Adapting Swarm Application

A Systematic and Quantitative Approach

Ben Zhang

December 6, 2019

TerraSwarm Research Center

https://github.com/nebgnahz/dissertation-talk

Table of contents

Introduction to the Swarm

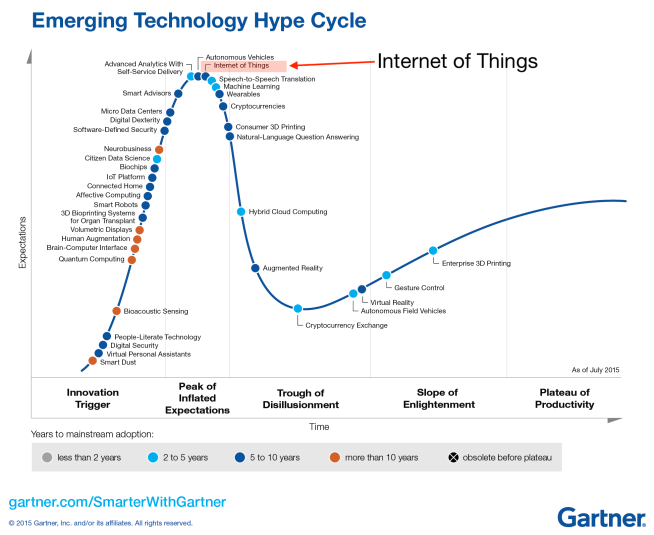
The Swarm at the Edge of the Cloud

J. Rabaey, ASPDAC'08

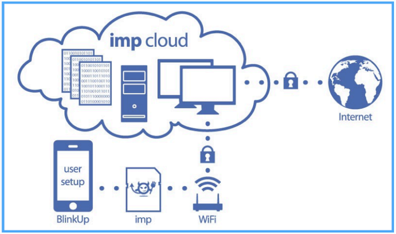
Swarm, or

- Internet of Things (IoT)
- Internet of Everything (IoE)
- Industry 4.0
- The Industrial Internet
- TSensors (Trillion Sensors)
- Machine to Machine (M2M)
- Smarter Planet

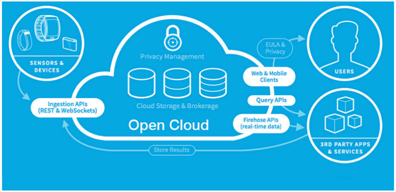
Gartner Hype Cycle (2015)



A Cloud-centric Approach



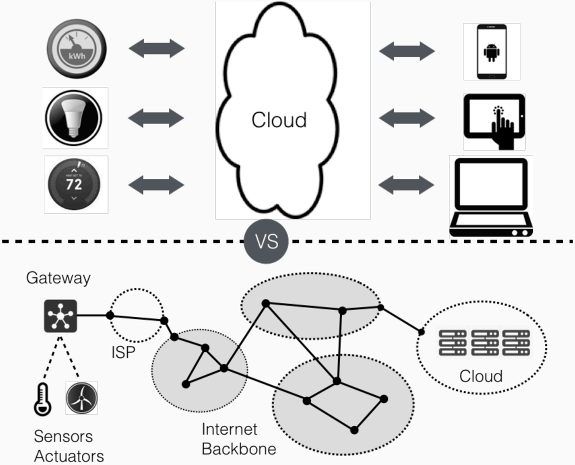
(a) Electric Imp



(b) Samsung SAMI

(c) Ninja Sphere

“The Cloud”: Model vs. Reality



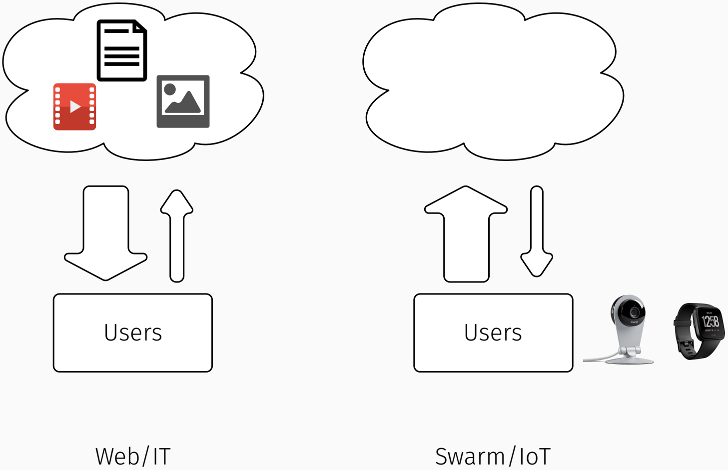
The Cloud is Not Enough

	Web/IT	Swarm/IoT
Privacy & Security	Open for access	Sensitive data
Scalability	Power law	Billion devices
Interaction Model	Human	Machine
Latency	Variable	Bounded
Bandwidth	Downstream	Upstream
Availability (QoS)	No guarantee	Requirement
Durability Management	Cloud controls	Users control

Pitfalls with Today's Approach to IoT [Zhang et al., 2015]

	Web/IT	Swarm/IoT
Privacy & Security	Open for access	Sensitive data
Scalability	Power law	Billion devices
Interaction Model	Human	Machine
Latency	Variable	Bounded

