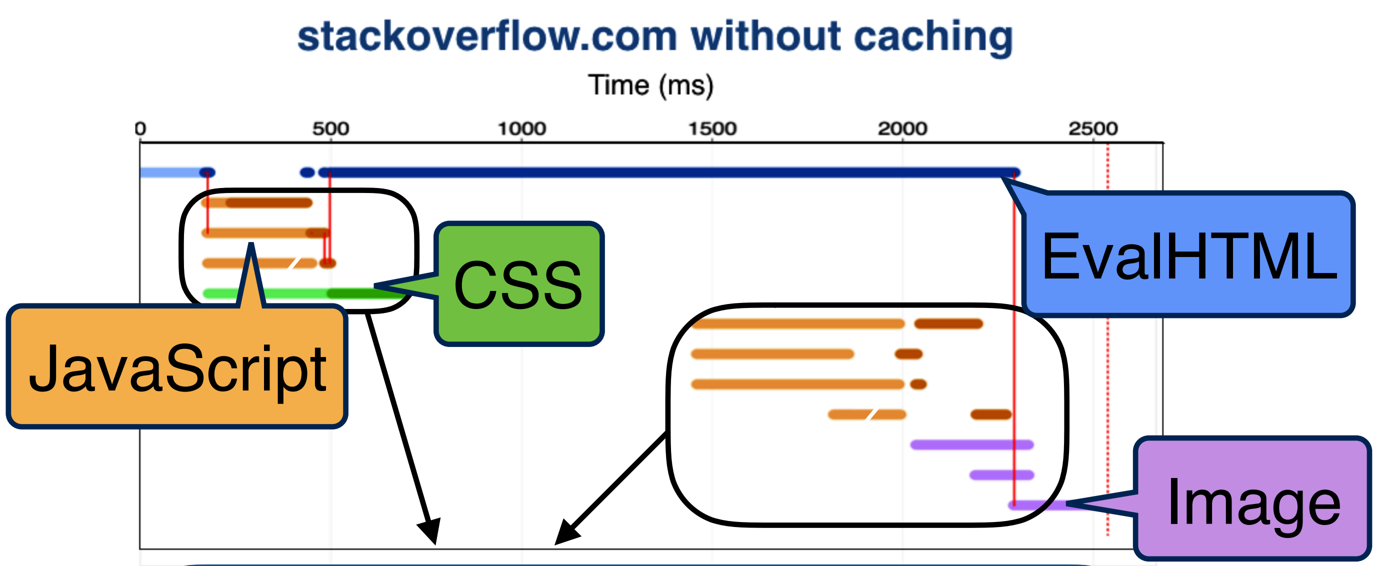


Case 1: Caching

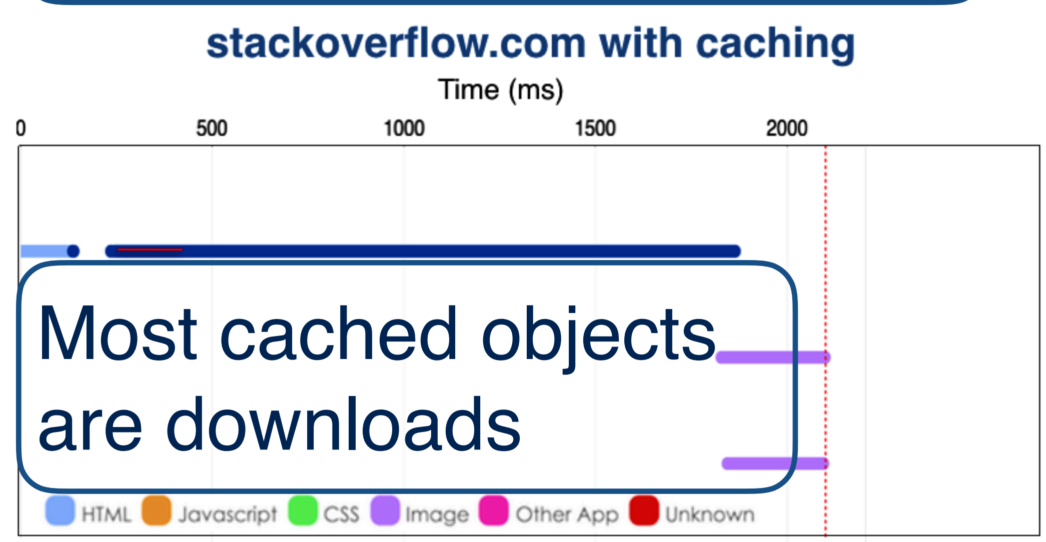
- How will PLT and Energy change due to caching?

