From Feast to Famine
Managing mobile network resources across environments and preferences

Rob Kiefer
Erik Nordström
Michael Freedman
Princeton University
Network Usage Makes Demands on Limited Resources

• Data

• Battery

• Performance
Limitations and Operating Conditions Are Not Static

• User mobility
  – indoor vs outdoor
  – WiFi vs LTE
  – multipath & migration

• User interest
  – foreground vs background apps
  – Interactive vs streaming
Divergent Goals = Resource Conflicts

• User and apps may differ
  – e.g., apps may prioritize perf, user is cost sensitive

• User has to moderate resource usage between apps
User Goals Don’t Match App Goals

Streaming Music App
User Goals Don’t Match App Goals

User wants to sacrifice quality; app overuses cell network
Flow-level Fairness != App-level Fairness
Flow-level Fairness ≠ App-level Fairness

Diagram:
- Video App
- Streaming Music App
- Network Usage
- Flows
Flow-level Fairness ≠ App-level Fairness
Interactive Apps Lack Prioritization

Browser

Streaming Music App

Network Usage
Interactive Apps Lack Prioritization

Browser

Streaming Music App

Network Usage
Interactive Apps Lack Prioritization

Interactive apps stuck behind long-running flows, hurting UX.
Strawman: Users micromanage across settings
Strawman: Users micromanage across settings

- Requires user to monitor and estimate usage over potentially long epochs (hours, days)
- Restricts typically all-or-nothing, e.g., allow on mobile or not
Strawman: Users micromanage across settings

Poor reuse in similar apps
Strawman: Users micromanage across settings

Poor reuse in similar apps

Varying levels of control

Many

Few
Strawman: Users micromanage across settings

Varying levels of control
Updates can add, remove, or change settings without user knowing
Strawman: OS-level limitations and restrictions

• OS limits background activity

• Only certain classes of apps

• Used by iOS and WinPhone

• User has no choice
Our Solution: Tango

• Usage preferences through programmatic policy; user is priority, but app flexibility

• Address conflicts *proactively*

• Migration / multipath in mind

• Handle dynamic operating conditions
Tango Overview
Tango Overview

State – metrics from kernel sources

Plans – actions to be taken

Policy – generates plans from given state
Tango Overview

1. Controller

2. User Policy

3. App Policies
Tango Overview

1. Controller

2. User Policy

3. App Policies
Tango: Controller

- Measures device **state**
- Generates policy constraints
- Carries out policy **plans**
  - Verify against constraints
  - Perform **actions** in plan
Tango: Controller

- Measures device **state**
- Executes the user policy
- Carries out policy **plans**
  - Verify against constraints
  - Perform **actions** in plan
Tango: Policies

• Programs

• Turn state into plans

• Two levels: user & app
Tango: Policies

- Programs

- Turn state into plans

- Two levels: user & app

- User level
  - Global management of network resources
  - Sets resource constraints for app policies
  - Default plan for (classes of) apps
Policies: Turning State into Actions
Policies: Turning State into Actions

net_name: eth0
net_type: CELL,
batt_charge: false
batt_perc: 50
flows: 10, 11, 12...
...
## Policies: Turning State into Actions

### State

<table>
<thead>
<tr>
<th>Source</th>
<th>Info</th>
</tr>
</thead>
<tbody>
<tr>
<td>Transport Layer</td>
<td># retrans., RTTs, cong. window...</td>
</tr>
<tr>
<td>Network Layer (IP)</td>
<td>addresses, routing rules...</td>
</tr>
<tr>
<td>Link Layer</td>
<td>type, signal quality, bit errors...</td>
</tr>
<tr>
<td>Battery</td>
<td>charging status, percent...</td>
</tr>
</tbody>
</table>

net_name: eth0  
net_type: CELL,  
batt_charge: false  
batt_perc: 50  
flows: 10, 11, 12...  
...
Policies: Turning State into Actions

