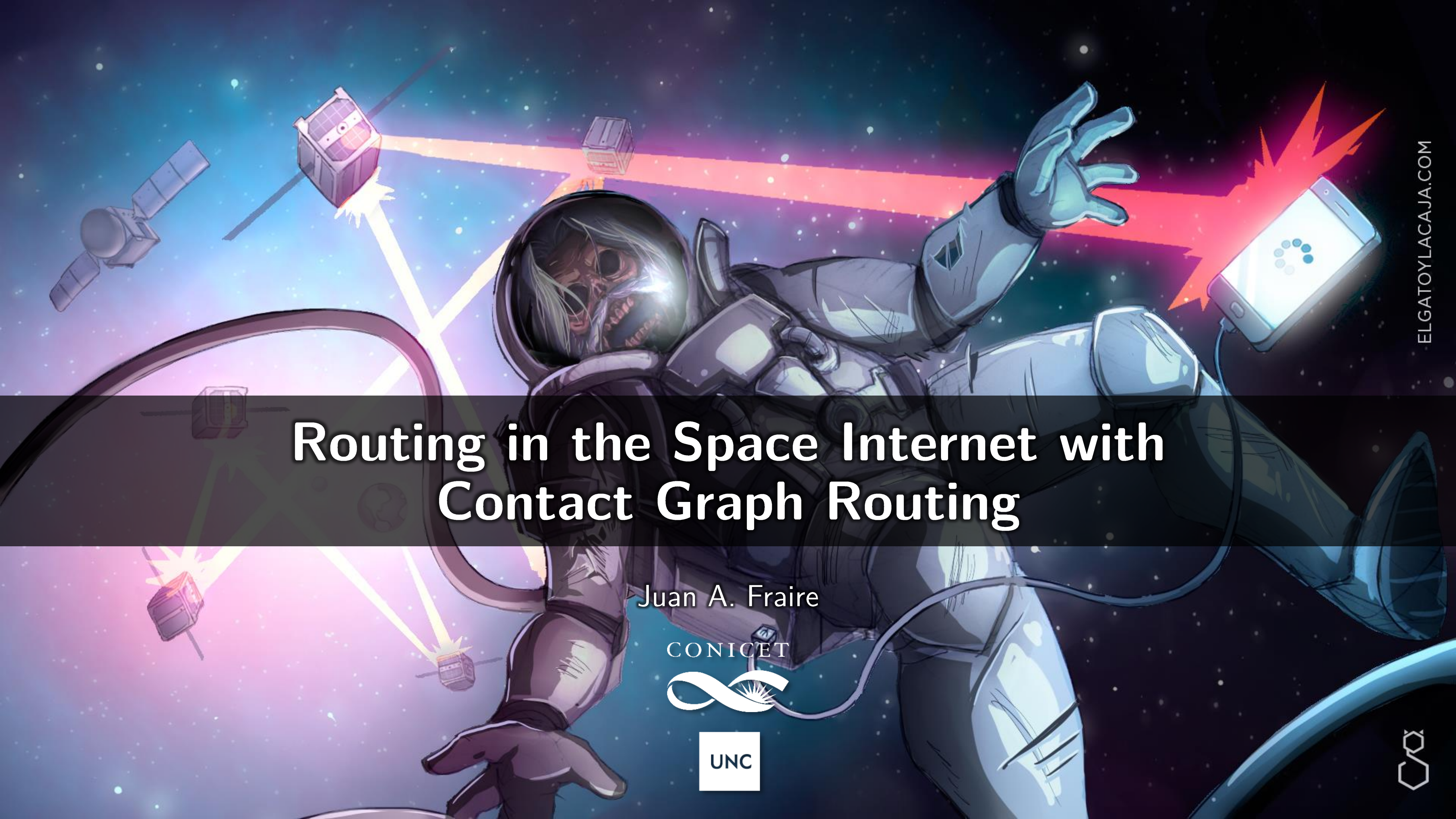


Routing in the Space Internet with Contact Graph Routing

Juan A. Fraire

CONICET

UNC

Agenda

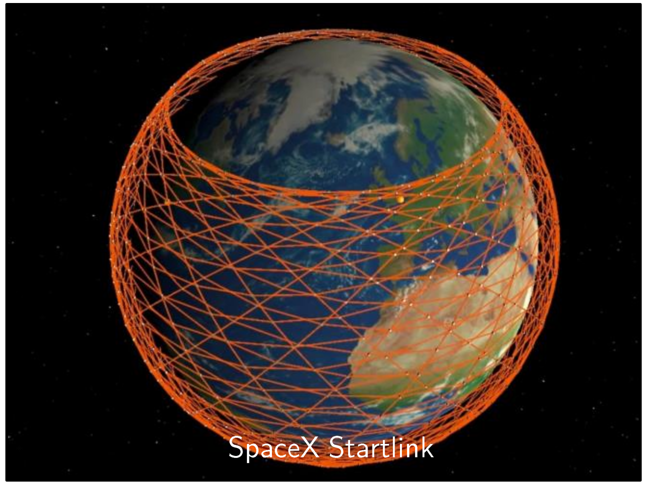
- Introduction
- Space Networks
  - Model
- Contact Graph Routing
  - Route Search
  - Route Management
  - Forwarding
- Trends and Outlook

Introduction

New Space

- Enabling technologies
  - Formation Flying & ISL
  - COTS components
  - SmallSat Paradigm
  - Cheap Launch
- Networked constellations
- Deep-Space exploration

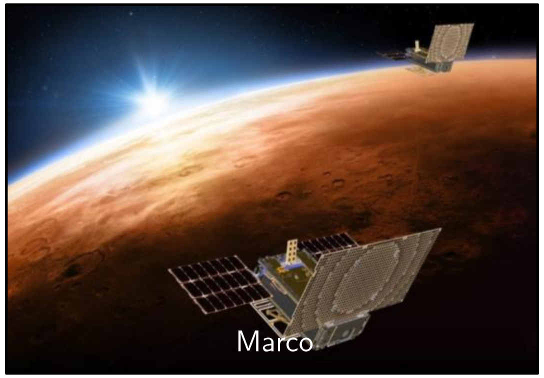
- Small satellites (~500kg)
- Nano satellites (~10kg)