	PLT(s)	Energy(J)
Original	2.5	8.2
Cached	2.1	5.7
Reduce%	16%	30%

Energy Reduction ~= 2X PLT Reduction



Most not on critical path!
But, they affect energy.



Most cached objects are downloads

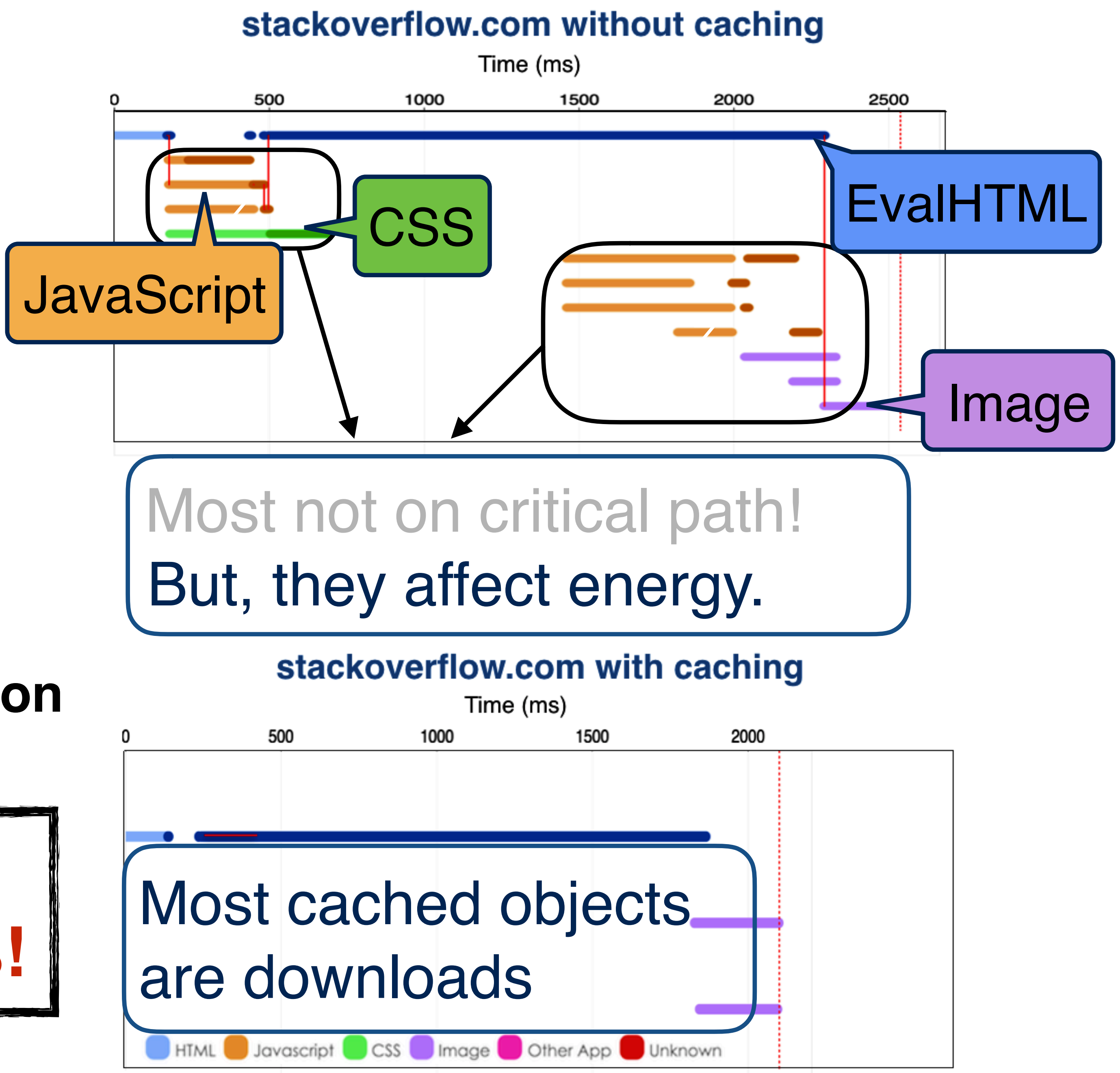
Case 1: Caching