Bandwidth: Downstream vs. Upstream



Network Resource Adaptation

Limited Network Resource

Demand
Huge Data Volume at the Edge

Resource
Insufficient WAN Bandwidth

Demand
Huge Data Volume at the Edge

Resource
Insufficient WAN Bandwidth

Demand
Huge Data Volume at the Edge

Resource
Insufficient WAN Bandwidth

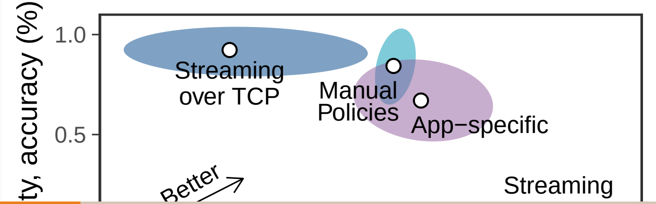
- Video surveillance, 3 mbps per camera [Amerasinghe, 2009]
- Electrical grid monitoring, 1.4 million data points per second [Andersen and Culler, 2016]
- Machine logs, 25 TB daily at Facebook (2009)

. . . Dropcam, a WiFi video-streaming camera and associated cloud backend service for storing and watching the resulting video. Dropcam has the fewest clients (2,940) . . . . Yet, each client uses roughly 2.8 GB a

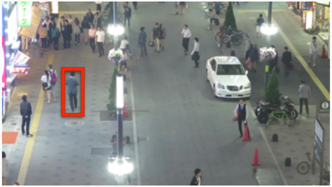
Fidelity vs. Freshness

When the network resource is not sufficient:

- TCP ensures data delivery, but hurts latency
- UDP sends as fast as possible, uncontrolled packet loss
- Manual policies (developer heuristics) are sub-optimal
  - JetStream [Rabkin et al., 2014] uses manual policy
  - "if bandwidth is insufficient, switch to sending images at 75% fidelity, then 50% if still not enough"
- Application-specific optimizations don't generalize
  - Video streaming often aims at Quality of Experience (limited degradation dimension, e.g. maintain 25FPS)
  - For object detection, resolution matters more than FPS



Application-specific Optimizations Don't Generalize

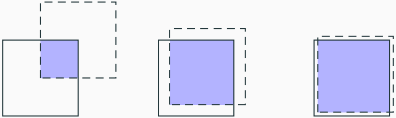


t=0s, small target in far-field views

t=1s, small difference

Positive if intersection over union (IOU) larger than 0.5.

$$\text{IOU} = \frac{\text{Area of Intersection}}{\text{Area of Union}}$$

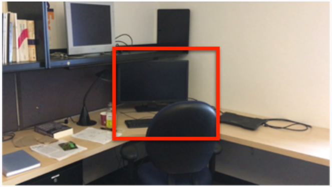


(a) IOU=0.14 (b) IOU=0.57 (c) IOU=0.82

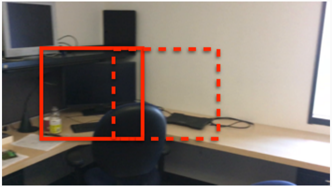
F1 score is the harmonic mean of precision and recall, ranging from 0 to 1:

	P	N
Y	True Positive	False Positive
N	False Negative	True Negative

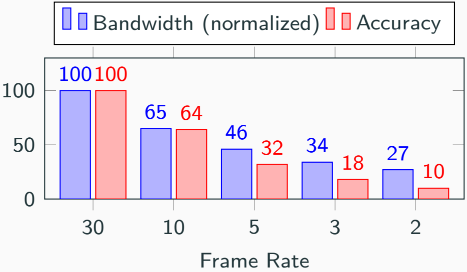
Application-specific Optimizations Don't Generalize

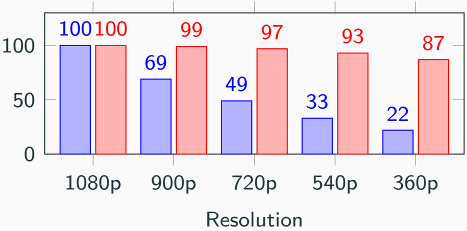


t=0s, nearby and large targets



t=1s, large difference





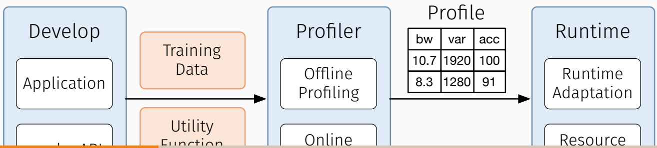
Making Adaptation Practical is Challenging

Goal

Minimize bandwidth while maximizing application accuracy

Challenges:

1. Application-specific optimizations don't generalize.
   - APIs: `maybe` operators to express adaptation.
2. It requires expertise and manual work to explore multidimensional adaptation.
   - Profiling: automatically learn Pareto-optimal strategy with multi-dimensional exploration.
3. The adaptation happens at the runtime.
   - Engineering an adaptation system to balance latency and accuracy.



(1) Stream Processing APIs

Data Stream $\longrightarrow$ Operator $\longrightarrow$ Data Stream

$\{x_1, x_2, x_3, x_4, \ldots\} \longrightarrow$ map(f) $\longrightarrow \{f(x_1), f(x_2), f(x_3), f(x_4), \ldots\}$

$\{x_1, x_2, x_3, x_4, \ldots\} \longrightarrow$ window(2, f) $\longrightarrow \{f(x_1, x_2), f(x_3, x_4), \ldots\}$

$\{x_1, x_2, x_3, x_4, \ldots\} \longrightarrow$ maybe($\vec{k}$, f) $\longrightarrow \{f(x_1, k_{i_1}), f(x_2, k_{i_2}), f(x_3, k_{i_3}), f(x_4, k_{i_4}), \ldots\}$

Normal	*map* (f: I ⇒ O)	Stream<I> ⇒ Stream<O>
	skip (i: Integer)	Stream<I> ⇒ Stream<I>
	window (count: Integer, f: Vec<I> ⇒ O)	Stream<I> ⇒ Stream<O>
	...	...
Adaptation	*maybe* (knobs: Vec<T>, f: (T, I) ⇒ I)	Stream<I> ⇒ Stream<I>
	maybe_skip (knobs: Vec<Integer>)	Stream<I> ⇒ Stream<I>
	maybe_head (knobs: Vec<Integer>)	Stream<Vec<I>> ⇒ Stream<Vec<I>>
	...	...