SpaceX Startlink

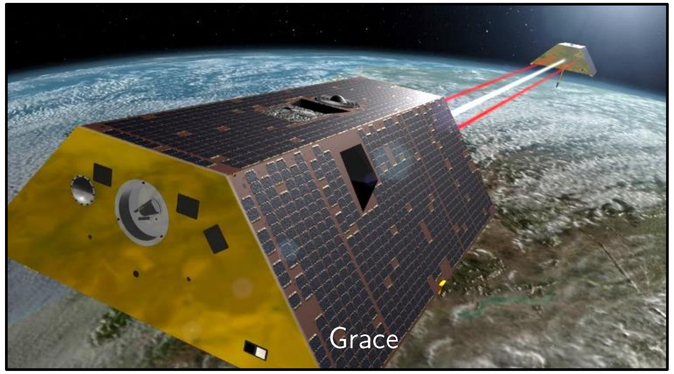

Tango & Mango


GomX-4


Lunar IceCube


Marco


Planet

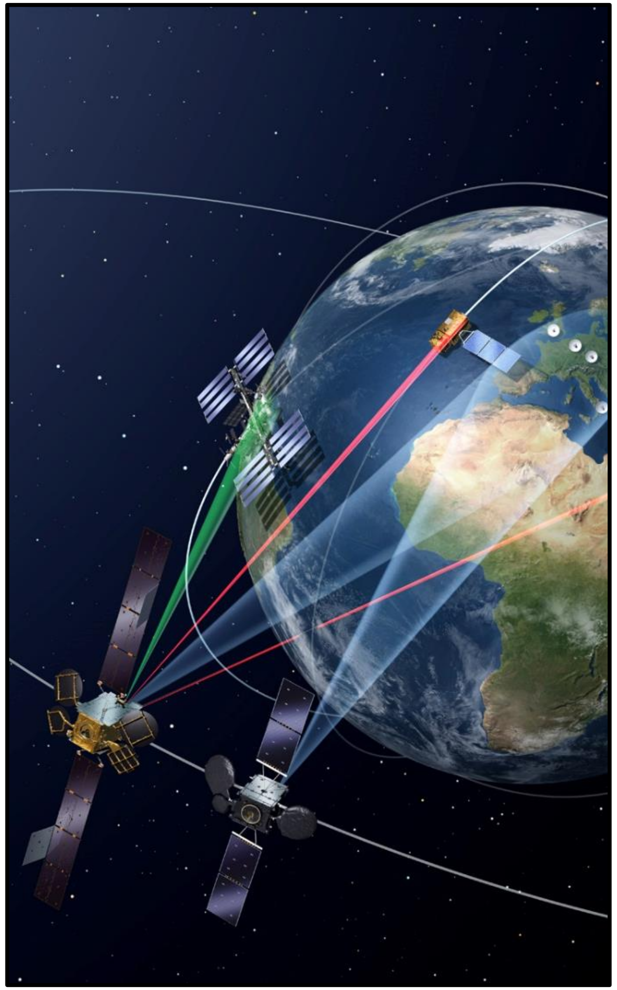

Grace

Introduction

Space Internet

- Space Internet (ground, near-Earth and deep-space)
  - Extended connectivity times
  - Reliability via multiple communication paths
  - In-orbit resource and service sharing
- Fundamental environmental challenges
  - Orbital dynamics
  - Varying and long-range distances
  - Effect of planet rotation
  - On-board power restrictions
  - Propagation delay in deep-space
  - Asymmetric or unidirectional links

Delay and Disruption



Introduction

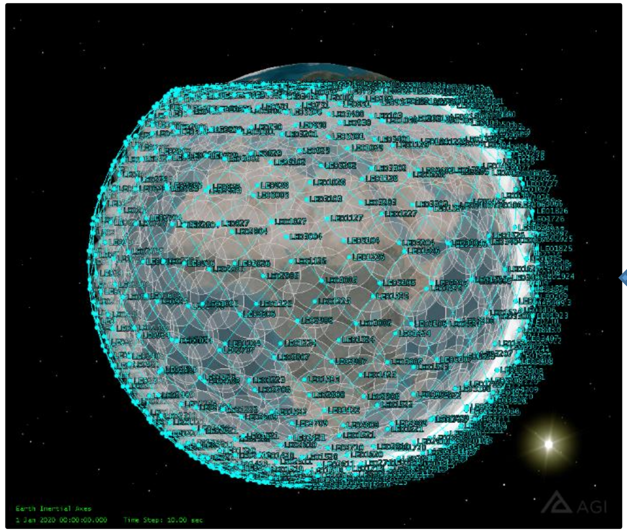
Space Internet Protocol Architecture

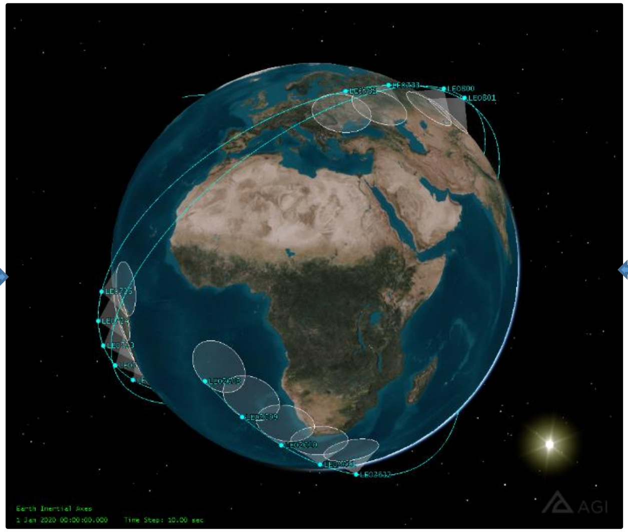
- Internet protocols (TCP)
  - Largely based on an effectively instantaneous flow of information between sending and receiver nodes.
- Delay Tolerant Networking (DTN)
  - Assumes no instantaneous feedback but potentially lengthy storage of data in intermediate nodes

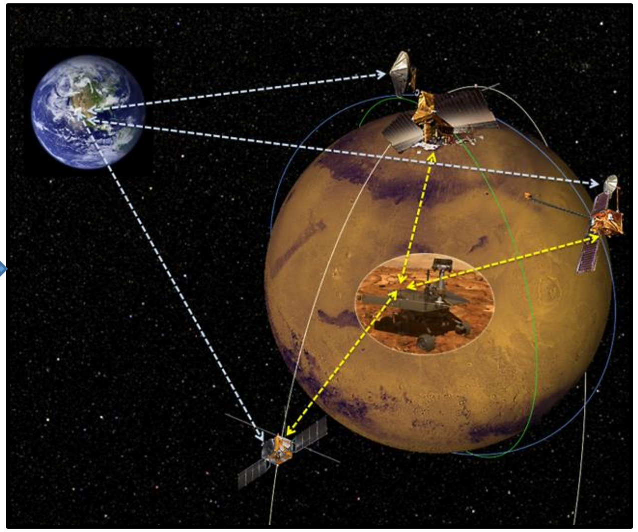
Telephony model

Epistolary model

Internet





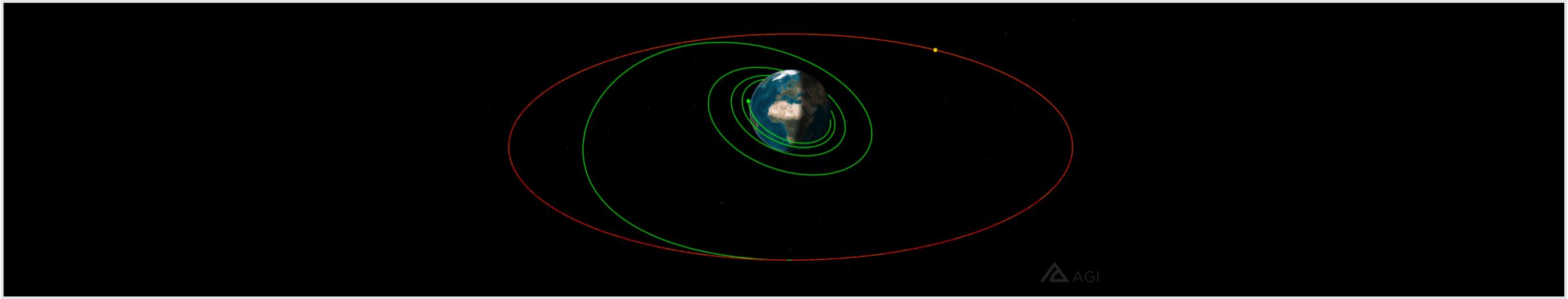


DTN

Introduction

Routing in Internet vs DTN

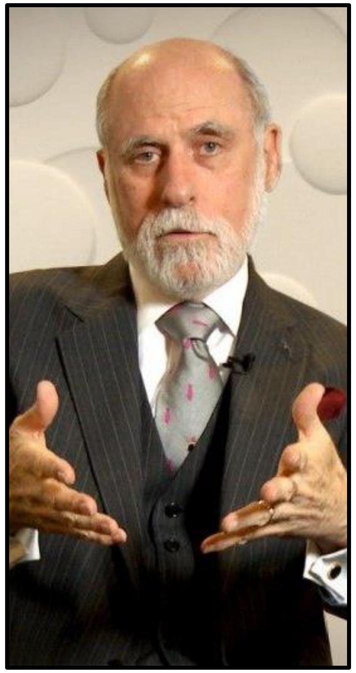
- Internet routing schemes are based on stable connections
  - Disruptions are considered an *error*
- DTN routing schemes are based on unstable connections
  - Disruptions are a *natural* part
  - Get *next hop* and *when* to send depending on *when* it is expected to arrive:
  - Space Networks: possible to know the future connectivity (*contact plan*)
    - **Contact Graph Routing (CGR)**