- How will PLT and Energy change due to caching?

	PLT(s)	Energy(J)
Original	2.5	8.2
Cached	2.1	5.7
Reduce%	16%	30%

Energy Reduction ~= 2X PLT Reduction

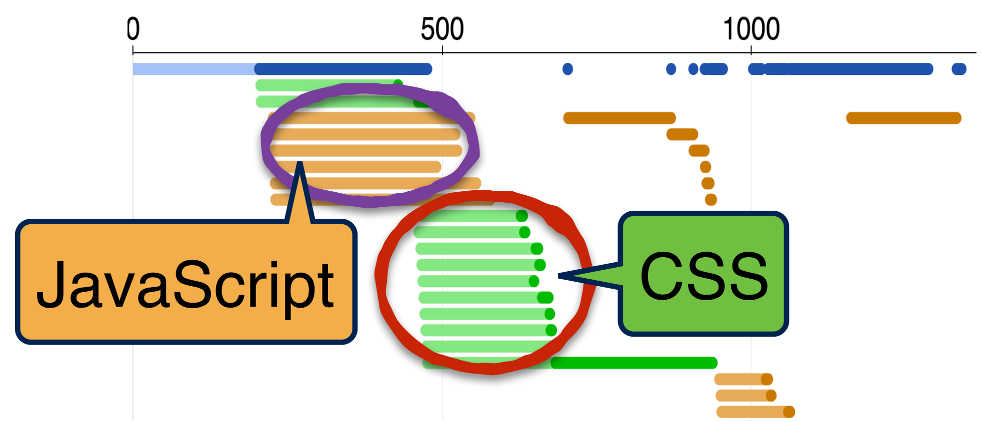
RECON: Energy for Downloads reduces by **81%!**



