Eat all you can in an all-you-can-eat buffet: A case for aggressive resource usage

Ratul Mahajan
Jitu Padhye, Ramya Raghavendra, Brian Zill

Microsoft Research
Avoiding hunger

Alice walks into a restaurant

How much to eat to minimize the chance of getting hungry before the next meal?

1. As much as stomach space allows; or

2. Based on expected time until next meal
Avoiding packet loss

Bob needs to transmit data over a noisy wireless channel

How much FEC to add to minimize the chance of losing packets?

1. As much as the available spectrum; or
2. Based on expected bit error rate
The focus on efficiency in current designs

Operating at the sweet spot tends to be the goal
• Low efficiency beyond it

Inappropriate in many scenarios
• If the resource is of “use it or lose it” kind
• If the sweet spot is hard to determine
The Buffet principle

Continue using more resources as long as the marginal cost can be driven lower than the marginal benefit

Key strength: performance limited by available resources, not artificial design choices

Key challenge: the default way to aggressively use resources is often problematic
Case: Adding FEC to data transmissions

Current practice: # of FEC bits is independent of load and available spectrum

Buffer: aggressively use available spectrum for FEC bits

Simulation results with 1000-byte packets, 1 Mbps channel, $10^{-6}$ BER
Making Buffet FEC practical

Greedy addition of FEC can hurt with multiple transmitters
  • Especially for CSMA systems (e.g., WiFi)

Possible solution [under development]
  a) Embed bits in separate packets
  b) Send at low priority
  c) Keep FEC packets small
Case: Erasure coding for lossy paths

Current practice: balance # of coded packets and bandwidth cost

Buffet: aggressively uses all available capacity

Managing the impact of aggressive redundancy

- Use priority on packets if router support is available
- Transmit only when queue is empty [under development]
More case studies

Mobility updates in ad hoc networks
• Must balance freshness of info and bw overhead
• Buffet: Aggressively send updates per spare capacity
  • Prioritization can minimize impact on data transfers

Routing in delay-tolerant networks (DTNs)
• Must balance message delivery prob. and fairness
• Recent design (RAPID) uses the Buffet principle
  • A utility-driven framework prevents disproportionate resource usage by some messages
Yet more case studies

Using replication to boost reliability

• Across disks within a computer
  o Use background tasks to manage overhead

• Across multiple computers
  o Use background transfers to manage overhead

Pre-loading binaries into memory

Pre-fetching Web pages into caches

Speculative execution of program branches
Considerations in applying the Buffet principle

What are the challenges?

What resources does it apply to (naturally)?

What are the side-effects?
Challenges in applying the principle

Aggressive resource usage should not detract from productive work

• Helpful techniques: explicit or implicit priorities; opportunistic usage; and utility-driven usage

Quantifying the marginal cost and benefit

• Must include the impact of side-effects as well
• Precise accounting not needed in many situations
Applicable resources

More natural for non-conservable resources
• E.g., storage, bandwidth, computation
• Feasible for others as well (e.g., battery)

Easier when the resource is not shared with non-Buffet users
Side-effects of aggressive resource use

New bottlenecks or concerns
  • E.g., disk I/O bw, energy consumption

Increased hardware wear-and-tear

Task completion latency may increase

Performance is now coupled with load
Conclusions

Continue using more resources as long as the marginal cost *can be driven* lower than the marginal benefit

The Buffet principle advocates aggressive resource usage instead of a singular focus on efficiency

It has the potential to provide the best performance for the level of available resources