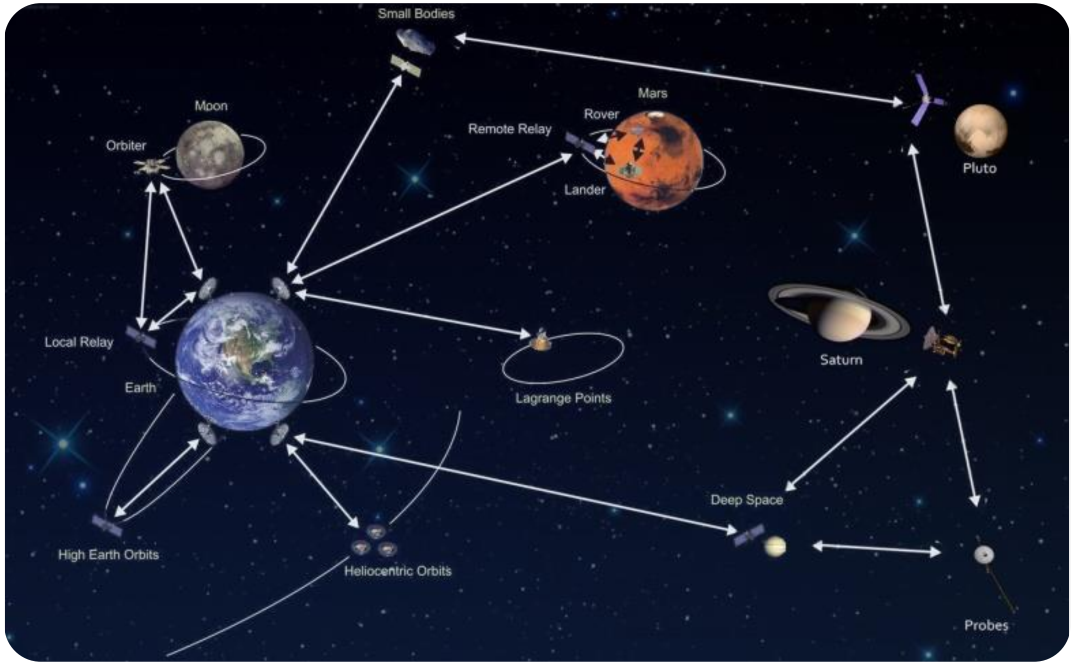
Space Networks

DTN Origins

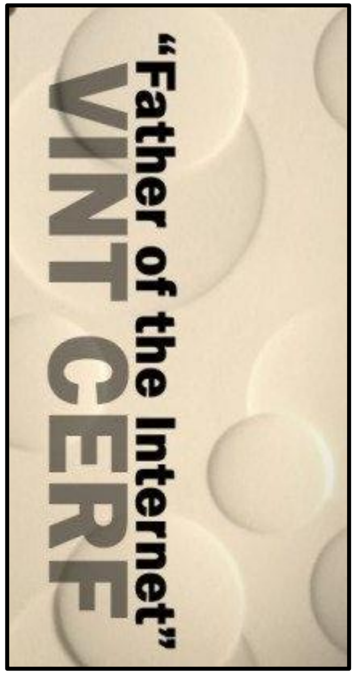
- DTN originated from the Interplanetary Internet project
  - Introduced by a team led by Vint Cerf in 2003
  - Motivated other applications (vehicular, social, underwater, IoT)



Propagation delay to…
Mercury
3 min
Venus
6 min
Mars
12 min
Jupiter
43 min



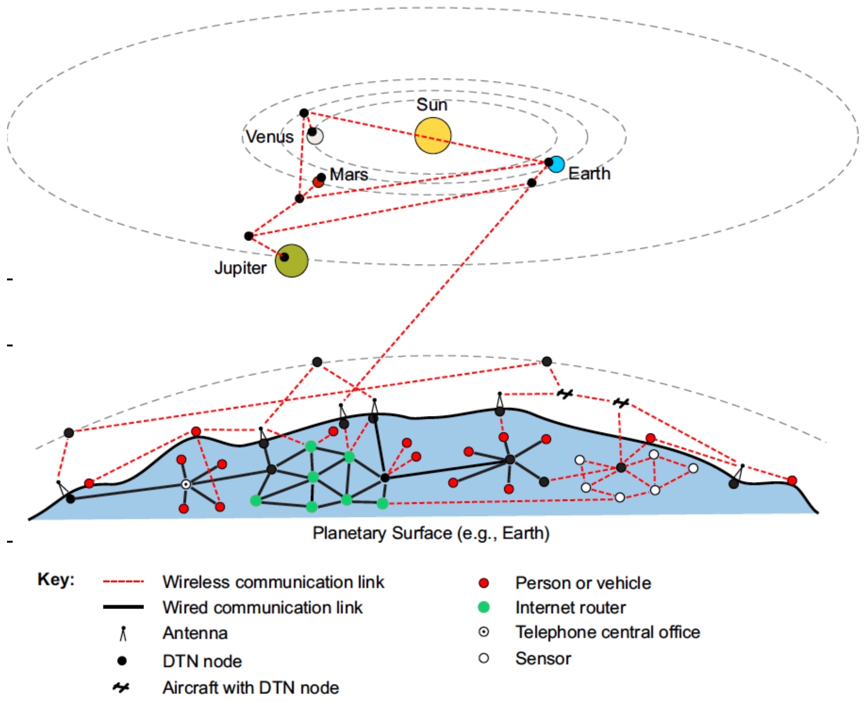
Propagation delay to…
Saturn
1h20
Uranus
2h40
Neptune
4h10
Voyager-1
19h30



Space Networks

DTN Taxonomy

- Opportunistic
  - No assumptions can be made on future contacts
- Probabilistic
  - Contact patterns can be inferred from history
- Scheduled
  - Contacts can be accurately predicted
    - Deep-Space: Links *disrupted* and *delayed*
    - Near-Earth (LEO, MEO, GEO): Links *disrupted*

