Overflow Management with Multipart Packets

Dror Rawitz
EE @ Tel-Aviv University

Joint work with Yishay Mansour and Boaz Patt-Shamir
To appear in INFOCOM 2011
Overview

- Competitive analysis
  - Deterministic online algorithms
  - Randomized online algorithms (oblivious adversary)

- Online scheduling of multipart packets
  - Motivation
  - Problem definition and special cases
  - Related Work

- Our results
  - Theoretical results
  - Experimental results

- Conclusion
Online Algorithm:

- Input is revealed piece by piece over time
- Algorithm must make irrevocable decisions without access to whole input
Online Algorithm:

- Input is revealed piece by piece over time
- Algorithm must make irrevocable decisions without access to whole input

Competitive Analysis:

- Solution is compared to offline OPT
- ALG has competitive ratio \( c \) if \( \exists \alpha \) s.t.

\[
\forall I, \quad \text{ALG}(I) \geq \frac{\text{OPT}(I)}{c} - \alpha
\]
Online Algorithm:

- Input is revealed piece by piece over time
- Algorithm must make irrevocable decisions without access to whole input

Competitive Analysis:

- Solution is compared to offline \( \text{OPT} \)
- \( \text{ALG} \) has competitive ratio \( c \) if \( \exists \alpha \) s.t.

\[
\forall I, \quad \text{ALG}(I) \geq \frac{\text{OPT}(I)}{c} - \alpha
\]

- Game against an adversary
  - Knows algorithm
  - Determines the input sequence
  - Can obtain offline optimum

Dror Rawitz, EE@TAU

Overflow Management with Multipart Packets – 3 / 24
Randomized Online Algorithm:

- Algorithm uses random choices
- Performance is measured by expectation
- ALG has competitive ratio $c$ if there is a constant $\alpha$ s.t.

$$\forall I, \ E(\text{ALG}(I)) \geq \frac{\text{OPT}(I)}{c} - \alpha$$
Oblivious Adversary

Randomized Online Algorithm:

- Algorithm uses random choices
- Performance is measured by expectation
- \( \text{ALG} \) has competitive ratio \( c \) if there is a constant \( \alpha \) s.t.

\[
\forall I, \quad \mathbb{E}(\text{ALG}(I)) \geq \frac{\text{OPT}(I)}{c} - \alpha
\]

Oblivious Adversary:

- Knows online algorithm but not random choices
- Determines the input sequence
- Can obtain offline optimum
Motivation

Video Transmission Over Networks:

- **Video source:**
  - Sequence consisting of large frames
    (I-frames: SD hundreds of Kb; HD several Mb)

- **Small Transfer Units:**
  - IP packet size $\leq 64$Kb
  - Practically $\leq 1.5$Kb (Ethernet)
Video Transmission Over Networks:

- **Video source:**
  - Sequence consisting of large frames
    (I-frames: SD hundreds of Kb; HD several Mb)

- **Small Transfer Units:**
  - IP packet size ≤ 64Kb
  - Practically ≤ 1.5Kb (Ethernet)

- Packets arrive in **bursts** at an outgoing link of a router

- **Problem:**
  - If enough packets are dropped, the whole frame may be lost!
  - Which packet are dropped when buffer is full?
Multihop Packet Scheduling:

- Task: delivery of a packet over multi-hop path
- Each router can serve at most one packet from each burst
- Packet is delivered only if it is not dropped by any of the routers along the path
- Each pair of (time, hop) corresponds to a conflict (burst)
Online Scheduling of Superpackets

Input:

- **Superpackets** consisting of $k$ packets
- Packets arrive in bursts of size at most $\sigma$
- Server **capacity** is $c$ packets per time unit
- Server has a **buffer** of size $b$
- Packets that are not transmitted or saved in buffer are lost
- Superpacket is **useful** only if $k^\star \triangleq (1 - \beta)k$ of its packets survive (FEC)
Online Scheduling of Superpackets

Input:
- Superpackets consisting of $k$ packets
- Packets arrive in bursts of size at most $\sigma$
- Server capacity is $c$ packets per time unit
- Server has a buffer of size $b$
- Packets that are not transmitted or saved in buffer are lost
- Superpacket is useful only if $k^* \triangleq (1 - \beta)k$ of its packets survive (FEC)

Question: which packet to serve?!
Online Scheduling of Superpackets

Input:
- Superpackets consisting of $k$ packets
- Packets arrive in bursts of size at most $\sigma$
- Server capacity is $c$ packets per time unit
- Server has a buffer of size $b$
- Packets that are not transmitted or saved in buffer are lost
- Superpacket is useful only if $k^* \triangleq (1 - \beta)k$ of its packets survive (FEC)

Question: which packet to serve?!