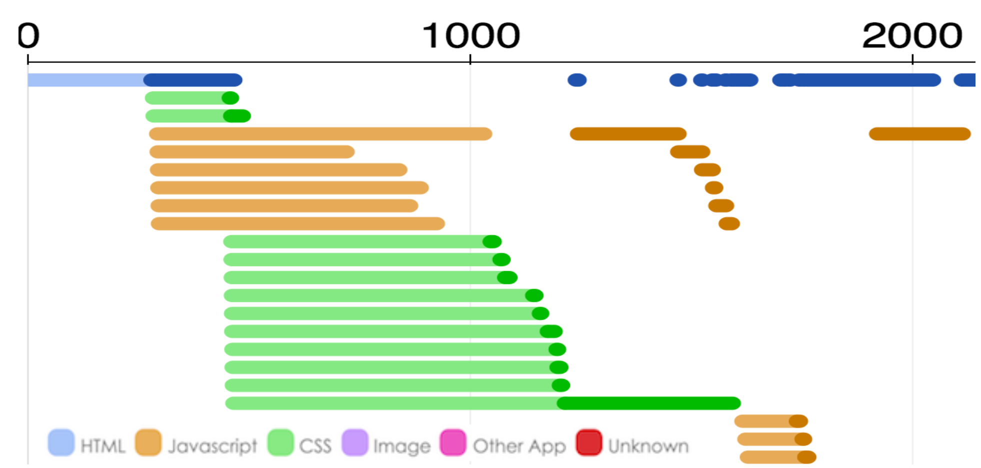
Case 2: Gzip Compression

- Compression level ranges from 1 to 9 (NGINX)
  - lv.9 is the highest compression level

irs.gov under compression level 1



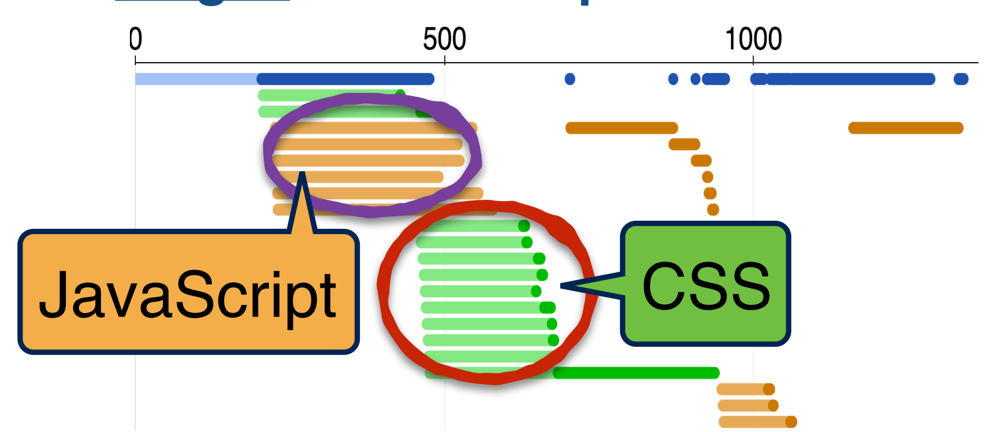
irs.gov under compression level 9



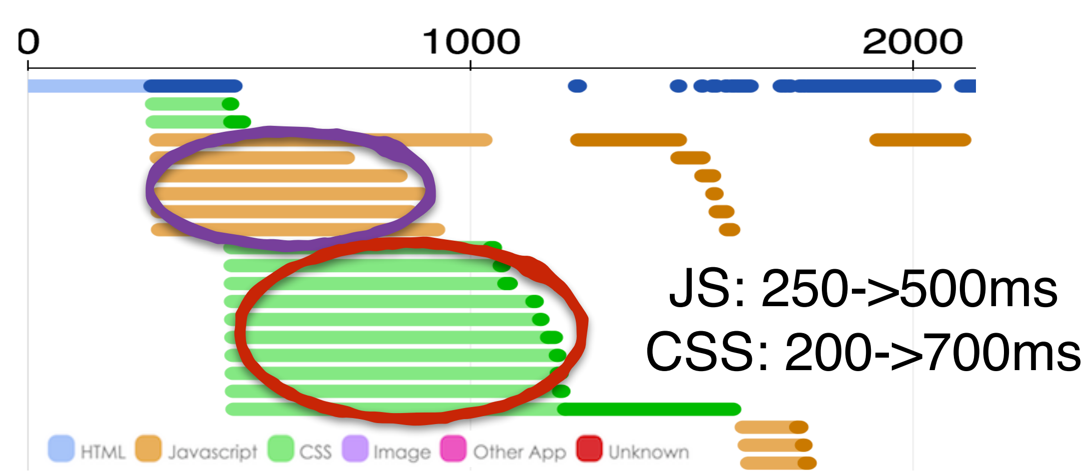
Case 2: Gzip Compression

- Compression level ranges from 1 to 9 (NGINX)
  - lv.9 is the highest compression level

