TCP’s protocol radius

Cathryn Peoples
University of Ulster

Co-authors: Dr. Lloyd Wood, Prof. Gerard Parr, Prof. Bryan Scotney, Dr. Adrian Moore

IWSSC ’07
Salzburg, Austria
September 2007

Consider the following transmission scenario

- A ground station on Earth wishes to communicate with a satellite orbiting Mars.
- What transport protocol can be used to perform the communication?

TCP doesn’t work over very long distances

Current TCP protocols have very poor performance in the Interplanetary Internet.
Akan, Fang, Akyildiz,
TP-Planet: A Reliable Transport Protocol for Interplanetary Internet, **IEEE Journal on Selected Areas in Communications, February 2004**

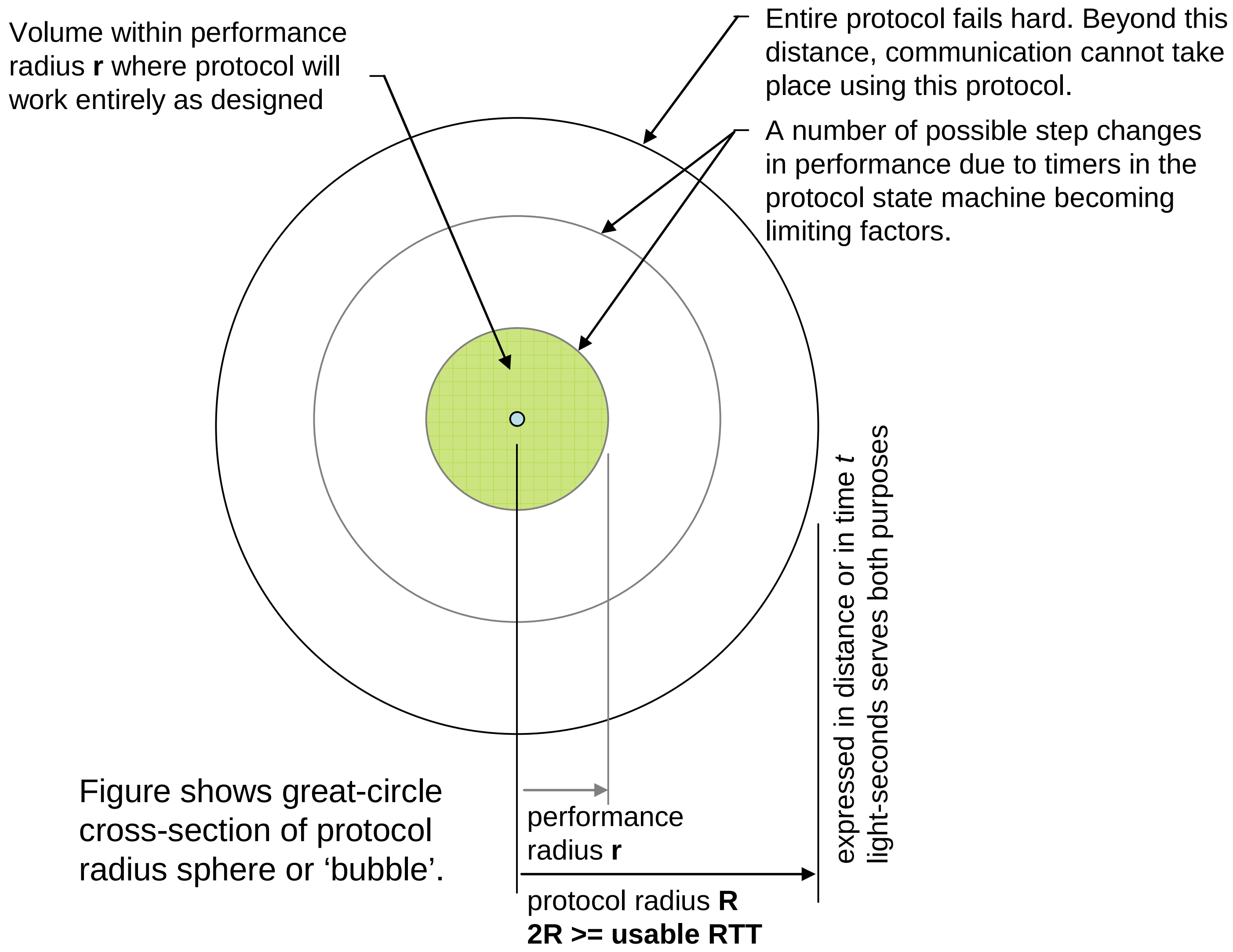
…once a spacecraft is **more than one minute away** (in terms of light-trip time), then every attempt to establish a TCP connection will fail.
Farrell, Cahill, *et al.,*
When TCP Breaks**, Internet Computing, August 2006**

For a two-minute timer, you need to get to the receiver and back again to the sender, so halve the distance…
but is that when TCP *really* breaks?