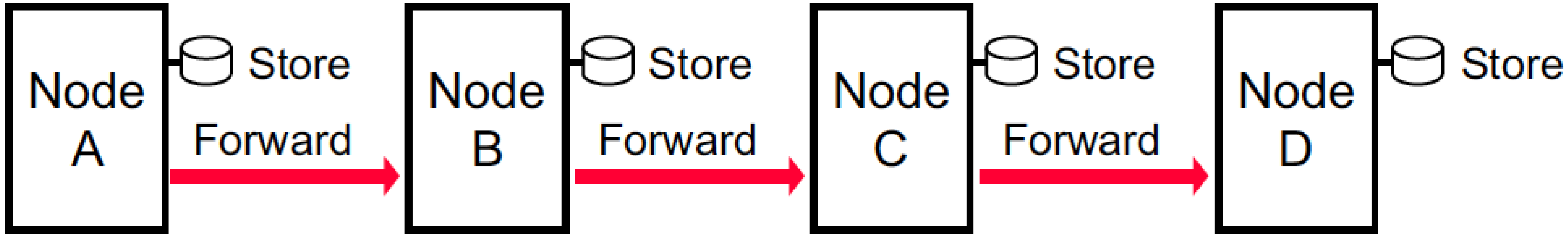


Space Networks

DTN Protocols and Procedures

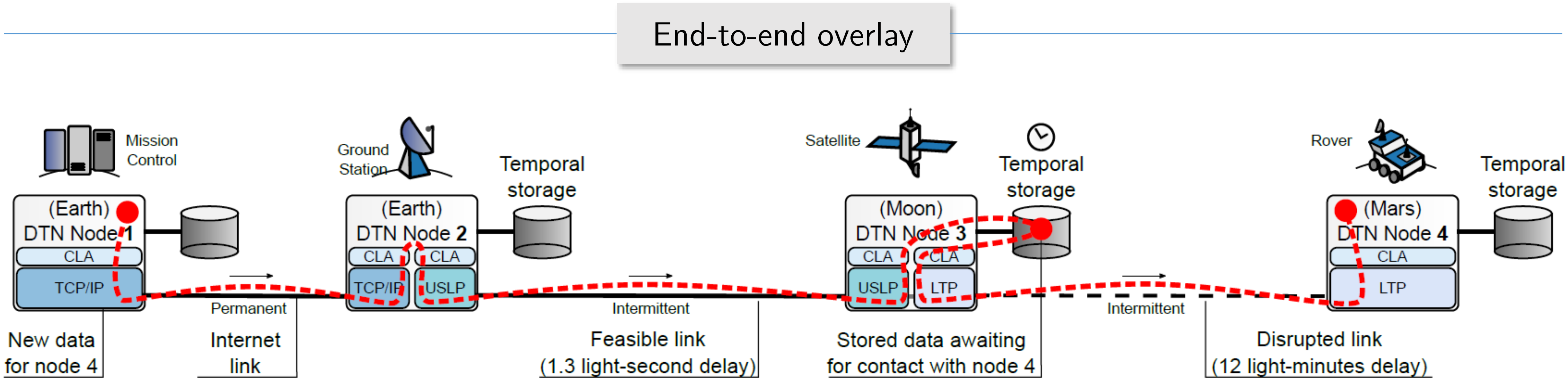
- Principles
  - Store-carry-and-forward
  - Minimal end-to-end messaging exchange (feedback)
- Architecture → Bundle layer (overlay)
  - Bundle protocol
    - Persistent storage resources (in contrast to Internet buffering)
    - Encoding and block format of bundles (BP data units)

- RFC 4838
- RFC 5050
- CCSDS BP
- CCSDS SABR



Space Networks

DTN Protocols and Procedures



	Mercury	Venus	Earth	Mars	Jupiter	Saturn	Uranus	Neptune
Avg. Distance to Sun (UA)	0,38	0.72	1.00	1.52	5.21	9.54	19.18	30.11
Latency	3 min	6 min	8 min	**12 min**	43 min	1h20	2h40	4h10

Space Networks

Bundle Protocol Features

- Convergence Layer Adapters
  - Adapt session, encoding, MTU to underlaying TCP, UDP, Bluetooth, Ethernet...
- Custody Transfers
  - Nodes can *take custody* to enhance reliability and reduce congestion (storage)
- Fragmentation
  - Bundles have no size limit, fragmentation to multiplex and fit contacts
- Priorities
  - Traffic priority (unicast) and *critical* flag (forward copies to all neighbors)
- Deadlines
  - Real Time to Live (TTL), to avoid congestion

Space Networks

Bundle Protocol Data Unit Parameters

Bundle parameters (relevant to routing)

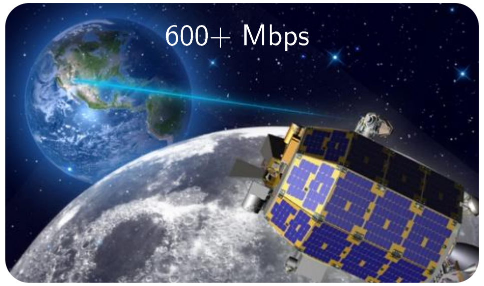
Bundle primary block parameters	
$B.src$ & $B.dst$	Source and destination nodes for the bundle
$B.size$	Bundle size, including both header and payload sizes
$B.p$ or $priority$	Priority class of the bundle (1... p)
$B.critical$	Bundle critical flag
$B.custody$	Custody transfer requested flag
$B.fragment$	Fragmentation authorized flag
$B.deadline$	Expiration time of the bundle
Computed parameters	
$B.EVC$	Estimated volume consumption ($size * 1.03$) [17]
$B.sender$	Previous sender of the bundle, informed by CLA

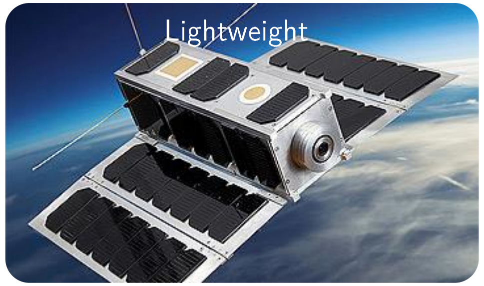
Space Networks

Bundle Protocol Implementations

- Interplanetary Overlay Network (ION) → Open Source
  - Developed by JPL, NASA – Full-featured solution (*microprocessors*)
- Micro-Planetary Communication Network (μPCN) → Open Source
  - Developed by TUD – Lightweight POSIX (*microcontrollers*)

- 2008: UK-DMC (ION)
- 2008: DINET (ION)
- 2012: JAXA GEO
- 2013: LADEE
- 2018: ISS (ION) until today
- 2019: OPS-SAT (μPCN)
- 2021: Lunar IceCube (ION)



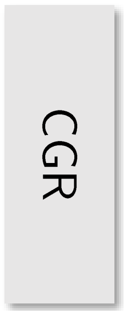


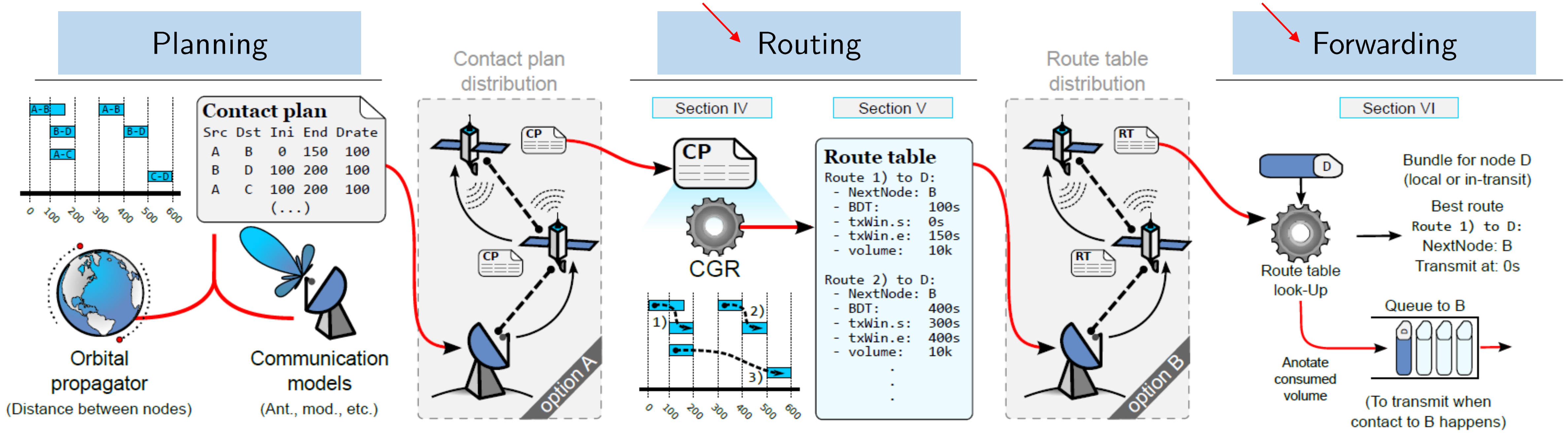
- Emulations (ION and μPCN virtualizations)
- Simulations (DtnSim)

Space Networks

Routing Framework

- Planning: pre- and post-processing of the contact plan
- Routing: path computation from contact plan
- Forwarding: path selection and queuing on outgoing interface