irs.gov under compression level 1



irs.gov under compression level 9



Case 2: Gzip Compression

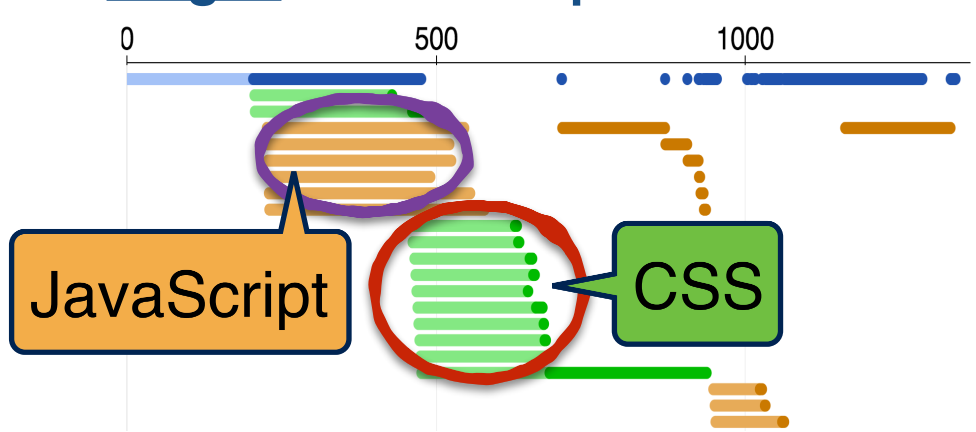
- Compression level ranges from 1 to 9 (NGINX)
  - lv.9 is the highest compression level

	PLT ↓	Energy ↓
Level 1	78%	75%
Level 9	47%	39%

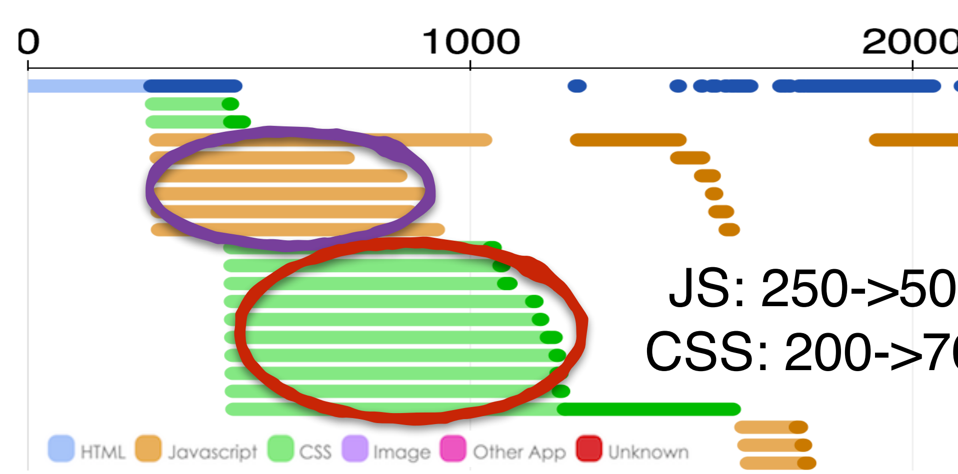
Better!

- **Lower compression level** provides more benefits!

irs.gov under compression level 1



irs.gov under compression level 9



JS: 250->500ms
CSS: 200->700ms

Case 2: Gzip Compression

- Compression level ranges from 1 to 9 (NGINX)
  - lv.9 is the highest compression level