Timers affect protocol performance

- The distance any protocol can communicate is limited by *physical* signal strength and *logical* timers – how long the sender waits before giving up.
- Translation between timers' time and distance is straightforward – **use speed of light** in vacuum (light-seconds).
- It can be hard to see the effects of timers, due to interactions of multiple timers at multiple layers (link and transport).

Experiments attempt to quantify protocol performance in terms of operational ranges



Volume within performance radius **r** where protocol will work entirely as designed

Entire protocol fails hard. Beyond this distance, communication cannot take place using this protocol.

A number of possible step changes in performance due to timers in the protocol state machine becoming limiting factors.

expressed in distance or in time *t*
light-seconds serves both purposes

performance radius **r**

protocol radius **R**
2R >= usable RTT

Figure shows great-circle cross-section of protocol radius sphere or ‘bubble’.

Experiment design

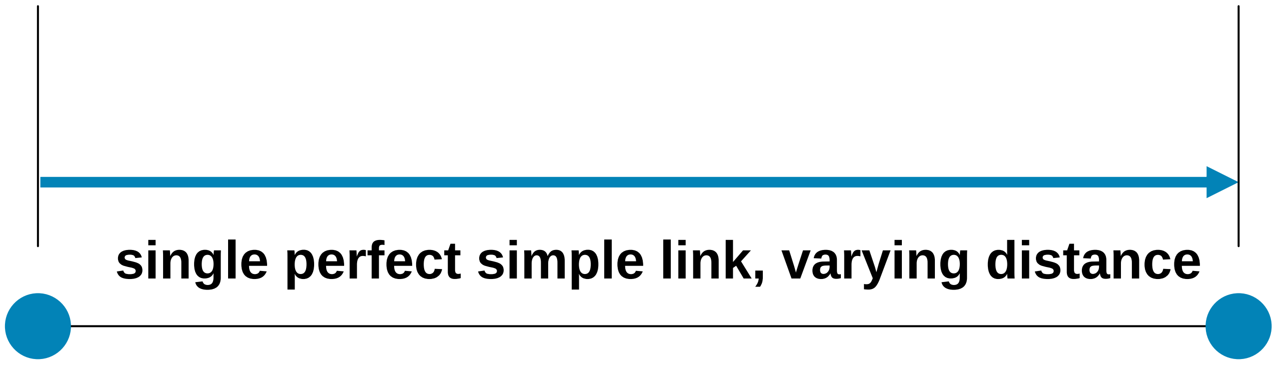
- In our experiments:

  Deliberately set up a really simple simulation scenario, using TCP over a simple serial link.

  No MAC or link timers. Only TCP timers to look at.

  No errors/losses, so we can examine timer behaviour without introducing noise/inducing backoff reactions.

