**Application Assisted Power Management in Multiplayer Mobile Games**

Bhojan Anand‡, A. L. Ananda‡, Mun Choon Chan‡, Rajesh Krishna Balan†

‡National University of Singapore and †Singapore Management University

---

**Introduction**

Parallel growth Converge to next “Killer” application
- Online games – phenomenal increase in market share
- The number of cell phone users continues to escalate
- Multiplayer Mobile Games – Next “Killer” application?!!

**Usability Constraint:**
- High Power Consumption
  - Processor load, Continuous NW traffic, Display intensity

**Objective:**
- Develop an energy saving scheme for multiplayer mobile gaming

---

**Characteristics of Networked Games**

**Game**
- Average bandwidth per client 7Kbps
- Average packet rate (incoming and outgoing): 20 pps
- Very small packet size, Average: 24 DATA +8 UDP + 20 IP Bytes

---

**Saving WNIC Power consumption**

**Safe sleep mode**
- Acceptable latency - 200 ms for FPS and 1500 ms for RPG (slow speed) with Dead Reckoning.
- Challenge 1: When to sleep without affecting the quality of game play?
  - When the actions are not significant for the quality of the game play.
- Challenge 2: How long is the sleep duration?
  - Long duration affects the quality of game
  - Short duration will result in more overhead power consumption
- Complex algorithm results in adverse results!

---

**Results & Conclusion**

**Average Amount Energy Saved in Various Modes:**
- With current implementation in Android platform (Mobile Armageddon - RPG game)
  - Full Mode (35%), Secured Mode (No Server Hints -> 20%), Black-box Mode (No Application Hints -> 13%)

We have:
- Learnt about Characteristics of Network Gaming
- Developed Linear Prediction Algorithm to predict “Importance of Game State”
- Developed Algorithm for Power Management using Application & Environment hints
- We have developed an API extension for existing game engine to learn about the entire game state for RPG games

---

**Architecture & Information Flow**

**Game State**
\[
\text{GameStateIndex}_i = f(\text{gameActionIndex}_i, \text{interestIndex}_i)
\]

**Network State**
\[
\text{NetworkStateIndex}_i = f(\text{latency}_i, \text{bandwidth}_i)
\]

**Interest State**
\[
\text{InterestStateIndex}_i = f(\text{interestData}_i)
\]

**Battery State**
\[
\text{BatteryStateIndex}_i = f(\text{remainingBatteryLife}_i)
\]

**Game State Prediction**
\[
\text{GameStateIndex}_{i+1} = f(\text{gameActionIndex}_{i+1}, \text{interestData}_{i+1})
\]

---

**Algorithms**

**GAME CLIENT APPLICATION**

**RESOURCE MANAGER**
\[
\text{PowerSavingDecision} = f(\text{Game State}, \text{Network State}, \text{Interest State}, \text{Battery State})
\]

**NETWORK MANAGER**
\[
\text{Game State Prediction} = f(\text{Previous 'n' Game States}, \text{Interest State})
\]

**PROXY / ACCESS POINT**

---

**Game State Prediction**
- High probability for important actions,
  - When the Avatar is near another Avatar or NPC
  - When Avatars and NPCs are approaching the player
- Avatar will have different actions in different zones