Space Networks

Contact Graph Routing

- Determinate *when* and to *which* neighbor a bundle should be forwarded
- Opportunistic/probabilistic DTNs: flooding, probability inference
- Scheduled DTNs: the contact plan must be efficiently processed
  - Theoretical models: (e.g., MILP) optimal (global) results at the expense of processing
  - Algorithmics: CGR has demonstrated sufficient accuracy and efficiency to become the de facto (local) routing framework for space DTNs

Space Networks

Contact Graph Routing History

- 2008: first paper on CGR (S. Burleigh, JPL)
- 2009: draft presented at IETF, early CGR versions released in ION
- 2011: source routing extension (Ed Birrane, APL), Dijkstra (J. Segui, JPL)
- 2012: consideration for LEO satellites, Ring Road networks
- 2014: arrival time estimation improvement and overbooking management
- 2015: survey in IEEE Coms Mag
- 2016: opportunistic contact extension
- 2017: forwarding improvements and regionalized scalability
- 2018: route table management, Yens' algorithm
- 2019: uncertain contact planning extensions, SABR in CCSDS

Space Networks Model

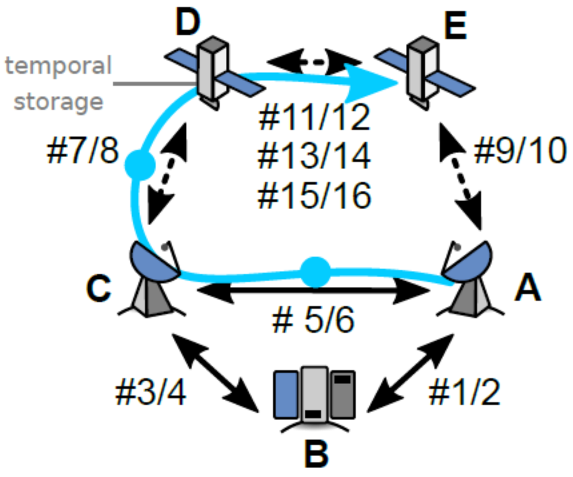
Contact

- A contact $C_{A,B}^{t1,t2}$ is defined as a time interval $(t1; t2)$ during which it is expected that data will be transmitted by node A at a rate R such that data will be received by node B.

Contact plan

#	src	dst	st	end	rate	range
1/2	A	B	0	60	1	1
3/4	B	C	0	60	1	1
5/6	A	C	0	60	1	1
7/8	C	D	0	30	1	1
9/10	A	E	10	20	1	1
11/12	D	E	0	10	1	1
13/14	D	E	30	40	1	1
15/16	D	E	50	60	1	1

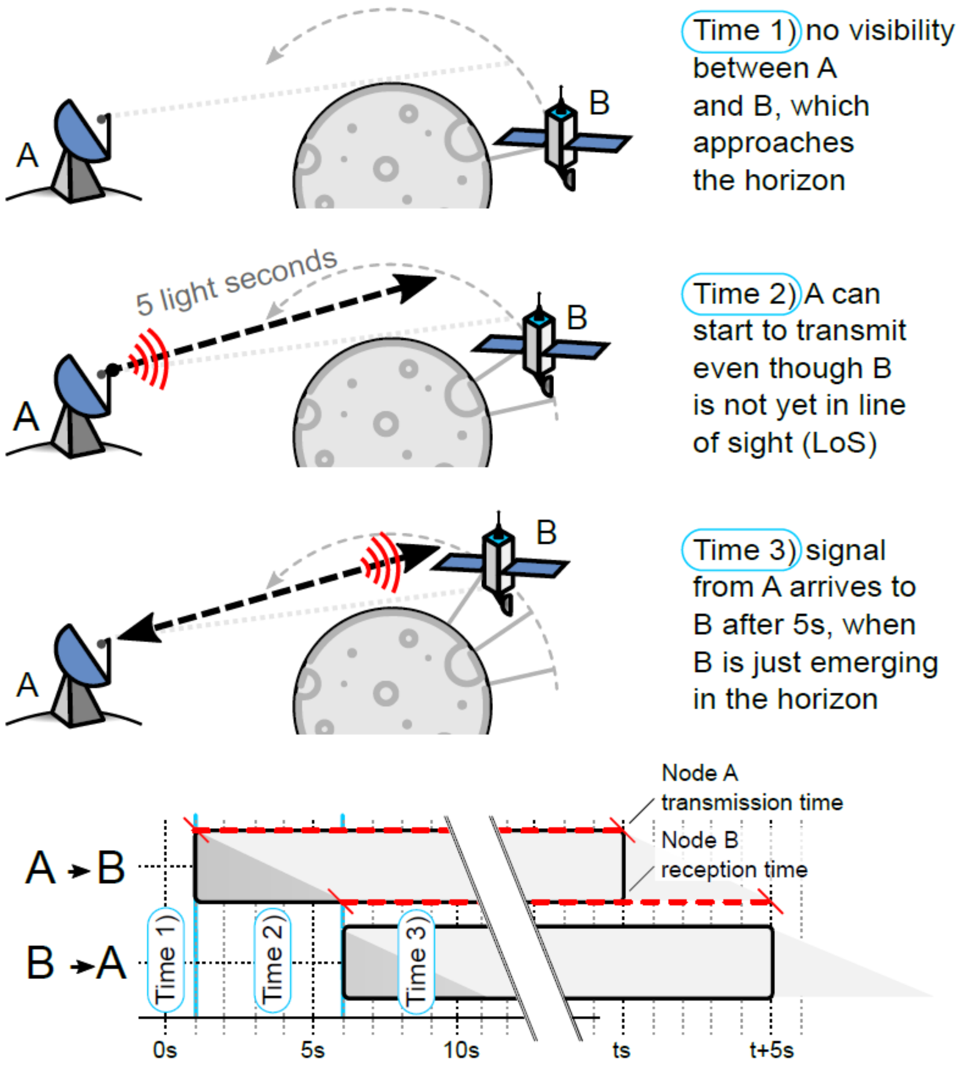
One-Way Light Time $(owlt)$



Space Networks Model

Contact Directionality

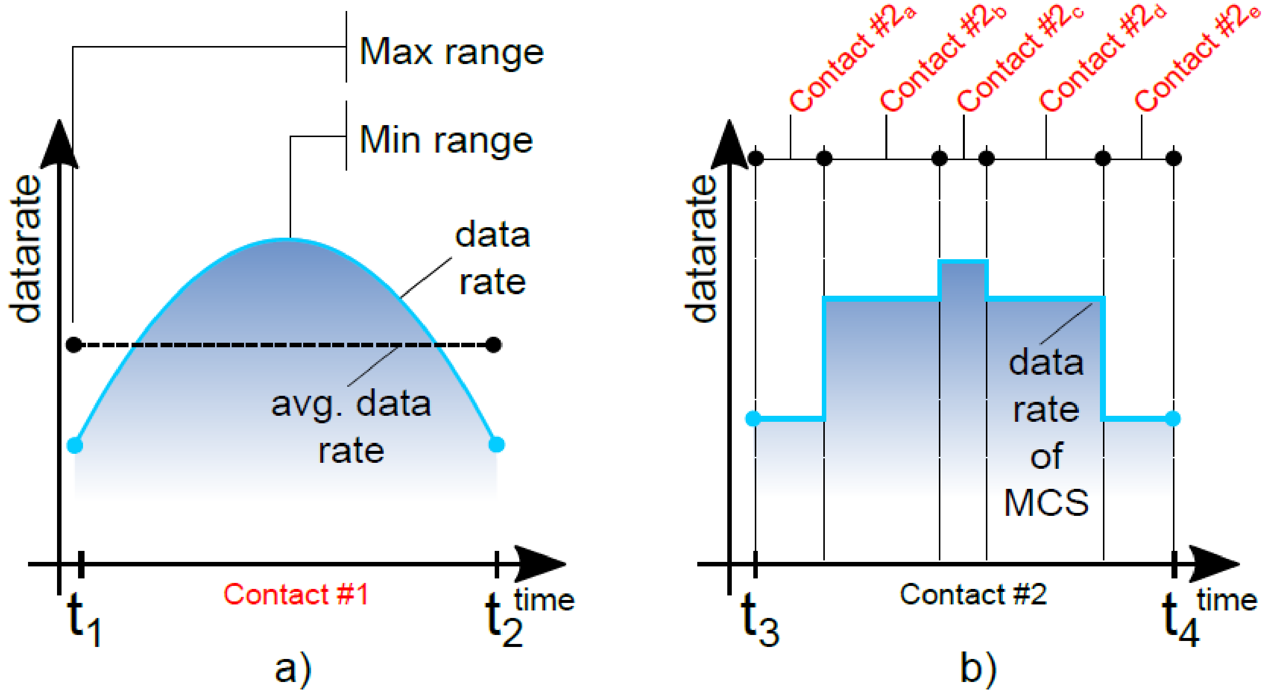
- Contacts are unidirectional
  - $\neq$ forward and return data rates
- Bidirectional $=$ pair of contacts
  - Due to *owlt*, the start time of a contact in one direction is typically **not** the same in the reverse channel