maybe(knobs: Vec<T>, f: (T, I) => I)

```
let quantized_stream = vec![1, 2, 3, 4].into_stream()
    .maybe(vec![2, 4], |k, val| val / k)
    .collect();
```

no degradation	k = 2	k = 4
[1, 2, 3, 4]	[0, 1, 1, 2]	[0, 0, 0, 1]

We rewrite the video streaming application as follows,

```
let app = Camera::new((1920, 1080), 30)
    .maybe_downsample(vec![(1600, 900), (1280, 720)])
    .maybe_skip(vec![2, 5])
    .map(|frame| pedestrian_detect(frame))
    .compose();
```

Example code in Rust, simplified for presentation.

(2) Profiling

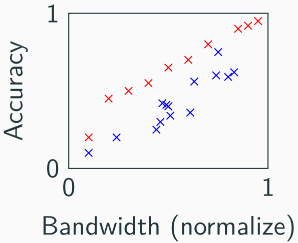
```
let app = Camera::new((1920, 1080), 30)
    .maybe_downsample(vec![(1600, 900), (1280, 720)])
    .maybe_skip(vec![2, 5])
    .map(|frame| pedestrian_detect(frame))
    .compose();
```

Training Data

Accuracy Function

downsample	skip	bandwidth	accuracy
(1920, 1080)	0	10.7	1.0
(1600, 900)	0	8.3	0.88
(1280, 720)	0	6.3	0.87
(1920, 1080)	2	9.3	0.90
...	...	...	...

Profile: Pareto-optimal Strategy

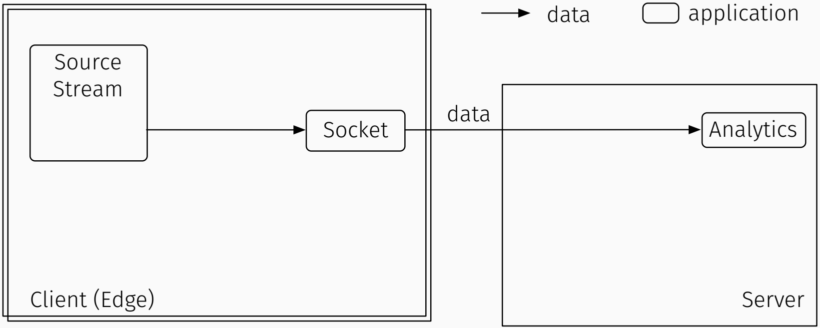


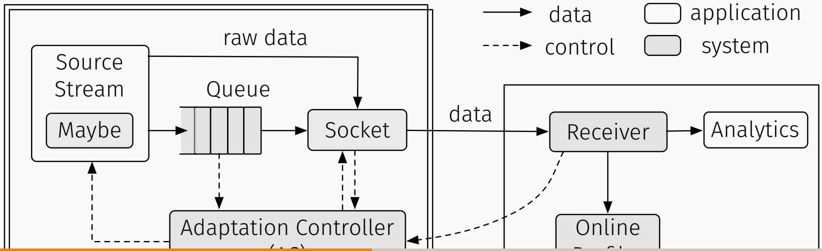
Symbol	Description
n	number of degradation operations
k_i	the i-th degradation knob
$c = [k_1, k_2, \ldots k_n]$	one specific configuration
$\mathbb{C}$	the set of all configurations
$B(c)$	bandwidth requirement for c
$A(c)$	accuracy measure for c
$\mathbb{P}$	Pareto-optimal set

$$\mathbb{P} = \{c \in \mathbb{C} : \underbrace{\{c' \in \mathbb{C} : B(c') < B(c), A(c') > A(c)\}}_{\text{set of better configurations } c'} = \varnothing\}$$

See red markers in the figure.

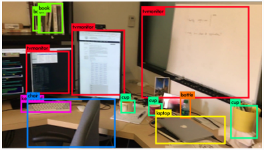
(3) Runtime Adaptation

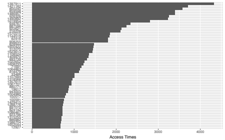




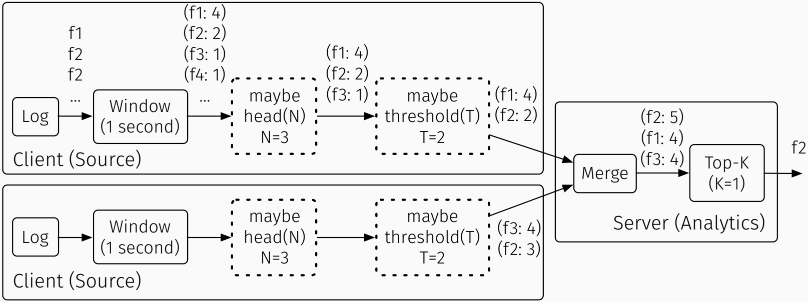
Applications

Application	Knobs	Accuracy	Dataset
Augmented Reality	resolution frame rate quantization	F1 score [Rijsbergen, 1979]	iPhone video clips training: office (24s) testing: home (246s)
Pedestrian Detection	resolution frame rate quantization	F1 score	MOT16 [Milan et al., 2016] training: MOT16-04 testing: MOT16-03
Log Analysis (Top-K, K=50)	head (N) threshold (T)	Kendall's τ [Abdi, 2007]	SEC.gov logs [DERA, 2016] training: 4 days testing: 16 days