Simulation scenario



single perfect simple link, varying distance

TCP sender

TCP receiver

TCP Simulation Scenario in Opnet

Opnet
11.5

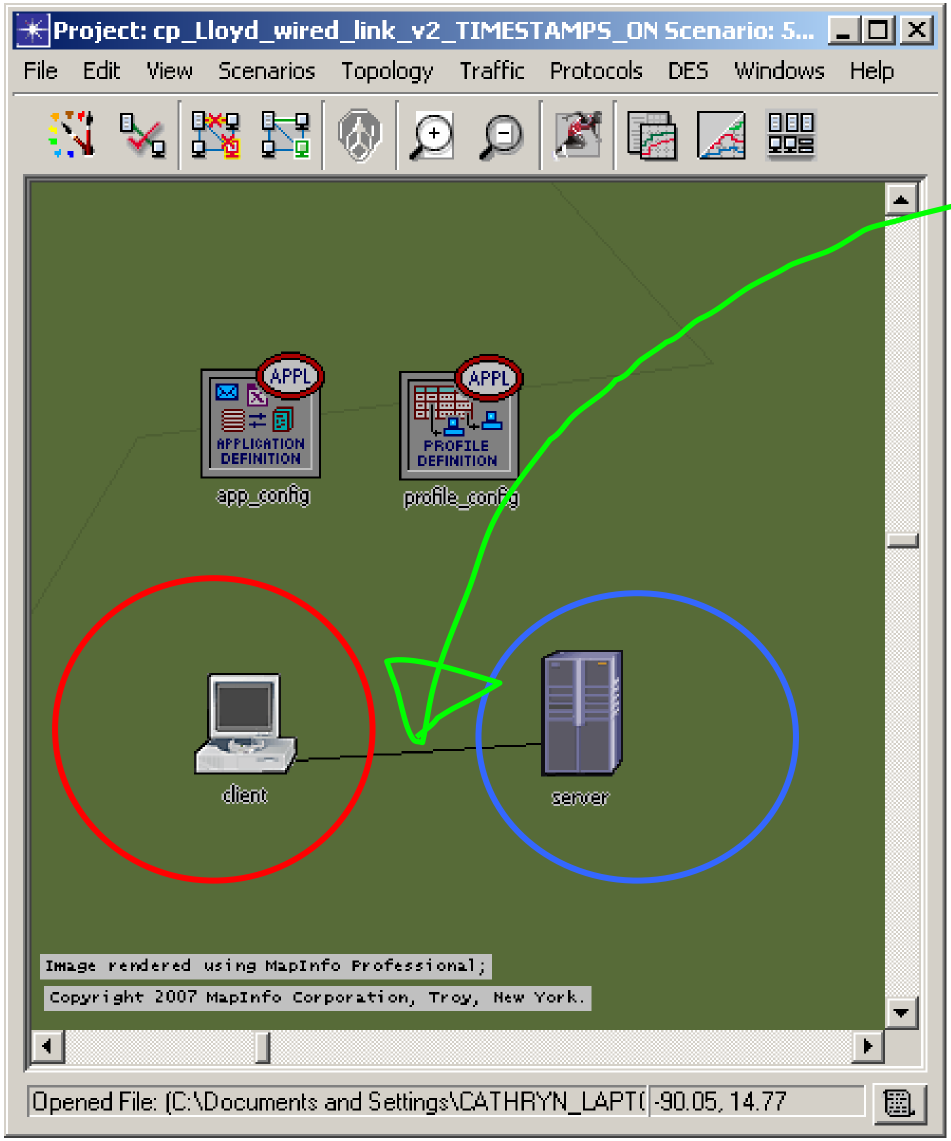


Image rendered using MapInfo Professional;
Copyright 2007 MapInfo Corporation, Troy, New York.

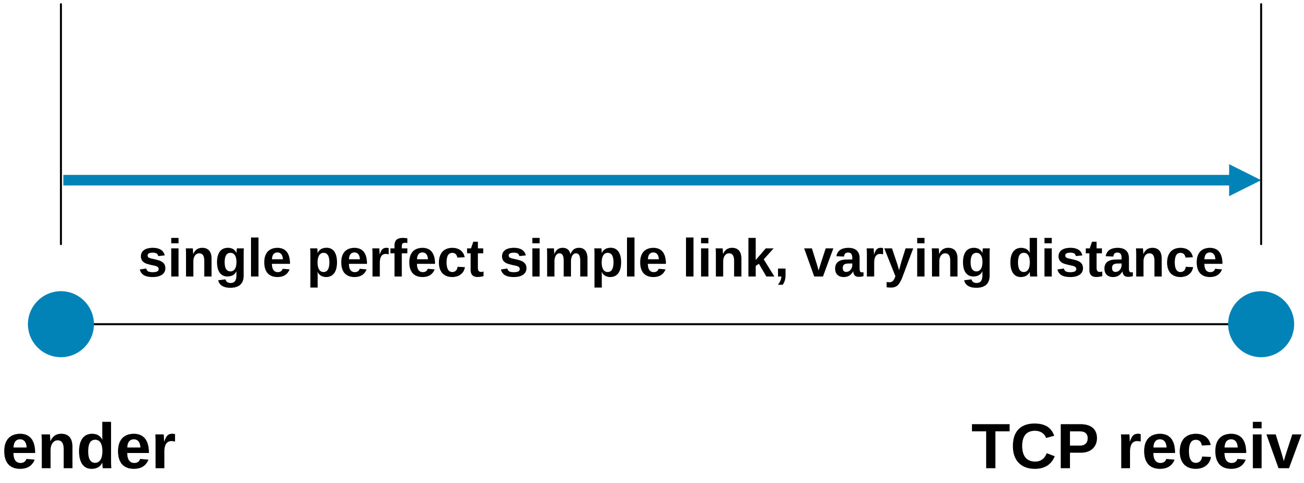
Opened File: (C:\Documents and Settings\CATHRYN_LAPT(-90.05, 14.77