Space Networks Model

Contact Volume

- Contact volume: $(C.end - C.start) * C.rate$
  - Variable bit rates can otherwise be approximated via averaging
  - Adaptive modulation and coding can be modeled via consecutive contacts



Space Networks Model

Contact Variables

Fixed parameters	
$C.snd$, $C.rcv$	Sending and receiving nodes
$C.start$, $C.end$	Start and end transmission time
$C.rate$	Data transmission rate
$C.owlt$	Distance (range) expressed in light-seconds
$C.volume$	Contact volume $((end - start) * rate)$
Variable parameters	
$C.MAV(p)$	Maximum available volume for priority p
Route search working area:	
$C.arr\_time$	Data arrival time at the destination (dst)
$C.visited$	Contact was visited in a previous loop
$C.visited\_n[\,]$	List of previously visited nodes
$C.pred$	Pointer to the predecessor contact in route
Route management working area:	
$C.suppr$	Contact suppressed by management
$C.suppr\_nh[\,]$	List of suppressed Next hop contacts
Forwarding working area:	
$C.fbtx$	First Byte transmission time for a route
$C.lbtx$	Last Byte transmission time for a route
$C.lbrx$	Last Byte arrival time for a route
$C.EVL$	Contact's effective volume limit

Contact parameters

Computed in route search and management phase

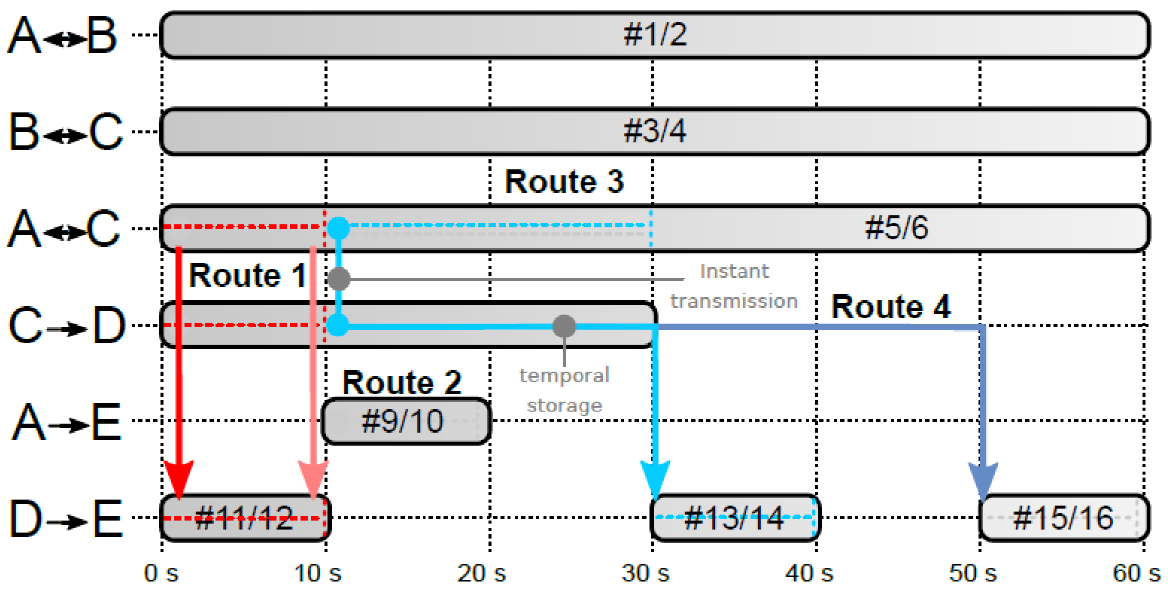
Computed in forwarding based on a specific bundle

Space Networks Model

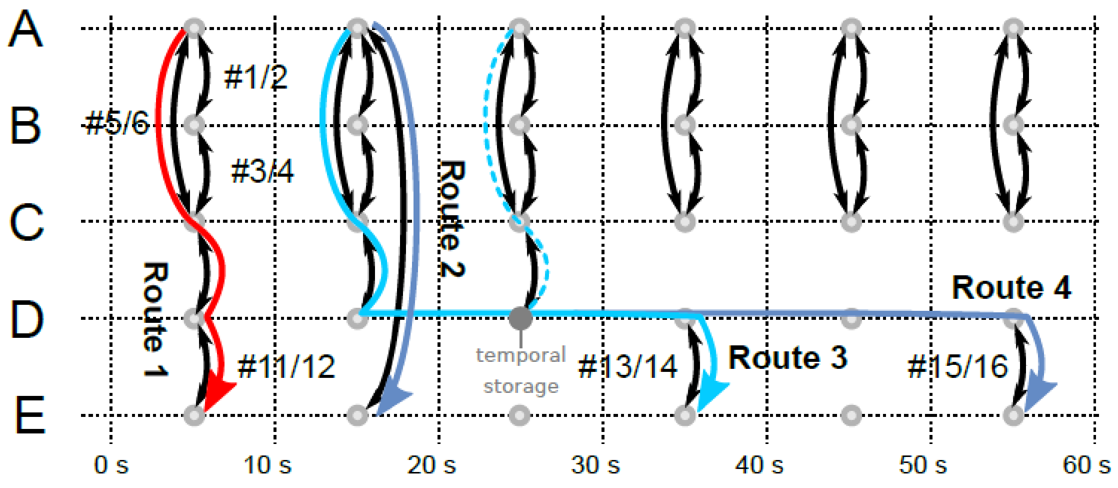
Contact Plan

- Contact plans capture the time evolving nature of the network

Timeline view



Time Evolving Graph (TEG)