State

<table>
<thead>
<tr>
<th>Source</th>
<th>Info</th>
</tr>
</thead>
<tbody>
<tr>
<td>Transport Layer</td>
<td># retrans., RTTs, cong. window...</td>
</tr>
<tr>
<td>Network Layer (IP)</td>
<td>addresses, routing rules...</td>
</tr>
<tr>
<td>Link Layer</td>
<td>type, signal quality, bit errors...</td>
</tr>
<tr>
<td>Battery</td>
<td>charging status, percent...</td>
</tr>
</tbody>
</table>

net_name: eth0
net_type: CELL,
batt_charge: false
batt_perc: 50
flows: 10, 11, 12...

Policy
Policies: Turning State into Actions

**State**

<table>
<thead>
<tr>
<th>Source</th>
<th>Info</th>
</tr>
</thead>
<tbody>
<tr>
<td>Transport Layer</td>
<td># retrans., RTTs, cong. window...</td>
</tr>
<tr>
<td>Network Layer (IP)</td>
<td>addresses, routing rules...</td>
</tr>
<tr>
<td>Link Layer</td>
<td>type, signal quality, bit errors...</td>
</tr>
<tr>
<td>Battery</td>
<td>charging status, percent...</td>
</tr>
</tbody>
</table>

**Policy**

- net_name: eth0
- net_type: CELL,
- batt_charge: false
- batt_perc: 50
- flows: 10, 11, 12...
  ...

- MIGRATE(10, eth0),
- MIGRATE(11, eth0),
- MIGRATE(12, eth0),
- ...
- RATELIMIT(eth0, 500kbps)
### Policies: Turning State into Actions

#### State

<table>
<thead>
<tr>
<th>Source</th>
<th>Info</th>
</tr>
</thead>
<tbody>
<tr>
<td>Transport Layer</td>
<td># retrans., RTTs, cong. window...</td>
</tr>
<tr>
<td>Network Layer (IP)</td>
<td>addresses, routing rules...</td>
</tr>
<tr>
<td>Link Layer</td>
<td>type, signal quality, bit errors...</td>
</tr>
<tr>
<td>Battery</td>
<td>charging status, percent...</td>
</tr>
</tbody>
</table>

#### Actions

<table>
<thead>
<tr>
<th>Action</th>
<th>Iface</th>
<th>Flow</th>
</tr>
</thead>
<tbody>
<tr>
<td>ENABLE</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td>RATELIMIT</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td>LOG</td>
<td>✔</td>
<td>✔</td>
</tr>
<tr>
<td>MANAGE</td>
<td>✔</td>
<td></td>
</tr>
<tr>
<td>MIGRATE</td>
<td>✔</td>
<td></td>
</tr>
</tbody>
</table>

net_name: eth0  
net_type: CELL  
batt_charge: false  
batt_perc: 50  
flows: 10, 11, 12...  
...

Policy

MIGRATE(10, eth0),  
MIGRATE(11, eth0),  
MIGRATE(12, eth0),  
...,  
RATELIMIT(eth0, 500kbps)
Tango: App Policies

- Leverage *local* state
- Only act on their flows
- Subject to user policy constraints
- App can provide **hints**
Tango: Constraints & Hints

• Proactively address resource conflicts
  – Constraints known \textit{a priori}
  – Simplifies controller & app policy interaction cycle

• Apps can hint at desired resource needs for future rounds
USER POLICY

evaluate(s, c):
    // Plan returned.

genConstraints(s):
    c = new Constraints()
    for a in s.apps():
        if a.isForeground():
            c.put(a, HIGH)

    return c
Tango: User & App Policy

USER POLICY

evaluate(s, c):
    // Plan returned.

gencConstraints(s):
    c = new Constraints()
    for a in s.apps():
        if a.isForeground():
            c.put(a, HIGH)

    return c

APP POLICY

evaluate(s, c_a):
    plan = new Plan()
    pstate = GetPlayerState()
    // Rest of policy omitted.