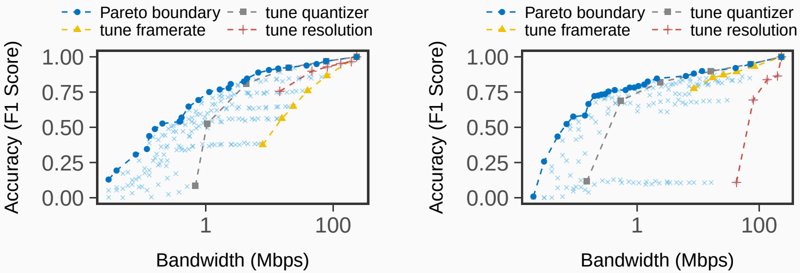
Top-K



A distributed Top-K application with two degradation operations: `head` and `threshold`. In this example, `f2`, which is not in Top-1 for either client, becomes the global Top-1 after the merge. It would have been purged if the clients use threshold T=3, demonstrating degradation that reduces data sizes affects fidelity.

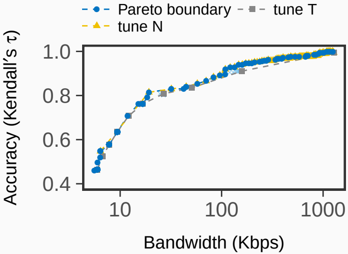
Evaluation: Generated Profiles

(a) Augmented Reality (AR) (b) Pedestrian Detection (PD)



- Optimal strategy is achieved with multiple dimensions; tuning one dimension leads to suboptimal performance.
- For the same application, different dimensions have different impact.
- For different applications, the same dimension has different impact.

Evaluation: Generated Profiles (Top-K)

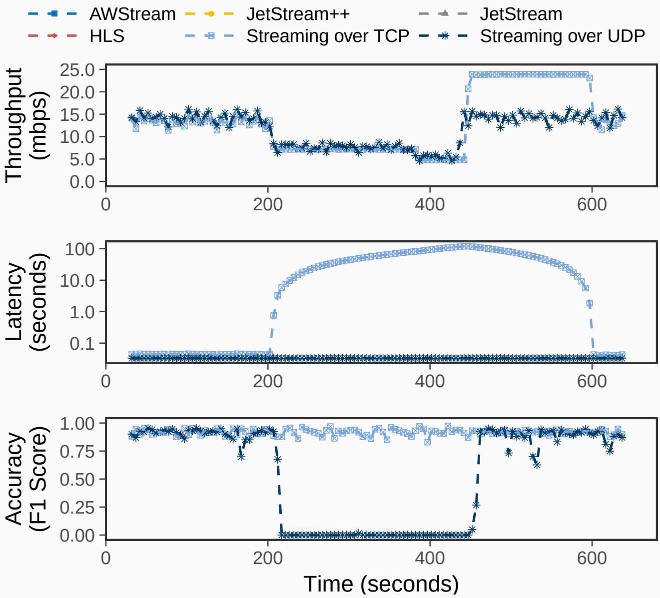


- The effect of each dimension is not significantly different.
- The profile offers quantified effects of degradation.

Evaluation: Runtime Experiment Baselines

Baseline	Description
Streaming over TCP	A non-adaptive approach
Streaming over UDP	A non-adaptive approach, represents RTP/UDP/RTSP video streaming
JetStream [Rabkin et al., 2014]	Manual Policy: *"if bandwidth is insufficient, switch to sending images at 75% fidelity, then 50% if there still isn't enough bandwidth. Beyond that point, reduce the frame rate, but keep the image fidelity."*
JetStream++	Uses adaptation policy generated by AWStream. JetStream runtime does not probe (hence may oscillate between policies).
HLS [Pantos and May, 2016]	HTTP Live Streaming represents popular adaptive video streaming techniques; used for Periscope video stream [Wang et al., 2016].

Evaluation: Runtime Performance



Compute Resource Adaptation

Edge Computing: Fog/Cloudlet/Swarmbox & New Infrastructure

Cisco Fog Computing [Bonomi et al., 2012]

Cloudlet [Satyanarayanan et al., 2009]

Philips Hue Hub

SmartThings

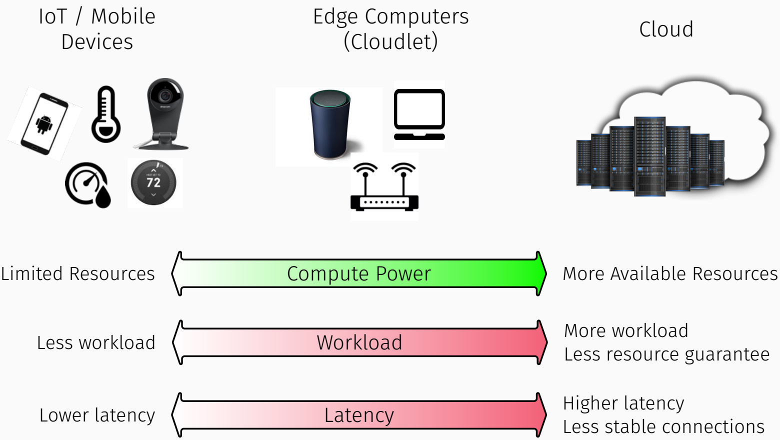
Smartphones

Google onHub

SwarmBox
Intel NUC

Many Gateways

Heterogeneous Environment



Characteristics of IoT/mobile, edge and cloud

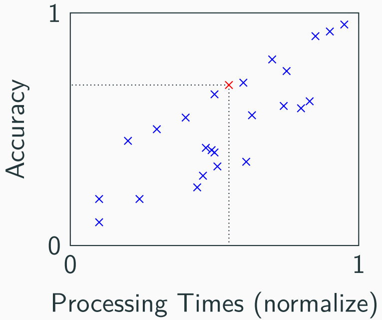
RPi Macbook Workstation

Accuracy and Processing Times Tradeoff

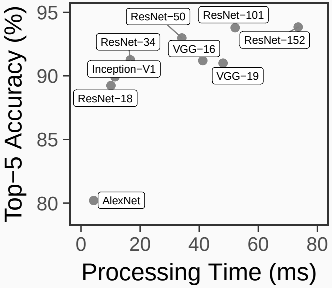
Adaptation

Different algorithm and parameters affect the accuracy and processing times.