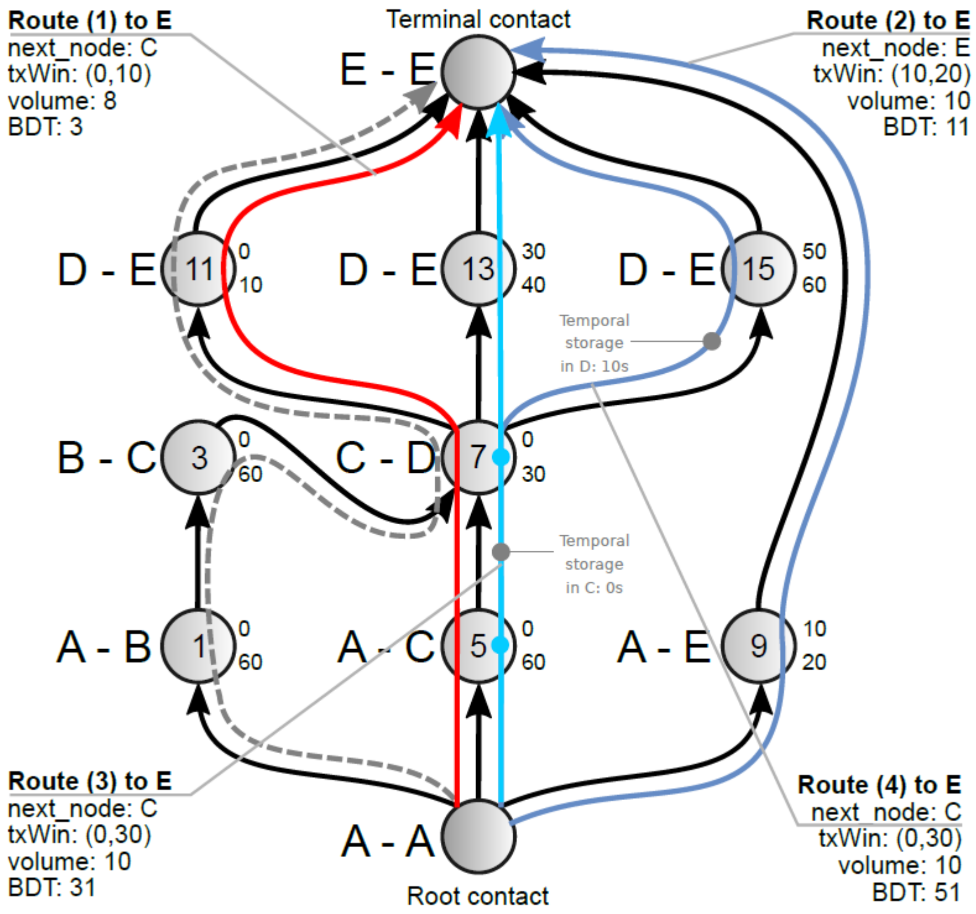


Scales poorly (nodes, contacts, time)

Space Networks Model

Contact Graph

- *Contact graph*
  - Destination node D
  - Source node S
  - Directed acyclic graph $CG_{DS} = (V; E)$
    - Edges E are episodes of data retention
    - Vertices V are episodes of contact



Space Networks Model

Contact Graph Construction

- *Vertices:* contact $\rightarrow$ transmission (directly or not) from A to E
  - Vertices model propagation delay or owlt
- *Edges* between contacts | destination = source
  - Edges represents a temporal storage in the connecting node
- Notional contacts from node A to itself and from node E to itself
  - *Root* and *terminal* contacts
- Different contact-graph is used for each source-destination pair

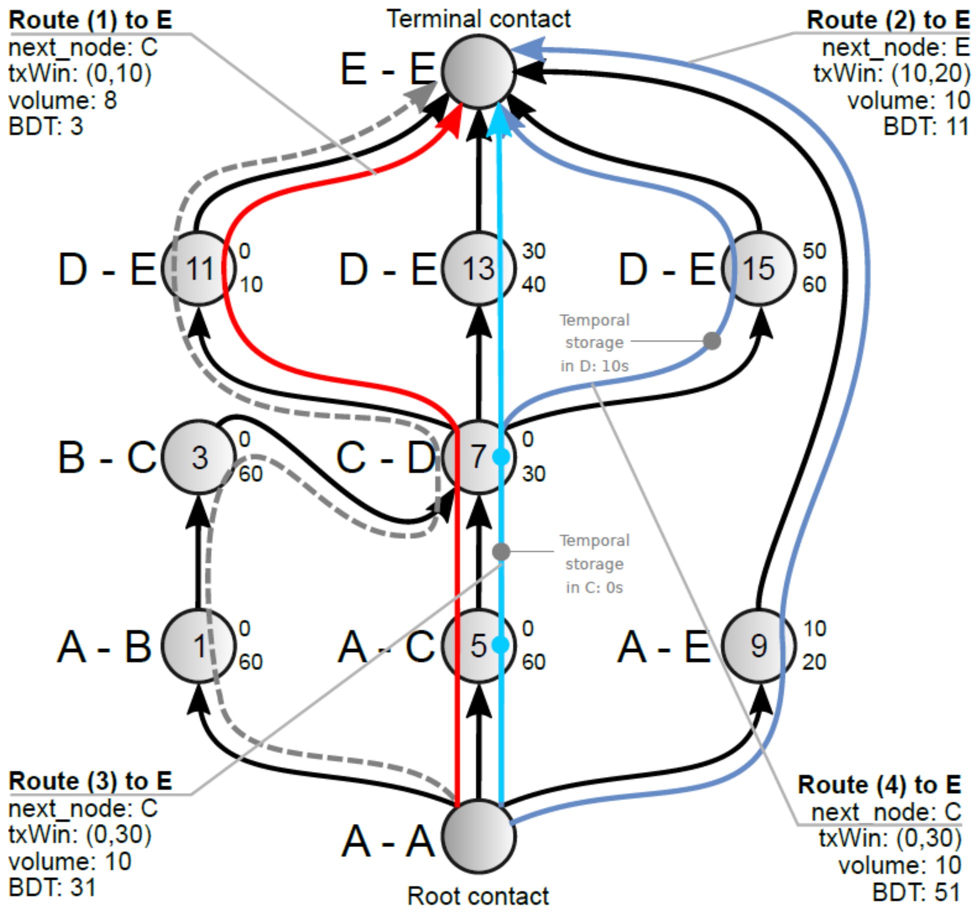
Space Networks Model

Routes

- A route R_A^E from A to E is a sequence $hops[\,]$ (contacts) such that
  1. the sending node for the first contact is A,
  2. the receiving node for the last contact is E,
  3. the receiving node for contact i is the sending node for contact $i+1$,
  4. the time at which contact $i+1$ ends is no earlier than the time at which contact i begins

Space Networks Model

Route Examples



- Route (1):
  - $R_A^E = \{C_{A,C}^{0,60}, C_{C,D}^{0,30}, C_{D,E}^{0,10}\}$
  - Best delivery time $BDT = 3$
- Route (3):
  - $R_A^E = \{C_{A,C}^{0,60}, C_{C,D}^{0,30}, C_{D,E}^{30,40}\}$
  - Storage at node D until $t = 30$
  - Best delivery time $BDT = 31$