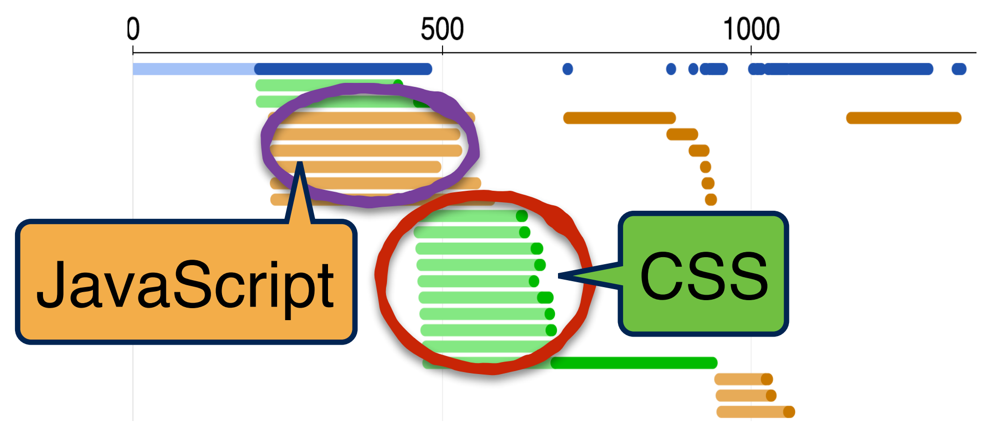
	PLT ↓	Energy ↓
Level 1	78%	75%
Level 9	47%	39%

Better!

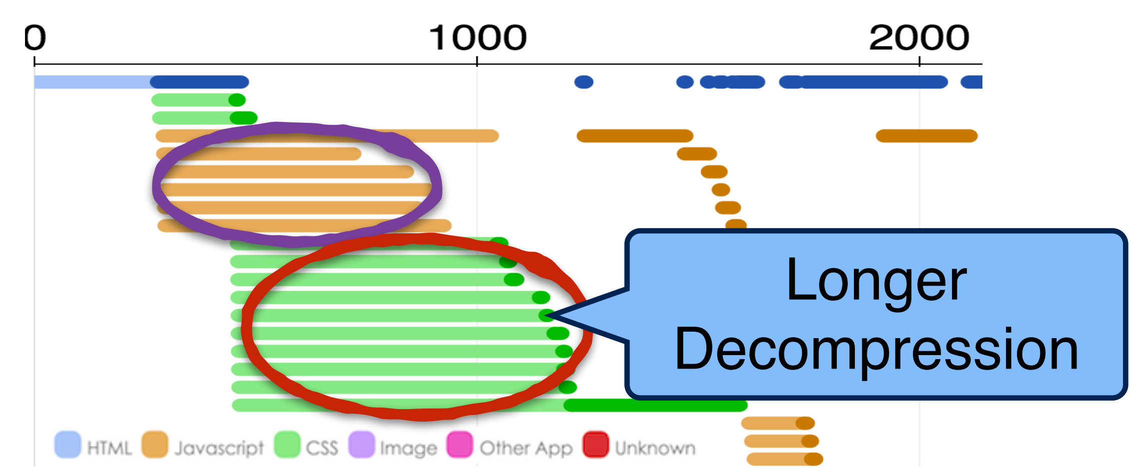
  - **Lower compression level** provides more benefits!