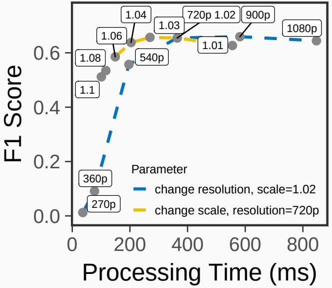
Within the tradeoff space, select appropriate algorithm and parameters to meet bounded response time goal.



Accuracy and Processing Times Tradeoff



(a) Benchmarks for popular convolutional neural network (CNN) models. Data source: https://github.com/jcjohnson/cnn-benchmarks.



(b) Benchmarks for Viola Jones face detection when changing different parameters (see explanation on the next slide).

detectMultiScale in Viola-Jones (or `CascadeClassifier`)

The OpenCV implementation of Viola-Jones [Viola and Jones, 2001] has three parameters,

- `scale`: how much the image size is reduced at each image scale.
- `min_size`: minimum detect-able object size.
- `min_neighbors`: how many neighbors each candidate rectangle should have to retain it.

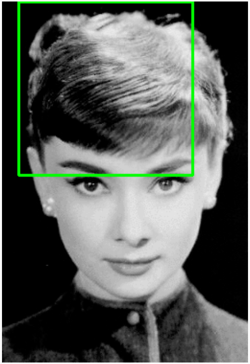
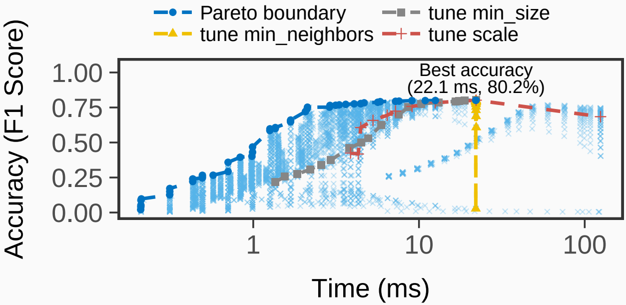


Image Source: pyimagesearch.

Exhaustive Search is Too Expensive



- `scale`: how much image size is reduced at each image scale.
- `min_size`: minimum detect-able object size.
- `min_neighbors`: how many neighbors each candidate rectangle should have to retain it.

detectMultiScale in Histogram of Oriented Gradients (HOG)

```
pub struct HogParams {
    pub win_size: Size2i,
    pub block_size: Size2i,
    pub block_stride: Size2i,
    pub cell_size: Size2i,
    pub nbins: c_int,
    pub win_sigma: f64,
    pub l2hys_threshold: f64,
    pub gamma_correction: bool,
    pub nlevels: usize,
    pub hit_threshold: f64,
    pub win_stride: Size2i,
    pub padding: Size2i,
    pub scale: f64,
    pub group_threshold: c_int,
    pub use_meanshift_grouping: bool,
    pub final_threshold: f64,
}
```

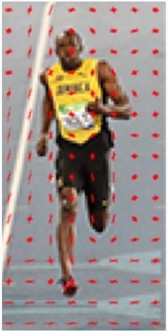


Image Source: learnopencv.com.

Challenges in Adapting Computation

Goal

Adapt computation to different platforms

Challenges:

1. Large parameter space
   - Previous approaches use random search or coordinate/greedy approach
   - We propose Bayesian Optimization (BO) for profiling
2. Heterogeneous capabilities (and not available when profiling)
   - Profile transfer: refine existing Pareto-optimal points

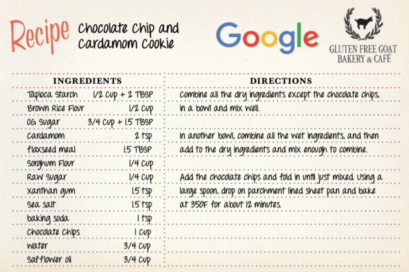
Bayesian Optimization 101

Bayesian optimization approximate black-box functions with proxy functions and iteratively proposes new sample point in the large parameter space. Effective for,

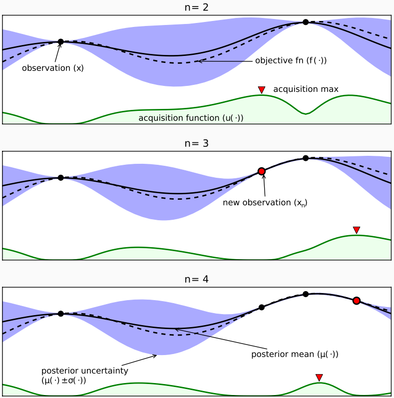
- Evaluating each sample is expensive.
- The objective is a black-box.
- The evaluation can be noisy.

Gaining attraction beyond ML scope:

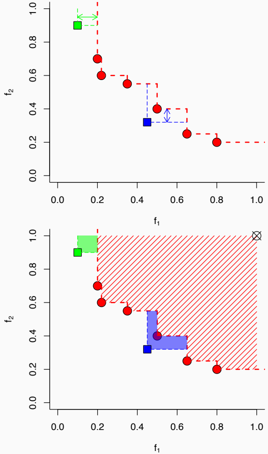
- CherryPick [Alipourfard et al., 2017] finds the best cloud configurations for big data analytics.
- Google optimize chocolate chip cookies recipes [Solnik et al., 2017].