    plan.hintPriority = NORM
    urgent = false
    buffer = pstate.bufferTime
    if buffer > 30:
        urgent = false
    elif buffer < 20 || urgent:
        urgent = true
        plan.hintPriority = HIGH

    return plan
Tango: User & App Policy

**USER POLICY**

```python
evaluate(s, c):
    // Plan returned.

genConstraints(s):
    c = new Constraints()
    for a in s.apps():
        hint = a.hintPriority
        if a.isForeground():
            c.put(a, HIGH)
        elif Allowed(a, hint):
            c.put(a, hint)

    return c
```

**APP POLICY**

```python
evaluate(s, c_a):
    plan = new Plan()
    pstate = GetPlayerState()
    // Rest of policy omitted.

    plan.hintPriority = NORM
    urgent = false
    buffer = pstate.bufferTime
    if buffer > 30:
        urgent = false
    elif buffer < 20 || urgent:
        urgent = true
        plan.hintPriority = HIGH

    return plan
```
Prototype

• Controller, sample policies, sample apps in Java for Android phones

• Flow migration provided by Serval (NSDI ’12)

• Explored the policy space for single and multiple apps
Evaluation: Streaming Music Across Spotty Campus WiFi

• Campus WiFi offers chance to offload

• But, even seamless switching w/ migration has problems
Evaluation: Streaming Music Across Spotty Campus WiFi

• Campus WiFi offers chance to offload

• But, even seamless switching w/ migration has problems

• How to use policy to improve when to switch?

• How to minimize cell usage?

• How can app policies help?
Clinging to WiFi Leads to Poor Performance
Clinging to WiFi Leads to Poor Performance

Good WiFi

Bad WiFi
Clinging to WiFi Leads to Poor Performance

Unneeded cell usage
Clinging to WiFi Leads to Poor Performance

Good WiFi

Bad WiFi
Improving Network Switching

• More aggressive in leaving WiFi; more conservative in joining

• Use heuristics on signal strength trends

• Based on measurements across campus of signal strength vs achieved throughput
Improving Streaming Music Through Policy

No rate limit

Buffer (secs)

Time (secs)

WiFi
Cell
Improving Streaming Music Through Policy

No rate limit

Rate limit
Improving Streaming Music Through Policy

No rate limit

Rate limit

App Policy
User and App Policy Cooperation Yields Best Usage

Rate uses **5-6x** less cell data than Unl.

App uses about **30x** less cell data than Unl.
Interactive App Competing w/ Streaming App

**Problem**
Poor resource isolation leads to erratic load times
Interactive App Competing w/ Streaming App

Problem
Poor resource isolation leads to erratic load times

Solution
App fair share, prioritize music only when buffer low
Interactive App Competing w/ Streaming App

![Graph showing page load time and playback buffer over time for Music app and Web app.](image)
Related Work

• Context awareness: CASS, CARISMA, JCAF
  – Using context to improve apps
  – Not looking at resource usage; not single device
Related Work

• Context awareness: CASS, CARISMA, JCAF
  – Using context to improve apps
  – Not looking at resource usage; not single device

• Network Choice/WiFi Offloading
  – Many: Whiffler, IMP, BreadCrumbs, SALSA...
  – Complementary to Tango
Related Work

• Context awareness: CASS, CARISMA, JCAF
  – Using context to improve apps
  – Not looking at resource usage; not single device

• Network Choice/WiFi Offloading
  – Many: Whiffler, IMP, BreadCrumbs, SALSA...
  – Complementary to Tango

• Serval, MPTCP, TCP-Migrate
  – Need policy to guide decisions on nets to use
  – Complementary to Tango
Conclusions

• Tango is a network resource management framework based on programmatic policy

• Proactively handle resource conflicts while including user and app prefs into decisions

• Demonstrated the value it adds for single apps as well as across apps
Thanks! Questions?