RECON: 37% more CPU energy due to CSS and Javascript decompression

irs.gov under compression level 1



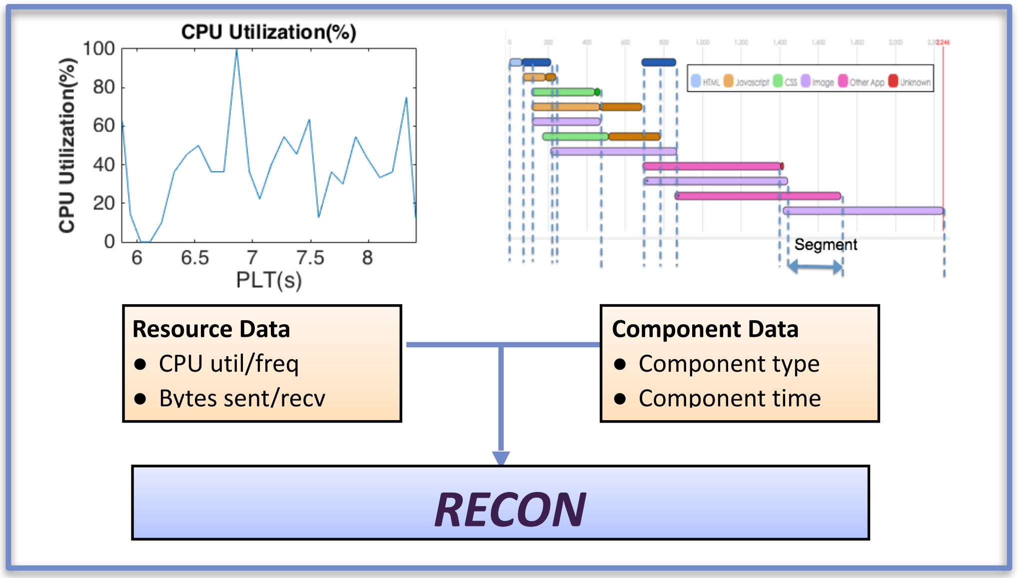
irs.gov under compression level 9



Outline

- RECON
- Evaluation & Results
- Application
- **Conclusion**

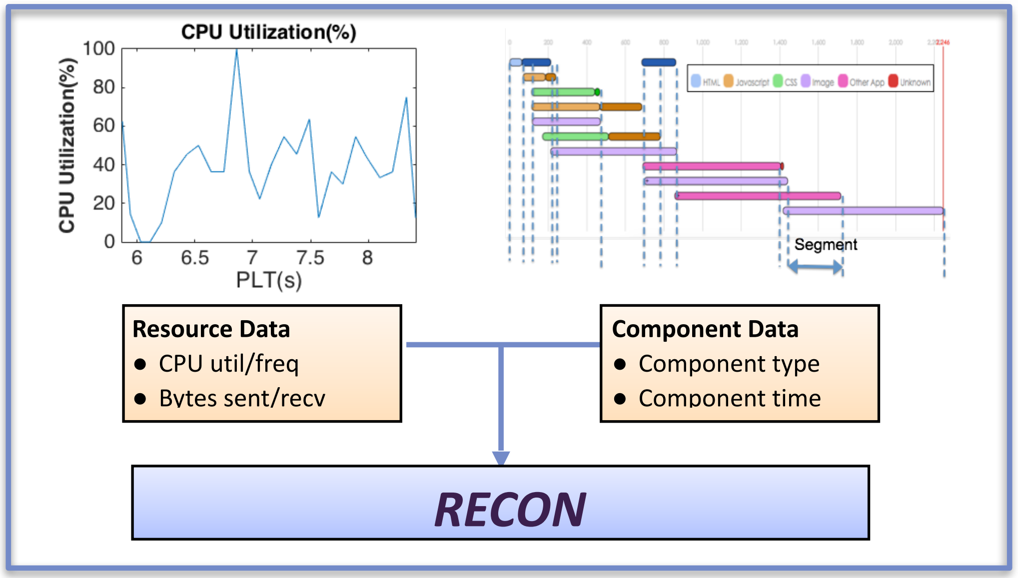
Conclusion



- Web performance critical
  - Overlook energy
  - Mobile devices are constrained by energy
- We present RECON
  - Leverages page load semantics and resource-level information
  - Less than 7% error across 80 webpages.
  - Enables evaluating the energy effects of Web optimizations

Stony Brook University

Conclusion



- Web performance critical
  - Overlook energy
  - Mobile devices are constrained by energy
- We present RECON
  - Leverages page load semantics and resource-level information
  - Less than 7% error across 80 webpages.
  - Enables evaluating the energy effects of Web optimizations
- **Thank you!**