PPP Link

client

server

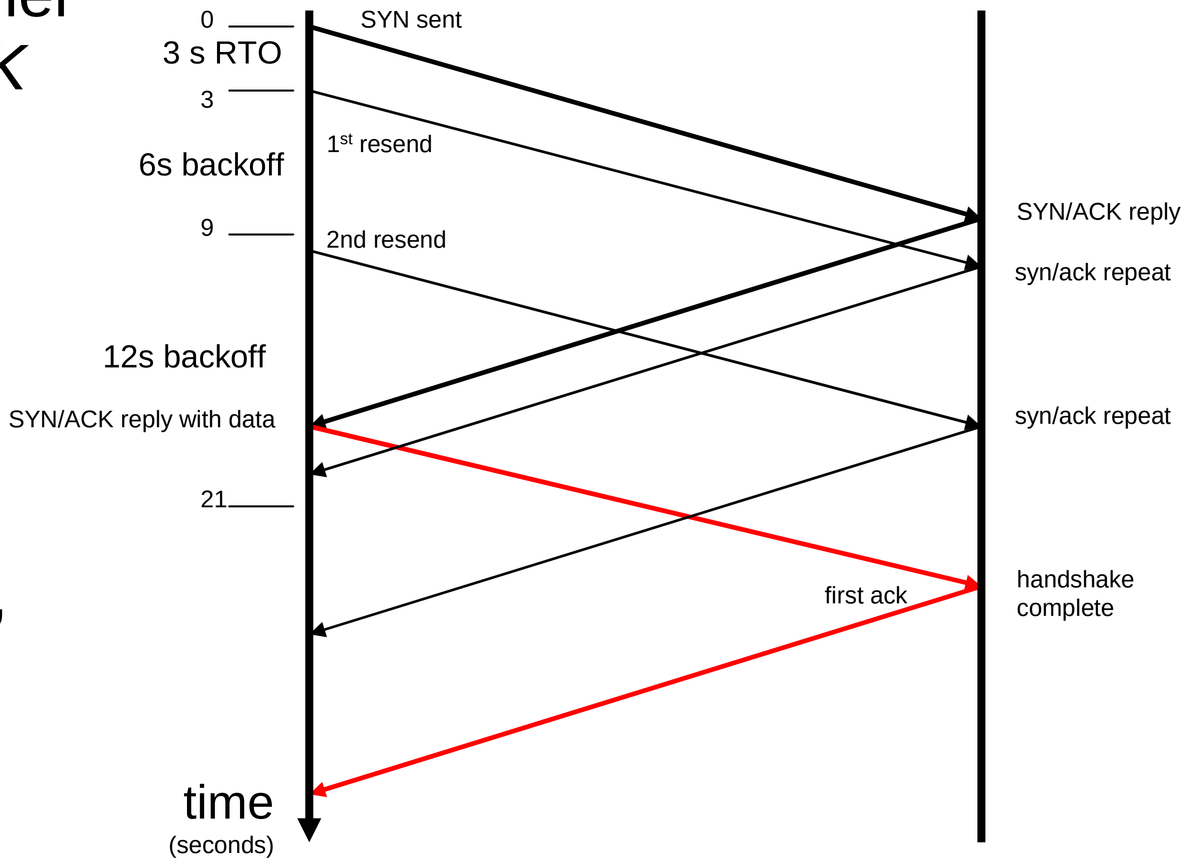
Simulation scenario



- Simulated using both *ns* and *Opnet*.
- Altered distance between nodes (up to distance of 30 seconds), reran simulation for different TCP variants (Reno, SACK, and timestamps). Thousands of simulations.
- Looked at time to transfer a file (variable packet sizes up to 500,000 bytes) to determine where TCP breaks.

