Congestion Metacontrol to achieve a Deadline Aware Less than Best Effort service

David Hayes
together with David Ros, Andreas Petlund, Iffat Ahmed, Lars Erik Storbukås, and Hugo Wallenburg

ICCRG 13 November 2017
Motivation

- Many bulk transfer applications do not need to send as fast as they can.
  - data-centre synchronisation
  - client to cloud backups
- They can send in a Less-than-Best-Effort (LBE) way
  - Avoid disrupting more quality constrained flows
Motivation

- Many bulk transfer applications do not need to send as fast as they can.
  - data-centre synchronisation
  - client to cloud backups
- They can send in a Less-than-Best-Effort (LBE) way
  - Avoid disrupting more quality constrained flows

Timeliness

- They often have some loose timeliness requirements
- A need for Deadline-Aware LBE (DA-LBE)
### Deadline Aware Less than Best Effort (DA-LBE)

**Transport qualities**
- Keep disruption of concurrent BE interactive services to a minimum
  - **Do Good** – react to network congestion earlier than a BE service
- Have a timeliness constraint
  - **Be pragmatic** – adjust aggressiveness as deadline approaches
  - **Do no harm** – never more aggressive than a BE type service

---

**Conclusion:** In principle the framework allows any E-to-E congestion control to become DA-LBE

Deadline Aware Less than Best Effort (DA-LBE)

Transport qualities
- Keep disruption of concurrent BE interactive services to a minimum
  - Do Good – react to network congestion earlier than a BE service
- Have a timeliness constraint
  - Be pragmatic – adjust aggressiveness as deadline approaches
  - Do no harm – never more aggressive than a BE type service

Our approach
- Model this behaviour and develop a framework for enabling it

Conclusion: In principle the framework allows any E-to-E congestion control to become DA-LBE

See our first publication:
Network Utility Maximization (NUM)

Kelly [2]
Low and Lapsley [3]


Network Utility Maximization (NUM)

Kelly [2]

Low and Lapsley [3]

CC: Optimisation problem

\[
\max_{x \geq 0} \sum_{s=1}^{S} U_s(x_s)
\]

s.t. \( \sum_{s \in S(l)} x_s \leq c_l \quad \forall l \)

Network Utility Maximization (NUM)

\[ U_1(x_1) \]

Kelly [2]

\[ U_2(x_2) \]

Low and Lapsley [3]

\[ U_S(x_S) \]

From Lagrangian Dual

Send rate, \( x \), proportional to "price" (congestion) and utility (CC)

\[ x_S = \left( (U_S)' \right)^{-1} \left( \sum_{l \in L(s)} p_l \right) \]

Congestion "price":
- loss, ECN,
- delay, etc

Network Utility Maximization (NUM) for LBE

Kelly [2]

Low and Lapsley [3]


DA-LBE with a homogeneous congestion control network

Congestion “price” inflation

\[ \hat{q} = \frac{\sum p_I}{w} \quad \text{where} \quad w \in [w_{\text{min}}, 1] \]

- when \( w = w_{\text{min}} \), maximum price inflation, maximum “LBEness”
- when \( w = 1 \), no inflation, BE service.
DA-LBE with a homogeneous congestion control network

**Congestion “price” inflation**

\[ \hat{q} = \frac{\sum p_l}{w} \text{ where } w \in [w_{\text{min}}, 1] \]

- when \( w = w_{\text{min}} \), maximum price inflation, maximum “LBEness”
- when \( w = 1 \), no inflation, BE service.

**Controlling price with respect to deadlines**

On short packet time scales:
- CC reacts to \( \hat{q} \) as normal

On longer time scales adjust \( w \):
- Relative to:
  - recent send rate: \( \bar{x} \)
  - required send rate: \( \zeta \)
- PID or Model based control
Applying this to TCP Cubic

For TCP Cubic, price is packet loss or ECN packet marks

Indirect: inflate response

Direct: inflate price

\[ \text{cwnd new} = \beta \times \text{cwnd} \]

\( \beta \) varies,
\( \beta \in [\beta_{\text{min}}, \beta_{\text{default}}] \)

\( \beta_{\text{min}} \) provides maximum LBEness

Direct: inflate price

Drop additional packets

Lose data causes retransmissions

Phantom ECN signals

Same congestion response

No loss in data
Applying this to TCP Cubic

For TCP Cubic, price is packet loss or ECN packet marks

**Indirect: inflate response**

- Change cwnd reduction
  - \( cwnd_{new} = \beta cwnd \)
  - Vary \( \beta, \beta \in [\beta_{min}, \beta_{default}] \)
  - \( \beta_{min} \) provides maximum LBEness

**Direct: inflate price**

- Drop additional packets
  - lose data
  - *causes retransmissions*

- Phantom ECN signals
  - same congestion response
  - *no loss in data*
Simple Scenario Experiments

Scenario

- 6 TCP flows start and stop at different overlapping times
  - No competing TCP flows \( t=[1000,1010] \) s
- DA-LBE flow file size equivalent to 10% capacity to deadline
- 10% random background traffic
TCP Cubic with a Cubic based DA-LBE — varying $\beta$

Lack of LBEness

The loss signal is not frequent enough
TCP Cubic with a Cubic based DA-LBE — Phantom ECN

Unable to take full advantage of available capacity \((t = 1000 \text{ s})\)

Delay-based CC may detect congestion or lack of it more quickly
Different network prices

- Tang, Wei, Low, and Chiang [5] maps prices to a standard price (or congestion signal).
  - E.g. mapping a packet delay “price” to a standard packet loss price

Issues

- Tang, Wei, Low, and Chiang required a special factor to make this work.
- More than mapping prices, CCs react differently to congestion signals

