ECOS: Practical Mobile Application Offloading for Enterprises

Aaron Gember, Charlotte Dragga, Aditya Akella

University of Wisconsin-Madison

Mobile Device Trends

- More mobile device usage in enterprises
  - Need to run complex applications





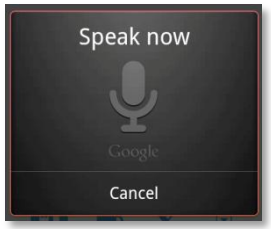
- Complex mobile applications have significant CPU, memory, and energy demands

Need to reconcile application demands and device capabilities

Designing for Remote Computation

- Mobile apps designed to use remote services
  - e.g., Google Voice Search, Apple's Siri, Amazon Silk

❌ Requires developers to use this paradigm



- Remote desktop / VNC

❌ User interface not designed for mobile devices

Application-Independent Offloading

- Dynamically divide execution between mobile device and compute resource
  - Application code unmodified
  - Informed by model and/or runtime monitoring

- Several proposed systems – Chroma *[Balan et al. 2003]*, MAUI *[Cuervo et al. 2011]*, CloneCloud *[Chun et al. 2011]*

Roadblocks to Offloading Adoption

- Privacy and trust 
  - Proposed systems largely ignore privacy
  - Privacy is paramount in enterprises

- Resource sharing and churn 
  - Proposed systems consider one device, plus a dedicated resource
  - Enterprises have *many devices* and *many resources*

Enterprise-Centric Offload System

allows **many devices** to opportunistically leverage **diverse compute resources**, while **controlling where** applications offload depending on **privacy, performance, and energy constraints** of users and apps.

Augment offloading decisions with privacy and resource considerations

Outline

- ✓ Offloading benefits and roadblocks
- Privacy and trust
- Resource sharing and churn
- ECOS Prototype
- Evaluation for small enterprise

Privacy and Trust

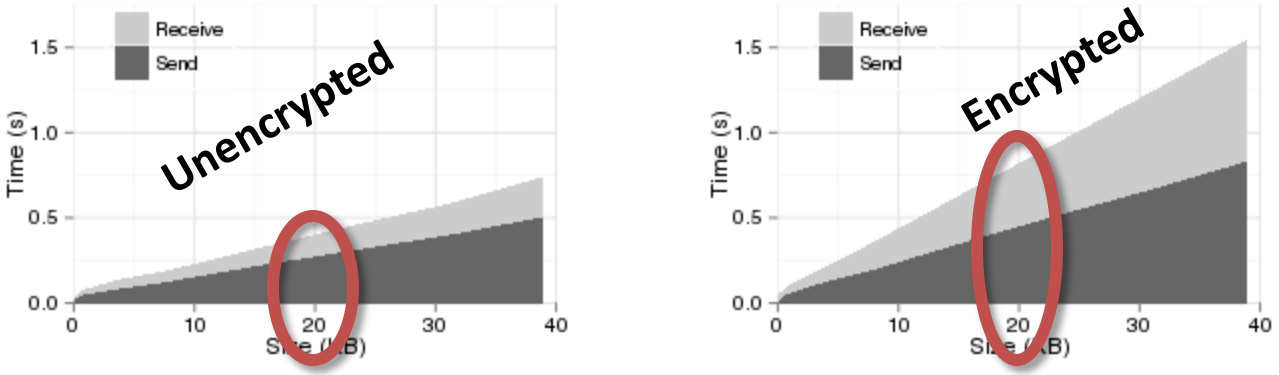
- Data may leave device without explicit actions

- Sufficient privacy requires
  - Protecting execution state in transit
  - Executing on trusted compute resources

Offloading benefits and opportunities should not be significantly diminished

Overhead of Privacy Protection

- Encrypting state in transit with TLS
  - ✗ High latency and energy overhead



- Limited number of trusted compute resources
  - ✗ Reduced offloading opportunities

Data-Driven Privacy Protection

Data Privacy	Resources Trusted	Communication Protection
None	Any	None
Enterprise	All internal resources	None
User	Select servers and desktops	Encrypt always

Policy specifies labels for devices & applications

Resource Sharing and Churn

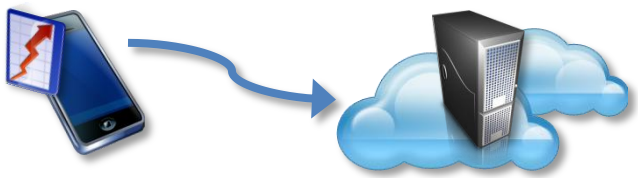
✖ Systems consider one device and static resource

Enable many mobile devices to opportunistically leverage diverse compute resources

- Challenges
  - Devices with varying applications and objectives
  - Limited resources and diverse capabilities
  - Offload requests not know a priori

Multiplexing Based on Objective

- Offloads with same objective use same resource



- Performance improvement
  - Resources with unused CPU > mobile CPU speed
- Energy savings
  - Separate from performance seeking offloads

Resource Affinity

- Use same resource for subsequent offloads
  - ✔ Cache state → less latency and energy overhead
  - ✘ Assumes constant resource availability/capacity
- Resource not capable/available
  - Deny offloads until capacity increases
  - Assign a new resource, re-transfer or migrate state