What we found – limits to communication

- TCP's *SYN/ACK* setup is determining factor for distance. If the *SYN* timer gives up before an *ACK* response comes in, transfer never starts.
- *SYN* timer is implemented as 3 seconds with doubling exponential backoff – sends a *SYN*, waits 3s, sends another *SYN*, waits 6 seconds…
- Any *SYN/ACK* coming back will do; first seen as response to a later *SYN*.



Eventually, TCP quits sending *SYN*s

- *Opnet* TCP fails to transmit after 5 *SYN*s – 3+6+12+24 = 45s
- Got to get a response back, so 45/2 = 22.5 light seconds, or 6.7 million kilometers. If *SYN/ACK* is sent before 22.5s and received before 45s, session starts.
- *ns* **never** gives up.
- Implementations give up earlier – Microsoft sends just two *SYN*s for a 9s total timeout and a 4.5 light-second distance[1]. That is still 1.3 million km – TCP will work (*very* poorly) out to Moon and lunar Lagrange points.
- *SYN/ACK* sets limit on range – **TCP's protocol radius**.

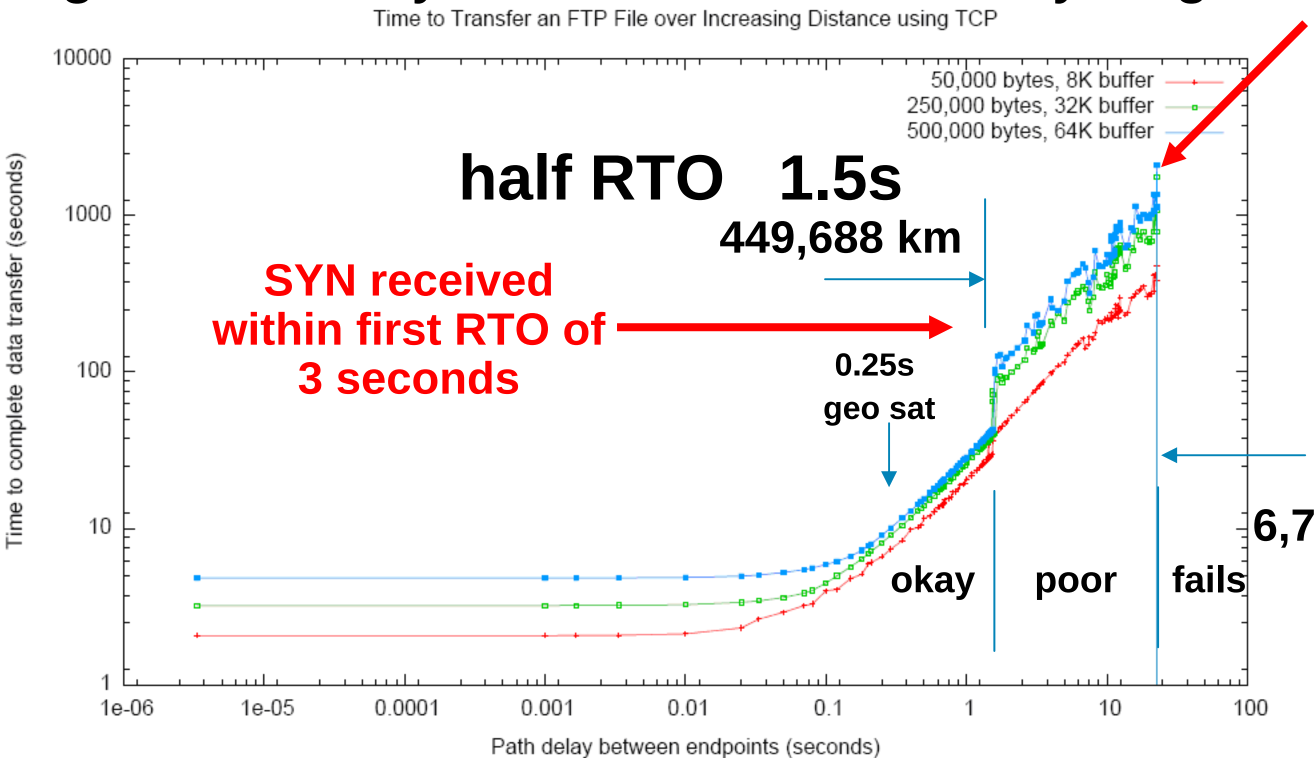
[1] Microsoft Windows 2003 TCP/IP Implementation, TechNet, Microsoft Corporation, June 2006.