Bayesian Optimization (Illustrated)



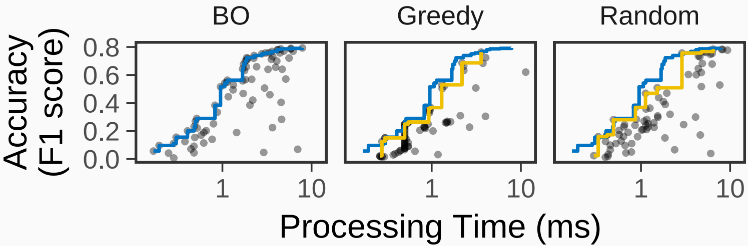
Acquisition function evaluates the utility of candidate points for the next evaluation of *f*, balancing a high objective (exploitation) and high uncertainty (exploration) [Shahriari et al., 2016]



For two-objective optimization, utility gain is based on additive-epsilon (top) or hypervolume (bottom) [Binois and Picheny, 2018]

Bayesian Optimization For Performance Modeling

We use PESMO[2] [Hernández-Lobato et al., 2016] and compare it with two baselines: (1) greedy/coordinate search; (2) random search.



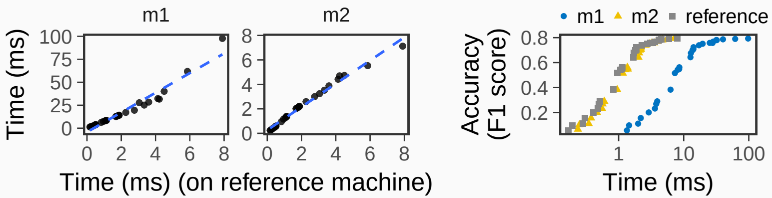
BO evaluates 50 configurations and recommends 29 configurations as the Pareto-optimal boundary (the blue line). Greedy and Random find sub-optimal Pareto configurations with a budget of 80 evaluations (the yellow line in each figure).

[2] A Python package based on Spearmint. It chooses evaluation points to maximally reduce the entropy of the posterior distribution over the Pareto set.

Profile Transfer (without re-running the entire BO)

We make the following observations:

- Accuracy remains for a given algorithm/parameter.
- Processing time order is preserved
  - More expensive algorithms/parameters remain the same across platforms.
- The "Pareto-optimal" is horizontally stretched/compressed.



(Left) Empirically, processing times follows a linear approximation. (Right) Stretched/compressed profile. See paper for details.

Conclusion and Acknowledgement

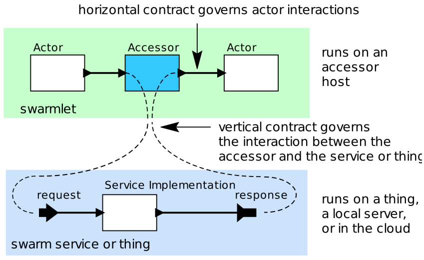
Summary and Contributions

- Swarm/IoT has huge potentials but also challenges
- Network resource adaptation
  - Addresses scarce and variable WAN bandwidth
  - Tradeoff between application accuracy and data size demand
- Compute resource adaptation
  - Addresses heterogeneous platforms and large parameter space
  - Tradeoff between application accuracy and processing times
- Overall, a systematic and quantitative approach for adaptation

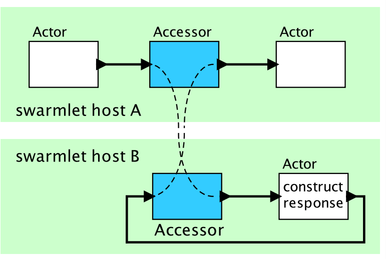
Current (other) and Future Work

TerraSwarm Vision

TerraSwarm applications are characterized by their ability to dynamically recruit resources such as sensors and data from the cloud, aggregate and use that information to make or aid decisions.



(a) Accessor in a network of actors.



(b) Instantiate accessors on another host.

Work in progress with Marten and Andrés. Maybe checkout Marten's dissertation talk in the future :)

Acknowledgment

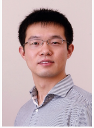

Prof. Edward Lee


Prof. John Wawrzynek


Prof. John Chuang

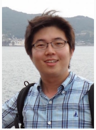

Prof. Sylvia Ratnasamy

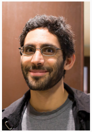

Prof. Björn Hartmann


Prof. Xin Jin


Prof. John Kubiatowicz


Dr. Ilge Akkaya


Dr. Hokeun Kim


Dr. David Mellis


Dr. Chris Shaver


Antonio Iannopollo


Kaifei Chen

The entire EAL research group (with many visiting scholars)

Acknowledgement

Edward Lee, John Wawrzynek, Syliva Ratnasamy, John Chuang, Björn Hartmann, John Kubiatowicz, Xiaofan Fred Jiang, Lin Zhang, David Mellis, Xin Jin, Mary Stewart, Christopher Brooks, Eunsuk Kang, Ilge Akkaya, Hokeun Kim, Marten Lohstroh, Matt Weber, Antonio Iannopollo, Mehrdad Niknami, Chris Shaver (Yvan Vivid), Michael Zimmer, Christos Stergiou, Dai Bui, Ben Lickly, Eleftherios Matsikoudis, Joseph Ng, Chadlia Jerad Ep Ben Haj Hmida, Moez Ben Haj Hmida, Maryam Bagheri, Victor Nouvellet, Ankush Desai, Nitesh Mor, Yu-Hsiang Sean Chen, Claire Tuna, Achal Dave, Jack Kolb, Eric Allman, Roy Wang, Bill N. Schilit, Jin Liang, Chao Mei, Kaifei Chen, Qifan Pu, Xiang Gao, Peihan Miao, Zhuo Chen, Yuting Wei, Chaoran Guo, Qian Zhong, Tianshi Wang, Meng Wei, Limin Chen
♡

1N73LL1G3NC3
15 7H3
4B1L17Y
70 4D4P7 70
CH4NG3

Image Source: 0stees.com

Google Network Infrastructure

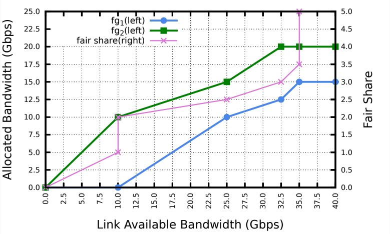


Figure 4: Bandwidth Sharing on a Bottleneck Link.