Prototype

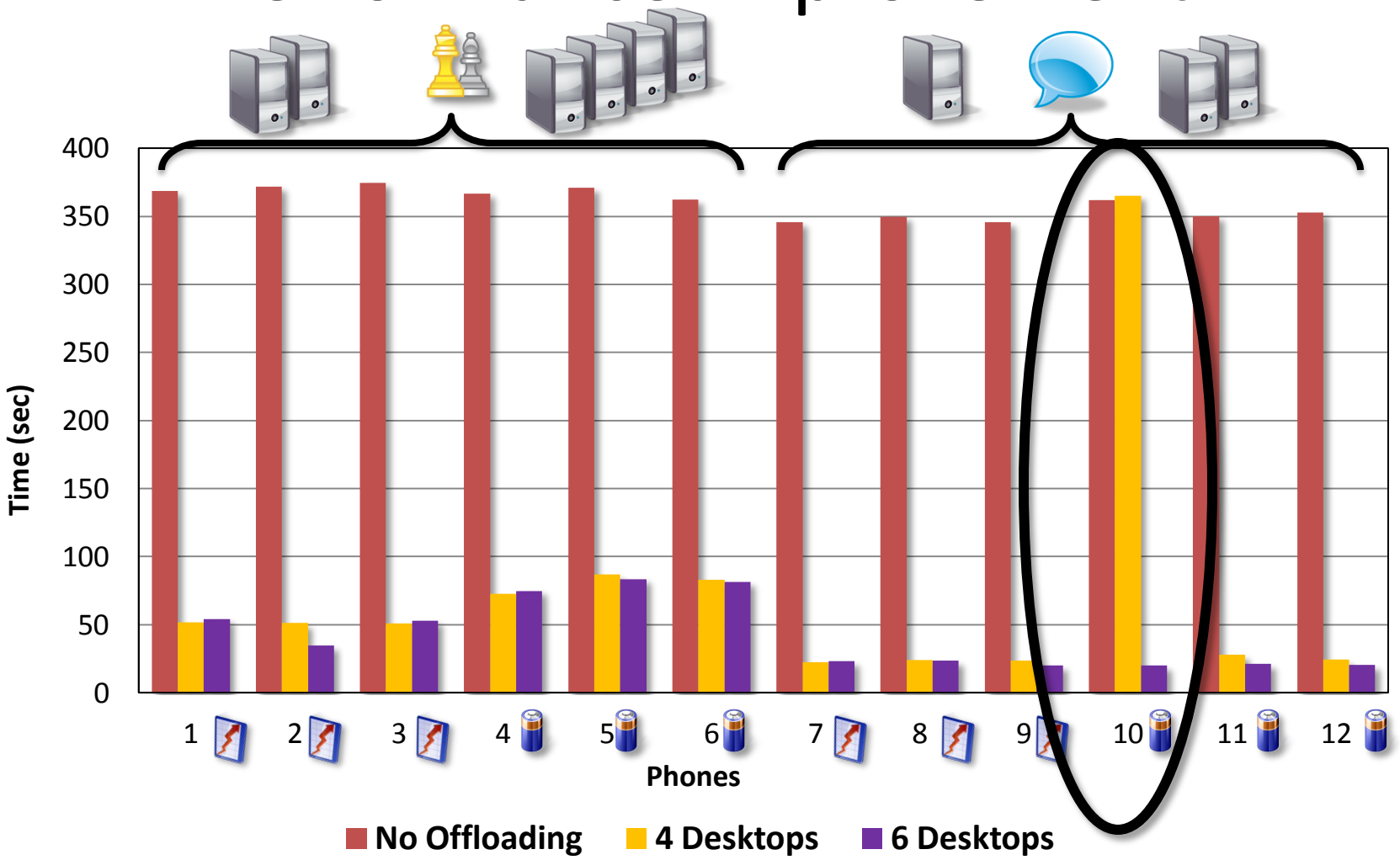
Evaluation

- Small enterprise setting
  - 12 phones (Android emulator)
  - 4 to 6 desktops (2.4Ghz quad-core, 4GB RAM)

- Two applications
  - AI-decision making (Chess)
  - Speech-to-text (emulated)

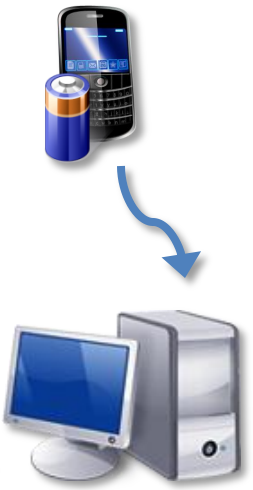
- Varying objectives and privacy levels

Performance Improvement



Summary & Future Work

- Enterprise-Centric Offloading System
  - Considers privacy requirements/costs
  - Resources assigned based on user goals
  - Maximize latency and energy benefits



- Future Work
  - Dynamic privacy label assignment
  - Network path provisioning – mobility, bandwidth
  - Larger evaluation