### Different network prices

- Tang, Wei, Low, and Chiang [5] maps prices to a standard price (or congestion signal).
  - E.g. mapping a packet delay “price” to a standard packet loss price

### Issues

- Tang, Wei, Low, and Chiang required a special factor to make this work.
- More than mapping prices, CCs react differently to congestion signals

### We build on this idea

- Composite congestion signals (delay, loss, and ECN)
- Weight ($\phi$) composite congestion signals by CC reaction
- We use a weighted $P[\text{cong\_ind}]$ to compare “prices”

---

Applying this to TCP Vegas

**Delay based part**
- Congestion signal:
  - Estimate of queueing delay ($Q$)
- Control:
  - $cwnd++$ or $cwnd--$

**Loss based part**
- Halve $cwnd$ on packet loss
Applying this to TCP Vegas

**Delay based part**
- Congestion signal:
  - Estimate of queueing delay ($Q$)
- Control:
  - $cwnd++$ or $cwnd--$

**Loss based part**
- Halve cwnd on packet loss

Vegas based DA-LBE

**Delay based part**
- inflate (or deflate) queueing delay
  - \( \hat{q} = \frac{Q}{\phi w} \)

**Loss based part**
- When $w = 1$ and packet loss
  - probabilistically ignore cwnd reduction
  - $\text{rand}() < (1 - \frac{1}{w\phi})$
TCP Cubic competing with a Vegas based DA-LBE

Able to take full advantage of available capacity \((t = 1000 \text{ s})\)

- Delay-based CC detects congestion or lack of it more quickly

\[(\text{MBC, } T_w = 10, T_\phi = 60 \text{ s})\]
TCP Cubic competing with a Vegas based DA-LBE

Does not quite meet deadline in this scenario
- But does no harm
- LBEness relatively good

(MBC, $T_w = 10$, $T_\phi = 60s$)
Completion time experimental setup

\[ N_1 \xrightarrow{T \text{mix}} N_2 \]

\[ N_1 \xleftarrow{1 \text{ Gbps}} N_2 \]

\[ N_1 \xleftarrow{100 \text{ Mbps}} N_2 \]

\[ N_1 \xrightarrow{10 \text{ ms}} N_2 \]

DA-LBE Tx \rightarrow DA-LBE Rx

Cubic

ICCRG 13 November 2017
Congestion metacontrol for DA-LBE

davidh@simula.no 12 / 17
Completion time experimental setup

Based on real measurements
Models application interaction

TCP connections come and go
Hundreds of concurrent connections
Completion time experimental setup

Tmix in Experiment
- Shuffle traces to remove non-stationarity
- Application session start times

- Scale application session arrival time to achieve a particular average offered load.
Completion time experimental setup

Experiment comparing Cubic and Vegas based DA-LBE

- NS2
- Deadline 1200 s
- > 60 000 s
- DA-LBE flows restart 10 s after completing
- File size equivalent to 10% of Capacity over 1200 s
DA-LBE Completion time results

![Chart showing completion time results for Cubic based DA-LBE and Vegas based DA-LBE.]

- Cubic based is more consistent, but less able to use available capacity
- 75% offered load is near congestion “knee”
  - Both do not always meet deadlines

(MBC, $T_w = 10$, $T_\phi = 60$ s)

ICCRG 13 November 2017
Congestion metacontrol for DA-LBE

davidh@simula.no 13 / 17
DA-LBE Completion time results

We have also looked at:
- impact of completion times on other traffic
- relative fairness, etc.

Refer to next publication :)
Stand alone DA-LBE in Linux (*mostly* working prototype)

Application

libdalbe

open()
close()
socket()
close()
stats
control

Base

Metacontrol thread

kernel

TCP

DA-LBE Stat

DA-LBE Ctl
DA-LBE in NEAT

What is NEAT?
- A new transport API (see work in the TAPS WG!)
  - applications request the service they need
  - agnostic to the specific choice of transport protocol underneath
- Allows deployment of new (and better) transports

Read more about it: https://www.neat-project.org/

DA-LBE in NEAT

What is NEAT?

- A new transport API (see work in the TAPS WG!)
  - applications request the service they need
  - agnostic to the specific choice of transport protocol underneath
- Allows deployment of new (and better) transports

Read more about it: https://www.neat-project.org/

DA-LBE in NEAT

What is NEAT?

- A new transport API (see work in the TAPS WG!)
  - applications request the service they need
  - agnostic to the specific choice of transport protocol underneath
- Allows deployment of new (and better) transports

Read more about it: https://www.neat-project.org/


DA-LBE in NEAT

- DA-LBE will be implemented as a meta-protocol in NEAT
- NEAT choosing the best underlying transport to adapt
DA-LBE meta-protocol in NEAT (work in progress)

NEAT system

NEAT Library

other

DA-LBE Meta

Policy Manager

Policy Information Base

Characteristic Information Base

Sockets

TCP

Kernel

DA-LBE Stat

DA-LBE Sig

Congestion metacontrol for DA-LBE

davidh@simula.no

ICCRG 13 November 2017
## Conclusions

**Deadline-Aware–Less-than-Best-Effort (DA-LBE)**
- valuable transport for bulk data transfers
  - soft deadline
  - disruption of other traffic minimised

**In principle allows *any* congestion control to become DA-LBE**
- Concepts based on NUM
  - *inflate* (or *discount*) network “prices” to achieve goals.
- Tested with TCP Cubic and Vegas
  - Delay based mechanisms generally perform better
  - Immediate ECN would have benefits of delay based mechanisms

**Ongoing work**
- Integration into NEAT
- Trans-Internet tests
- Modularisation of kernel elements

---

*This work has received funding from the European Union’s Horizon 2020 research and innovation programme under grant agreement No. 644334 (NEAT). The views expressed are solely those of the authors.*
Bibliography