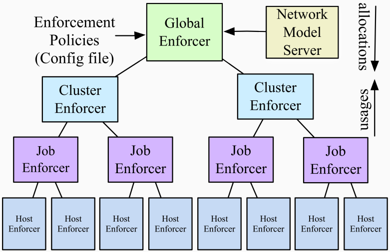


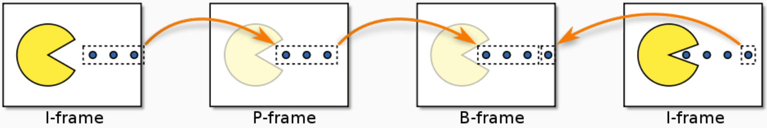
Figure 5: BwE Architecture.

BwE: Flexible, Hierarchical Bandwidth Allocation for WAN Distributed Computing [Kumar et al., 2015]

Move from Lagrangian to Eulerian (ask Edward if you don't know what these words refer to).

Backup Slides.

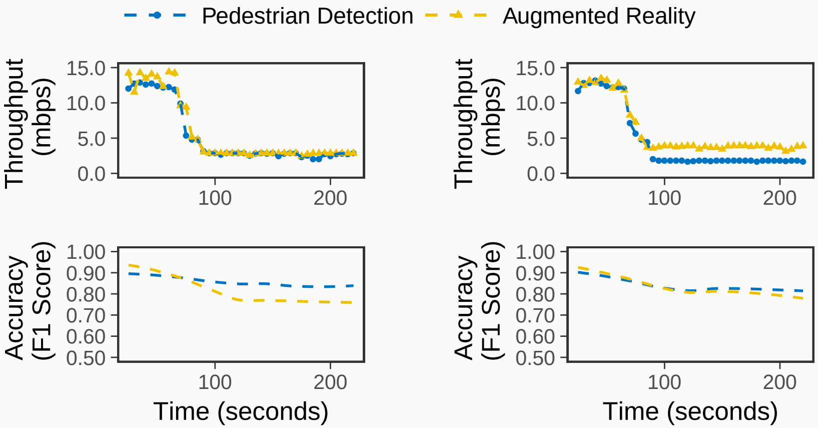
Video Encoding Frames



PC: https://en.wikipedia.org/wiki/Video_compression_picture_types

- **I-frames** are the least compressible but don't require other video frames to decode. I-frames are further compressed with quantization.
- **P-frames** can use data from previous frames to decompress and are more compressible than I-frames.
- **B-frames** can use both previous and forward frames for data reference to get the highest amount of data compression (not an option in live streaming).

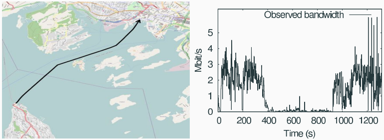
Evaluation: Resource Allocation for Multiple Applications



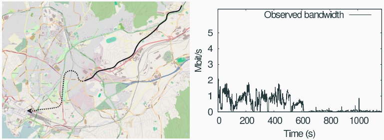
(a) Resource Fairness

(b) Utility Fairness

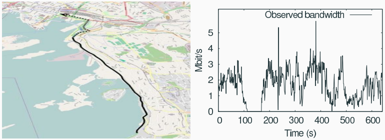
Bandwidth Fluctuations (Cellular)



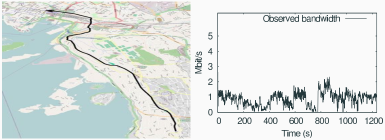
(a) Ferry



(b) Metro (dotted line in tunnel)



(c) Bus



(d) Tram

Riiser, Haakon, et al. "A comparison of quality scheduling in commercial adaptive HTTP streaming solutions on a 3G network." Proceedings of the 4th Workshop on Mobile Video. ACM, 2012.

Bandwidth Fluctuations (WiFi)

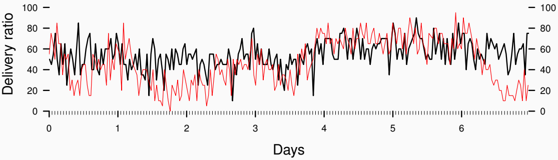


Figure 4: Delivery ratio variation over a week for two randomly chosen 2.4 GHz links.

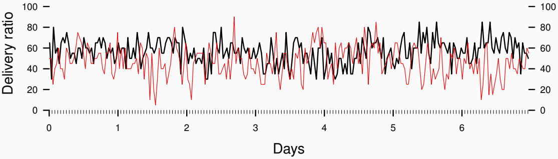


Figure 5: Delivery ratio variation over a week for two randomly chosen 5 GHz links.

Biswas et al, Cisco Meraki, Large-scale Measurements of Wireless Network Behavior, SIGCOMM'15. Two randomly chosen links.

Continue with the main slides.