Special Cases:
- Capacity ($c = 1$ or $c > 1$)
- Buffer ($b = 0$ or $b > 0$)
- Redundancy ($\beta = 0$ or $\beta > 0$)
Special Case: FIFO buffer, unit capacity \((c = 1)\), no redundancy \((\beta = 0)\)

- FIFO Buffer Management [Kesselman, Patt-Shamir, Scalosub 09]

  - No competitive online algorithm, even for \(k = 2\)
  - Assumption: order respecting sequences
    - \(\Omega(k)\) lower bound
    - \(O(k^2)\)-competitive algorithm
Related Work: Deterministic Algorithms

Special Case: FIFO buffer, unit capacity \((c = 1)\), no redundancy \((\beta = 0)\)

- **FIFO Buffer Management** [Kesselman, Patt-Shamir, Scalosub 09]
  - No competitive online algorithm, even for \(k = 2\)
  - Assumption: order respecting sequences
    - \(\Omega(k)\) lower bound
    - \(O(k^2)\)-competitive algorithm

- **Aggregated Streaming Data** [Scalosub, Marbach, and Liebeherr 10]
  - Assumption: traffic consists of \(M\) aggregated streams
    - \(\Omega(kM/b)\) deterministic lower bound
    - \(O((kMb + M)^k + 1)\)-competitive algorithm
    - Simulation results: algorithm outperforms various versions of tail-drop
Special Case: no buffer ($b = 0$), no redundancy ($\beta = 0$)

- Scheduling with Interval Conflicts: [Halldórsson, Patt-Shamir, Rawitz 11]
  - Tasks are sent to different servers
  - Arrive at servers at the same order, but with different burstiness
  - Burst is an interval of tasks

- Assumptions: Order respecting sequences, Epochs
  - $\Omega(\log \sigma)$ lower bound (centralized)
  - $O(\log \sigma)$-competitive algorithms (noncontiguous, weighted)
  - $O(\log(\sigma/c))$-competitive algorithm (distributed, capacitated)
Special Case: no buffer \((b = 0)\), no redundancy \((\beta = 0)\)

- **Set Packing:** [Emek, Halldórsson, Mansour, Patt-Shamir, Radhakrishnan, Rawitz 10]
  - Model:
    - Elements (bursts) arrive with a list of sets (superpackets) that contain it
    - Upon arrival an element must be assigned to a set
Related Work: Online Set Packing

Special Case: no buffer ($b = 0$), no redundancy ($\beta = 0$)

Set Packing: [Emek, Halldórsson, Mansour, Patt-Shamir, Radhakrishnan, Rawitz 10]

- Model:
  - Elements (bursts) arrive with a list of sets (superpackets) that contain it
  - Upon arrival an element must be assigned to a set

- Problem:
  - As hard to approximate as Independent Set [Halldórsson et al. 00]
  - Not approximable within $O\left(\frac{k}{\log k}\right)$ [Hazan et al. 03]
Related Work: Online Set Packing

Special Case: no buffer \((b = 0)\), no redundancy \((\beta = 0)\)

- **Set Packing:** [Emek, Halldórsson, Mansour, Patt-Shamir, Radhakrishnan, Rawitz 10]
  - **Model:**
    - Elements (bursts) arrive with a list of sets (superpackets) that contain it
    - Upon arrival an element must be assigned to a set
  - **Problem:**
    - As hard to approximate as Independent Set [Halldórsson et al. 00]
    - Not approximable within \(O\left(\frac{k}{\log k}\right)\) [Hazan et al. 03]
  - **Results:**
    - \(\sigma^{k-1}\) deterministic lower bound
    - \(\tilde{\Omega}(k\sqrt{\sigma})\) randomized lower bound (centralized)
    - \(O(k\sqrt{\frac{\sigma}{c}})\)-competitive randomized algorithm (capacitated, distributed) (can be refined depending on uniformity of parameters)
Algorithm Priority

Idea: Consistent randomization

Algorithm:

- For each superpacket $S$ pick a random priority $r(S) \sim U[0, 1]$
- Upon arrival of a burst:
  - Service $c$ superpackets with highest priority in burst
Algorithm Priority

Idea: Consistent randomization

Algorithm:

- For each superpacket $S$ pick a random priority $r(S) \sim U[0, 1]$
- Upon arrival of a burst:
  Service $c$ superpackets with highest priority in burst

Advantages:

- Very simple
- Easily distributed
- Extends to weighted superpackets
Our Results