Found a step change in TCP's *performance*

- File transfers take longer with longer distance. But it's not linear, due to TCP window behavior.
- Governed by TCP's retransmission timeout (RTO) value, which defaults to 3 seconds. The Internet is normally less than 1.5 seconds across end-to-end, so that's okay.
- TCP over geostationary satellite is in the 'okay' region.

log/log graph

time to transfer complete file *vs* path delay or distance



half RTO 1.5s
449,688 km

SYN received within first RTO of 3 seconds

0.25s geo sat

5th SYN fails to be received within timeout period

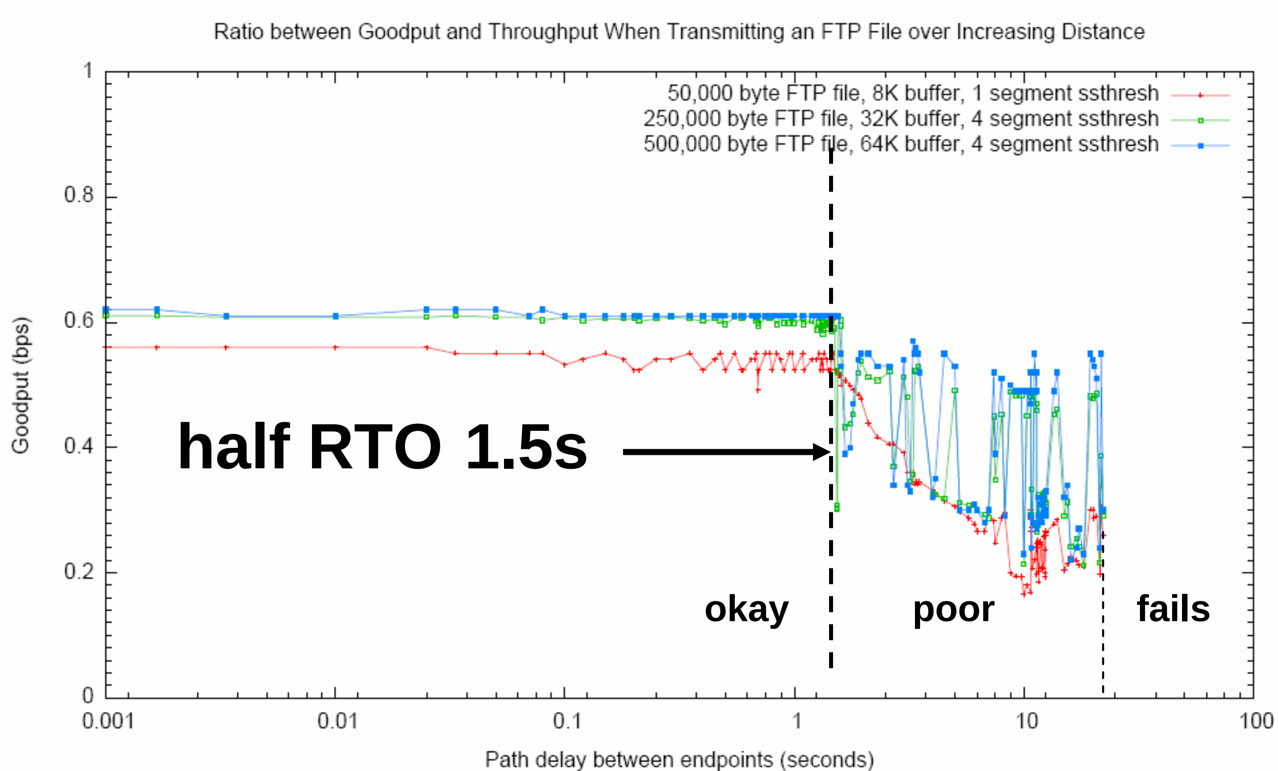
22.5s
6,745,320 km

Found a step-change in TCP's *goodput*

- Goodput/throughput ratio gives scalable view of performance.
- Goodput degrades beyond 1.5 seconds.
- Variations in delay due to crude timer granularity in Opnet
- Results are independent of file size, buffer size or *ssthresh* slow-start threshold.