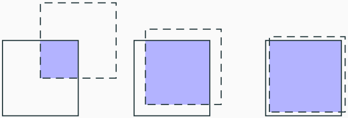
Augmented Reality

- Training and testing data characteristics
  - 1920x1080 resolution with 30 FPS
  - training: 707 frames (23.5 seconds), testing: 1384 frames (46 seconds)
- Object Recognition
  - Darknet: Open Source Neural Networks in C
  - Developed by Joseph Redmon, "Do whatever you want with it" license
  - It supports CPU/GPU
  - In this work, I am using a pre-trained model with Coco dataset
- Other systems such as TensorFlow, Caffe would also work

IOU and F1

Positive if intersection over union (IOU) larger than 0.5.

$$\text{IOU} = \frac{\text{Area of Intersection}}{\text{Area of Union}}$$



(a) IOU=0.14 (b) IOU=0.57 (c) IOU=0.82

F1 is the harmonic mean of precision and recall:

	P	N
Y	True Positive	False Positive
N	True Positive	False Positive

$$\text{Precision} = \frac{\text{true positive}}{\text{all positive}}$$

$$\text{Recall} = \frac{\text{true positive}}{\text{all detection}}$$

$$\text{F1} = \frac{2}{\frac{1}{\text{Recall}} + \frac{1}{\text{Precision}}}$$

references i

Abdi, H. (2007).
The Kendall Rank Correlation Coefficient.
Encyclopedia of Measurement and Statistics. Sage, Thousand Oaks, CA, pages 508–510.

Alipourfard, O., Liu, H. H., Chen, J., Venkataraman, S., Yu, M., and Zhang, M. (2017).
CherryPick: Adaptively Unearthing the Best Cloud Configurations for Big Data Analytics.
In *14th USENIX Symposium on Networked Systems Design and Implementation (NSDI 17)*, pages 469–482, Boston, MA. USENIX Association.

Amerasinghe, K. (2009).
H. 264 for the rest of us.

Andersen, M. P. and Culler, D. E. (2016).
Btrdb: Optimizing storage system design for timeseries processing.
In *FAST*, pages 39–52.

Binois, M. and Picheny, V. (2018).
GPareto: An R Package for Gaussian-Process Based Multi-Objective Optimization and Analysis.

Biswas, S., Bicket, J., Wong, E., Musaloiu-e, R., Bhartia, A., and Aguayo, D. (2015).
Large-scale measurements of wireless network behavior.
In *ACM SIGCOMM Computer Communication Review*, volume 45, pages 153–165. ACM.

references ii

Bonomi, F., Milito, R., Zhu, J., and Addepalli, S. (2012).
Fog Computing and Its Role in the Internet of Things.
In *Proceedings of the first edition of the MCC workshop on Mobile cloud computing*, pages 13–16. ACM.

DERA (2003–2016).
EDGAR Log File Data Set.
https://www.sec.gov/data/edgar-log-file-data-set.
Accessed: 2017-01-25.

Hernández-Lobato, D., Hernandez-Lobato, J., Shah, A., and Adams, R. (2016).
Predictive Entropy Search for Multi-Objective Bayesian Optimization.
In Balcan, M. F. and Weinberger, K. Q., editors, *Proceedings of The 33rd International Conference on Machine Learning*, volume 48 of *Proceedings of Machine Learning Research*, pages 1492–1501, New York, New York, USA. PMLR.

Kumar, A., Jain, S., Naik, U., Raghuraman, A., Kasinadhuni, N., Zermeno, E. C., Gunn, C. S., Ai, J., Carlin, B., Amarandei-Stavila, M., et al. (2015).
Bwe: Flexible, hierarchical bandwidth allocation for wan distributed computing.
In *ACM SIGCOMM Computer Communication Review*, volume 45, pages 1–14. ACM.

Milan, A., Leal-Taixé, L., Reid, I., Roth, S., and Schindler, K. (2016).
MOT16: A Benchmark for Multi-Object Tracking.
arXiv preprint arXiv:1603.00831.

Pantos, R. and May, W. (2016).
HTTP Live Streaming.

Rabkin, A., Arye, M., Sen, S., Pai, V. S., and Freedman, M. J. (2014).
Aggregation and Degradation in JetStream: Streaming Analytics in the Wide Area.
In *Proceedings of the 11th USENIX Conference on Networked Systems Design and Implementation*, NSDI'14, pages 275–288, Berkeley, CA, USA. USENIX Association.

Rijsbergen, C. J. V. (1979).
Information Retrieval.
Butterworth-Heinemann, Newton, MA, USA, 2nd edition.

Satyanarayanan, M., Bahl, P., Caceres, R., and Davies, N. (2009).
The Case for VM-based Cloudlets in Mobile Computing.
IEEE Pervasive Computing, 8(4):14–23.

Shahriari, B., Swersky, K., Wang, Z., Adams, R. P., and De Freitas, N. (2016).
Taking the human out of the loop: A review of bayesian optimization.
Proceedings of the IEEE, 104(1):148–175.

Solnik, B., Golovin, D., Kochanski, G., Karro, J. E., Moitra, S., and Sculley, D. (2017).
Bayesian Optimization for a Better Dessert.

Viola, P. and Jones, M. (2001).
Rapid Object Detection Using a Boosted Cascade of Simple Features.
In *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001*, volume 1, pages I–511–I–518 vol.1.

Wang, B., Zhang, X., Wang, G., Zheng, H., and Zhao, B. Y. (2016).
Anatomy of a Personalized Livestreaming System.
In *Proceedings of the 2016 ACM on Internet Measurement Conference*, pages 485–498. ACM.

Zhang, B., Mor, N., Kolb, J., Chan, D. S., Lutz, K., Allman, E., Wawrzynek, J., Lee, E. A., and Kubiatowicz, J. (2015).
The Cloud is Not Enough: Saving IoT from the Cloud.
In *7th USENIX Workshop on Hot Topics in Cloud Computing (HotCloud 15)*. USENIX Association.