Overview
Competitive Analysis
Online Scheduling of Multipart Packets
Theoretical Results
- Our Results
  - Redundancy
  - Improved Analysis
  - Capacity
  - Large Capacity/Buffer
Experimental Results
Conclusion

Online set packing with redundancy

- \( O(\sqrt{kk^* \sigma / c}) \) upper bound on the competitive ratio,
  where \( k^* \triangleq (1 - \beta)k \)
  (can be refined depending on uniformity of parameters)
Our Results

- Online set packing with redundancy
  - \( O(\sqrt{kk^*\sigma/c}) \) upper bound on the competitive ratio, where \( k^* \triangleq (1 - \beta)k \)
  (can be refined depending on uniformity of parameters)

- Large capacity/buffer with uniform burst size
  - \( b = 0, c = O\left(\frac{1}{\varepsilon^2} \log \frac{k^*}{\varepsilon}\right) \Rightarrow \text{goodput} \geq (1 - \varepsilon)^2(1 - \beta) \)
  - Dual buffer, \( b = O\left(\frac{6k}{\varepsilon^2} \log \frac{k}{\varepsilon}\right) \Rightarrow \text{goodput} \geq (1 - \varepsilon)^3(1 - \beta) \)
Our Results

Online set packing with redundancy
- $O(\sqrt{kk^*\sigma/c})$ upper bound on the competitive ratio,
  where $k^* \triangleq (1 - \beta)k$
  (can be refined depending on uniformity of parameters)

Large capacity/buffer with uniform burst size
- $b = 0$, $c = O(\frac{1}{\varepsilon^2} \log \frac{k^*}{\varepsilon}) \Rightarrow \text{goodput} \geq (1 - \varepsilon)^2(1 - \beta)$
- Dual buffer, $b = O(\frac{6k}{\varepsilon^2} \log \frac{k}{\varepsilon}) \Rightarrow \text{goodput} \geq (1 - \varepsilon)^3(1 - \beta)$

Ignoring large bursts
- Bursts that satisfy $\sigma(t) > \alpha \cdot c$
- Such bursts can be ignored using redundancy
- Analysis when ignoring large bursts
Special Case: no buffer \((b = 0)\), unit capacity \((c = 1)\)

\[ \square \text{ Consider a superpacket } S \]

- Let \( S' \subseteq S \) such that \( |S'| = k^* = (1 - \beta)k \)
- \( N(S') = \{T : T \text{ is in conflict with } S'\} \)
Special Case: no buffer \((b = 0)\), unit capacity \((c = 1)\)

☐ Consider a superpacket \(S\)

- Let \(S' \subseteq S\) such that \(|S'| = k^* = (1 - \beta)k\)
- \(N(S') = \{T : T \text{ is in conflict with } S'\}\)
- Observation: \(|N(S')| \leq k^*\sigma\)
**Online Set Packing with Redundancy**

**Special Case:** no buffer \((b = 0)\), unit capacity \((c = 1)\)

□ Consider a superpacket \(S\)
  - Let \(S' \subseteq S\) such that \(|S'| = k^* = (1 - \beta)k\)
  - \(N(S') = \{T : T \text{ is in conflict with } S'\}\)
  - Observation: \(|N(S')| \leq k^*\sigma\)

\[\Rightarrow \Pr[S \in \text{ALG}] \leq \frac{1}{k^*\sigma}\]
Special Case: no buffer \((b = 0)\), unit capacity \((c = 1)\)

\(\square\) Consider a superpacket \(S\)

- Let \(S' \subseteq S\) such that \(|S'| = k^* = (1 - \beta)k\)
- \(N(S') = \{T : T \text{ is in conflict with } S'\}\)
- Observation: \(|N(S')| \leq k^*\sigma\)

\(\Rightarrow\) \(\Pr[S \in \text{ALG}] \leq \frac{1}{k^*\sigma}\)

\(\Rightarrow\) \(\mathbb{E}[|\text{ALG}|] \geq \sum_{S} \Pr[S \in \text{ALG}] \geq \frac{n}{k^*\sigma} \geq \frac{|\text{OPT}|}{k^*\sigma}\)
Special Case: no buffer \((b = 0)\), unit capacity \((c = 1)\)

- Consider a superpacket \(S\)
  - Let \(S' \subseteq S\) such that \(|S'| = k^* = (1 - \beta)k\)
  - \(N(S') = \{T : T \text{ is in conflict with } S'\}\)
  - Observation: \(|N(S')| \leq k^* \sigma\)

\[\Pr[S \in \text{ALG}] \leq \frac{1}{k^* \sigma}\]

\[\mathbb{E}[|\text{ALG}|] \geq \sum_{S} \Pr[S \in \text{ALG}] \geq \frac{n}{k^* \sigma} \geq \frac{|\text{OPT}|}{k^* \sigma}\]