lin/log graph

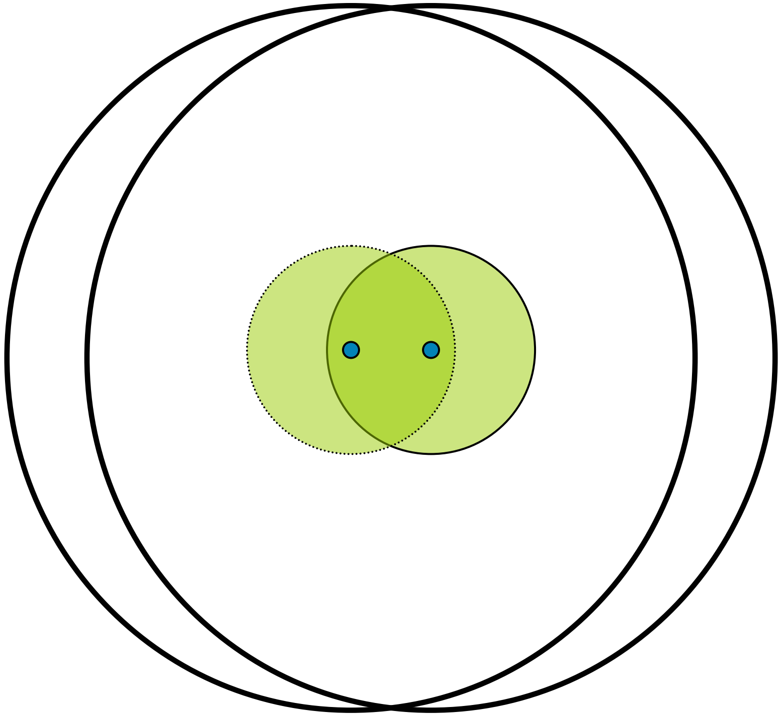
ratio between goodput and throughput *vs* path delay or distance



TCP performance alters with distance

🟢 **inner performance radius**

○ **limiting performance radius**



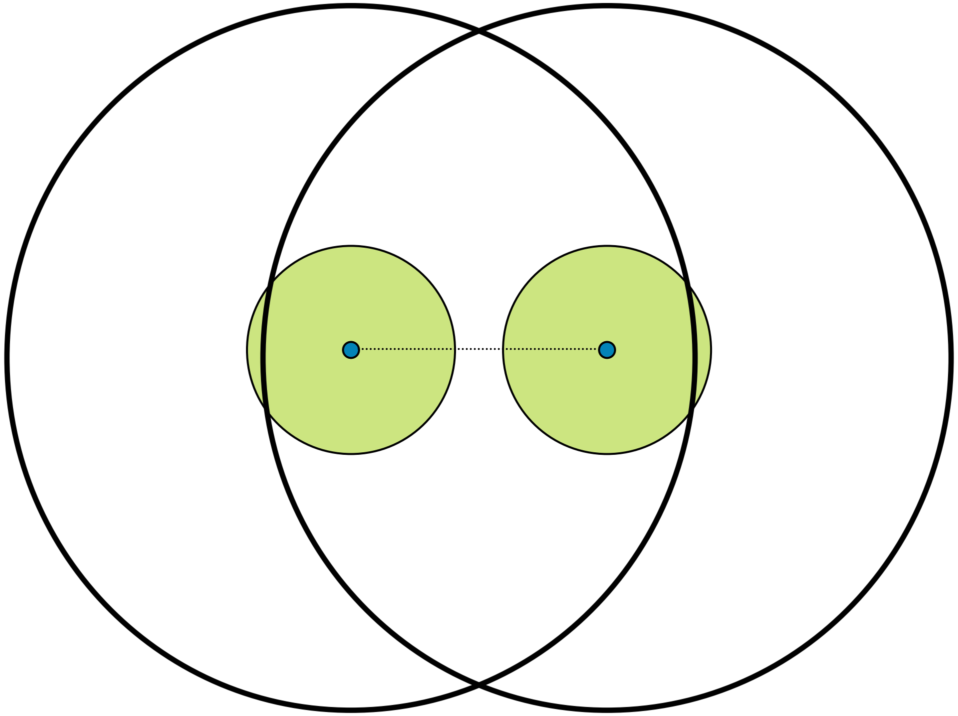
okay

highest performance –
within inner performance radius
(for TCP this is 3s RTO – 1.5s distance)