- Algorithm Priority is \(k^* \sigma\)-competitive
Bounds on \textit{ALG}:

- Claim 1: $\mathbb{E}[|ALG|] > \frac{n}{k^*\sigma}$

(previous slide)
 Bounds on ALG:

- Claim 1: $\mathbb{E}[|ALG|] > \frac{n}{k^* \sigma}$ (previous slide)

- Claim 2: $\mathbb{E}[|ALG|] \geq \frac{|C|}{\sqrt{kk^* \sigma}}$, where $C = \left\{ S : \exists S' \subseteq S, N(S') < \sqrt{kk^* \sigma} \right\}$
Bounds on ALG:

- Claim 1: $\mathbb{E}[|\text{ALG}|] > \frac{n}{k^* \sigma}$ (previous slide)

- Claim 2: $\mathbb{E}[|\text{ALG}|] \geq \frac{|C|}{\sqrt{kk^* \sigma}}$, where $C = \{ S : \exists S' \subseteq S, N(S') < \sqrt{kk^* \sigma} \}$

Bound on OPT:

- Claim 3: $|\text{OPT} \cap C| \leq |C|$
OSP with Redundancy: Improved Analysis

□ Bounds on ALG:

– Claim 1: \( E[|\text{ALG}|] > \frac{n}{k^*\sigma} \) (previous slide)

– Claim 2: \( E[|\text{ALG}|] \geq \frac{|C|}{\sqrt{kk^*\sigma}} \), where \( C = \left\{ S : \exists S' \subseteq S, N(S') < \sqrt{kk^*\sigma} \right\} \)

□ Bound on OPT:

– Claim 3: \( |\text{OPT} \cap C| \leq |C| \)

– Claim 4: \( |\text{OPT} \setminus C| \leq \frac{n}{\sqrt{(1-\beta)\sigma}} \)

Proof:

\( S \notin \text{OPT} \) “sees” at most \( k \) packets that are used by OPT

\( \text{There are at most } kn \text{ sightings} \)

\( |\text{OPT} \setminus C| < \frac{kn}{\sqrt{kk^*\sigma}} \)
OSP with Redundancy: Improved Analysis

□ Bounds on ALG:

− Claim 1: $\mathbb{E}[|\text{ALG}|] > \frac{n}{k^*\sigma}$

− Claim 2: $\mathbb{E}[|\text{ALG}|] \geq \frac{|C|}{\sqrt{kk^*\sigma}}$, where $C = \left\{ S : \exists S' \subseteq S, N(S') < \sqrt{kk^*\sigma} \right\}$

□ Bound on OPT:

− Claim 3: $|\text{OPT} \cap C| \leq |C|$

− Claim 4: $|\text{OPT} \setminus C| \leq \frac{n}{\sqrt{(1-\beta)\sigma}}$

Proof:

▷ $S \notin \text{OPT} \text{ “sees” at most } k \text{ packets that are used by } \text{OPT}$

▷ There are at most $kn$ sightings

▷ $|\text{OPT} \setminus C| < \frac{kn}{\sqrt{kk^*\sigma}}$

$\Rightarrow |\text{OPT}| \leq |C| + \frac{n}{\sqrt{(1-\beta)\sigma}} \leq 2\sqrt{kk^*\sigma} \cdot \mathbb{E}[|\text{ALG}|]$
Capacitated Instances

Special Case: no buffer \((b = 0)\)

Our Result: Algorithm Priority is \(O\left(\sqrt{kk^*\sigma/c}\right)\)-competitive
Capacitated Instances

Special Case: no buffer \((b = 0)\)

Our Result: Algorithm Priority is \(O(\sqrt{kk^*\sigma/c})\)-competitive

Example:
- \(n\) superpackets, \(S_1, \ldots, S_n\)
- \(\binom{n}{\ell}\) bursts, each burst consists of a distinct subset of size \(\ell\)
- \(\sigma = \ell\)
- \(k = \binom{n-1}{\ell-1}\)
Capacitated Instances

Special Case: no buffer \((b = 0)\)

Our Result: Algorithm Priority is \(O\left(\sqrt{kk^*\sigma/c}\right)\)-competitive

Example:

- \(n\) superpackets, \(S_1, \ldots, S_n\)
- \(\binom{n}{\ell}\) bursts, each burst consists of a distinct subset of size \(\ell\)
- \(\sigma = \ell\)
- \(k = \binom{n-1}{\ell-1}\)
- \(c = \ell\) \(\Rightarrow\) \(\text{OPT} = n\)
Capacitated Instances

Special Case: no buffer ($b = 0$)

Our Result: Algorithm Priority is $O(\sqrt{kk^*\sigma/c})$-competitive

Example:

- $n$ superpackets, $S_1, \ldots, S_n$
- $\binom{n}{\ell}$ bursts, each burst consists of a distinct subset of size $\ell$
- $\sigma = \ell$
- $k = \binom{n-1}{\ell-1}$

- $c = \ell \Rightarrow \text{OPT} = n$
- $c = \ell - 1 \Rightarrow \text{OPT} = \ell - 1$
Large Capacity/Buffer

Large Capacity

- Let $r(S')$ be the priority of $S$

- If $\sigma$ is uniform and $c$ is “large enough”
  - roughly $r(S') \cdot \sigma$ superpackets from burst have smaller priorities
  - $r(S') \geq 1 - \frac{c}{\sigma} + \delta \Rightarrow S'$ survives all its bursts w.h.p.
  - $r(S') \leq 1 - \frac{c}{\sigma} - \delta \Rightarrow S'$ is dropped in all its bursts w.h.p.

- Priority’s goodput is close to $(1 - \beta)$
Large Capacity/Buffer

- **Large Capacity**
  - Let $r(S)$ be the priority of $S$
  - If $\sigma$ is uniform and $c$ is “large enough”
    - roughly $r(S) \cdot \sigma$ superpackets from burst have smaller priorities
    - $r(S) \geq 1 - \frac{c}{\sigma} + \delta \Rightarrow S$ survives all its bursts w.h.p.
    - $r(S) \leq 1 - \frac{c}{\sigma} - \delta \Rightarrow S$ is dropped in all its bursts w.h.p.
  - Priority’s goodput is close to $(1 - \beta)$

- **Large Buffer**
  - We use Priority with a Dual Buffer to “simulate” large capacity
  - If $\sigma$ is uniform and $b$ is “large enough” then goodput is close to $(1 - \beta)$
Simulations

- **Our tunable parameters:**
  - \( k \): number of packets/superpacket
  - \( c \): capacity
  - \( b \): buffer size
  - \( \beta \): redundancy
Simulations

- Our tunable parameters:
  - \( k \) = number of packets/superpacket
  - \( c \) = capacity
  - \( b \) = buffer size
  - \( \beta \) = redundancy

- Traffic:
  - Aggregate of 10 on/off processes with a tunable parameter \( \lambda = \lambda_{on}/\lambda_{off} \)
  - Packets are associated with superpackets using a random permutation
  - Each data point represents the average of 10 runs
Simulations

- Our tunable parameters:
  - \( k \) = number of packets/superpacket
  - \( c \) = capacity
  - \( b \) = buffer size
  - \( \beta \) = redundancy

- Traffic:
  - Aggregate of 10 on/off processes with a tunable parameter \( \lambda = \lambda_{on}/\lambda_{off} \)
  - Packets are associated with superpackets using a random permutation
  - Each data point represents the average of 10 runs

- Performance compared to:
  - Upper bound on maximum possible goodput
  - Tail-drop policy
Size of Superpacket \((k)\)

\[ \sigma = 4.67 \]

\[ c = 5; \ b = 10 \]
Offered Load ($\sigma$)

- $k = 4; c = 6; b = 10$
\[ k = 4; \quad b = 10; \quad \bar{\sigma} = 4.5 \]
\(k = 4; c = 5; \bar{\sigma} = 4.66\)
Redundancy Percentage ($\beta$)

\[ c = 5; \quad b = 10; \quad \sigma = 4.94 \]
\[ k^* = 10 \]
Previous papers considered special cases (e.g., order respecting sequence, no buffer)

This work:
- Consider capacities and a FIFO buffer
- Introduce superpacket redundancy
- Analyze OSP with redundancy
- Provide both theoretical and experimental evidence that Algorithm Priority works well when $\beta$ is small
Previous papers considered special cases (e.g., order respecting sequence, no buffer)

This work:
- Consider capacities and a FIFO buffer
- Introduce superpacket redundancy
- Analyze OSP with redundancy
- Provide both theoretical and experimental evidence that Algorithm Priority works well when $\beta$ is small

Open Questions:
- Analysis of Algorithm Priority with buffer
- Variable packet sizes (with or without a buffer)
- Another algorithm for large redundancy?