TCP performance alters with distance

● inner performance radius

○ limiting performance radius



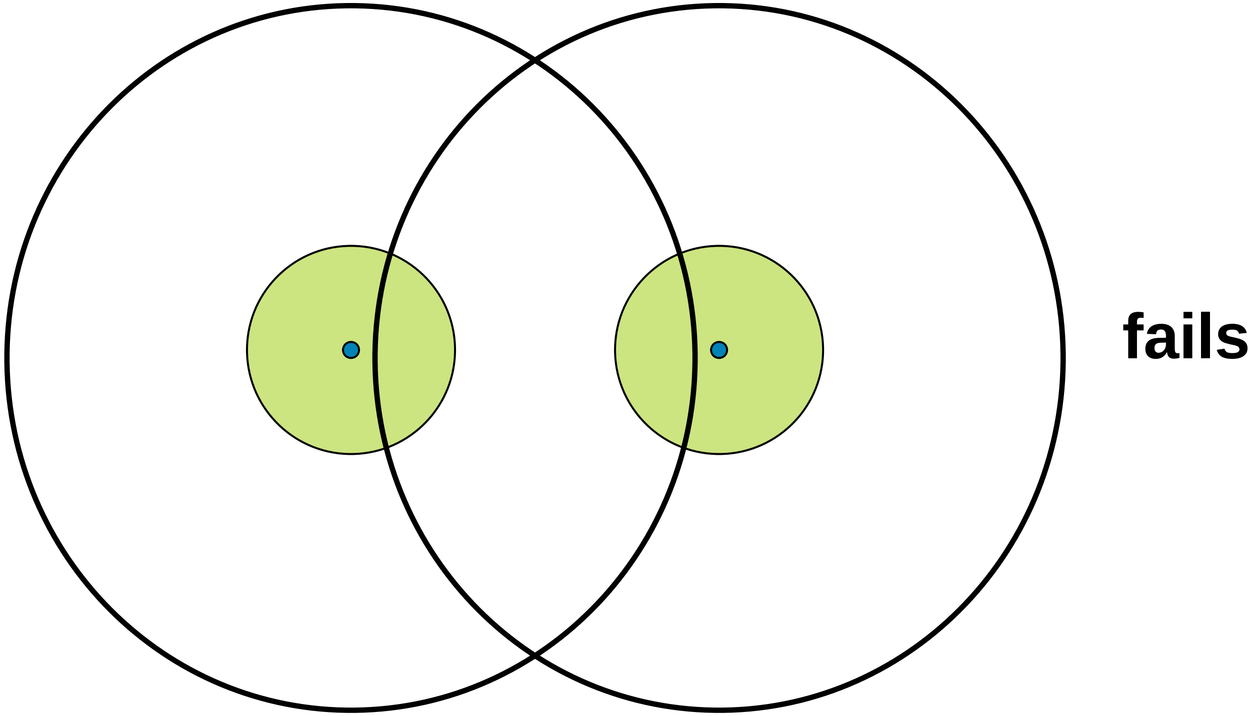
poor

**step change to range of lower performance –
still within bounding protocol radius**

TCP performance alters with distance

● inner performance radius

○ limiting performance radius



TCP fails –
path distance is now beyond bounding protocol radius
(SYN/ACK exchange times out)

How does this apply to other protocols?

- Looked through IETF protocols for timer dependencies and default values that limit distance. Routing protocols, BGP, even Mobile IP – *everything* has timers. Everything is distance-limited at a logical level.
- Would like to simulate 802.11 performance to find limits.
- But, even with TCP, we found differences between simulators that affected results.
- Wireless simulators not matching standards or each other is now well-known; new detailed papers comparing 802.11 simulators, and pointing out problems.
- It will be a while before clear conclusions about timer limitations can be drawn for complex link protocols.
- Optimising protocols to perform as well as possible across their operating ranges is a promising area – *e.g.* TCP has a max RTO of 64s. Is that reasonable, or just too large?

How can this information be used?

- An understanding of a protocol’s radius can help to influence decisions made by context-aware applications

- Friday 14th September
- 14:00
- TRACK III
- A Reconfigurable Context-Aware Protocol Stack for Interplanetary Communication
- Presenter: Cathryn Peoples

Questions?

Thankyou.