Space Networks Model

Route Parameters

Route parameters

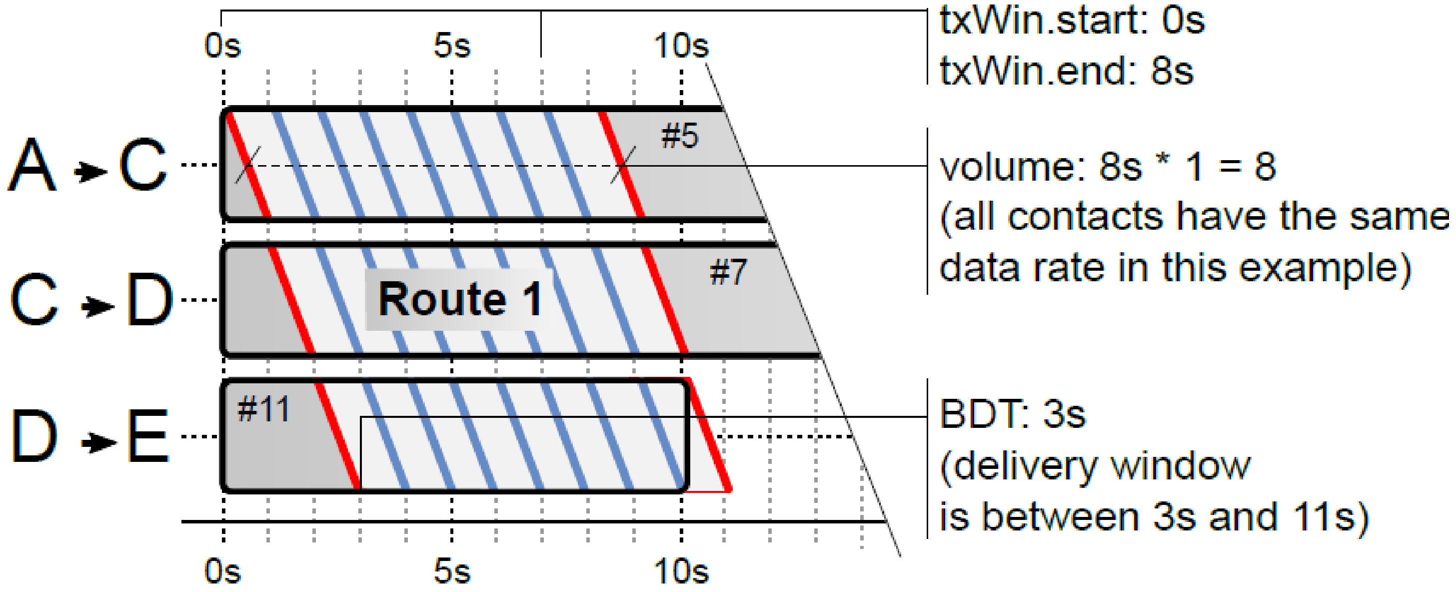
Fixed parameters	
$R.hops[\,]$	List of contacts in the route
$R.to\_node$	Final node in the path ($hops[-1].dst$)
$R.next\_node$	First neighbour in the path ($hops[1].dst$)
$R.tx\_win(s, e)$	Interval of time (s, e) where the route is valid
$R.BDT$	Best time at which data can arrive to dst
$R.volume$	Maximum data the route can carry
Variable parameters	
Forwarding working area:	
$R.ETO$	Earliest Transmission Opportunity
$R.PAT$	Projected arrival time
$R.EVL$	Route's effective volume limit

Computed in forwarding based on a specific bundle

Space Networks Model

Volume

- Volume modelling enables congestion control
- Consider route (1) $R_A^E = \{C_{A,C}^{0,60}, C_{C,D}^{0,30}, C_{D,E}^{0,10}\} \rightarrow R.volume = 8$



Route Search

Contact Graph Dijkstra Search

Routing

- Dijkstra's shortest path algorithm can be adapted for contact graphs
  - Result: a single route R_S^D from source S to destination D in contact plan CP

Route Search

Contact Graph Dijkstra Search

Algorithm 1: Contact Graph Dijkstra Search

Data: root contact C_{root}, destination D, contact plan CP (with cleared working area)

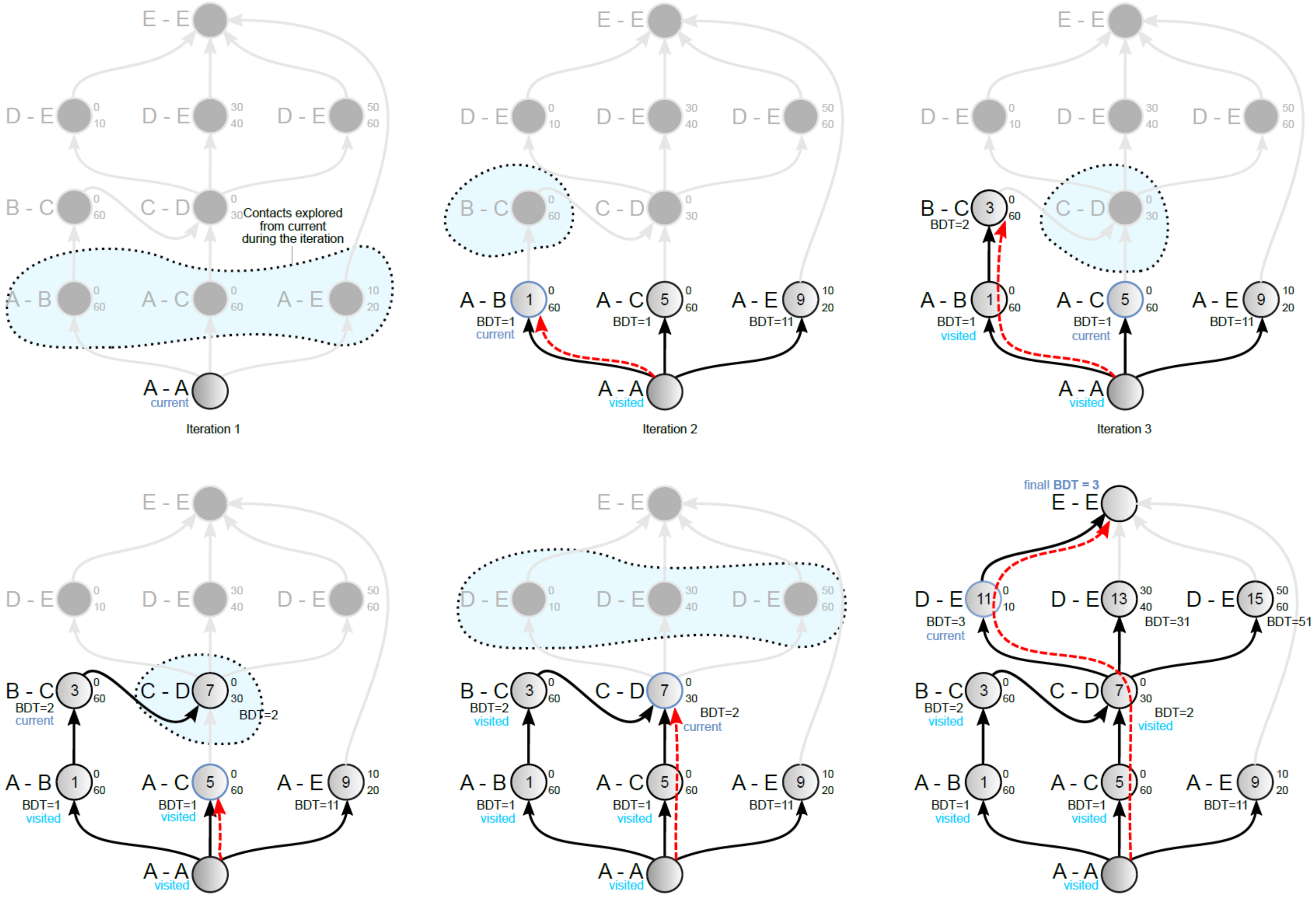
Result: Route R_S^D from source S to destination D

$R_S^D \leftarrow \{\}$
$C_{fin} \leftarrow \{\}$ // final contact
$BDT = \infty$ // final arrival time
$C_{curr} = C_{root}$ // current contact is root
/* contact plan exploration loop */
while *True* **do**
  /* contact review procedure */
  $C_{fin}, BDT \leftarrow$CRP $(CP, C_{curr}, C_{fin}, BDT)$
  /* contact selection procedure */
  $C_{next} \leftarrow$CSP (CP, C_{curr}, BDT)
  if $C_{next} \neq \{\}$ **then**
    $C_{curr} \leftarrow C_{next}$
  else
    break // review and selection completed
/* route reconstruction loop */
if $C_{fin} \neq \{\}$ **then**
  $C = C_{fin}$
  while $C \neq C_{S,S}^{0,\infty}$ **do**
    $R_S^D.hops \leftarrow \{C\}$
    $C = C.pred$ // previous contact in path
  Compute$(R_S^D.tx_win,\ R_S^D.volume)$

- CG is constructed on the fly
- Contact Review Procedure (CRP) → Alg. 2
  - Update $C.arr\_time$ ∀ successors contacts of C_{curr}
- Contact Selection Procedure (CSP) → Alg. 3
  - Selects best arr_time metric
  - $C_{curr} \leftarrow C_{next}$, ends when no is C_{next} found
- Route is reconstructed, if any
  - Based on C_{fin} and BDT

Route Search

Contact Graph Dijkstra Search



Route Management

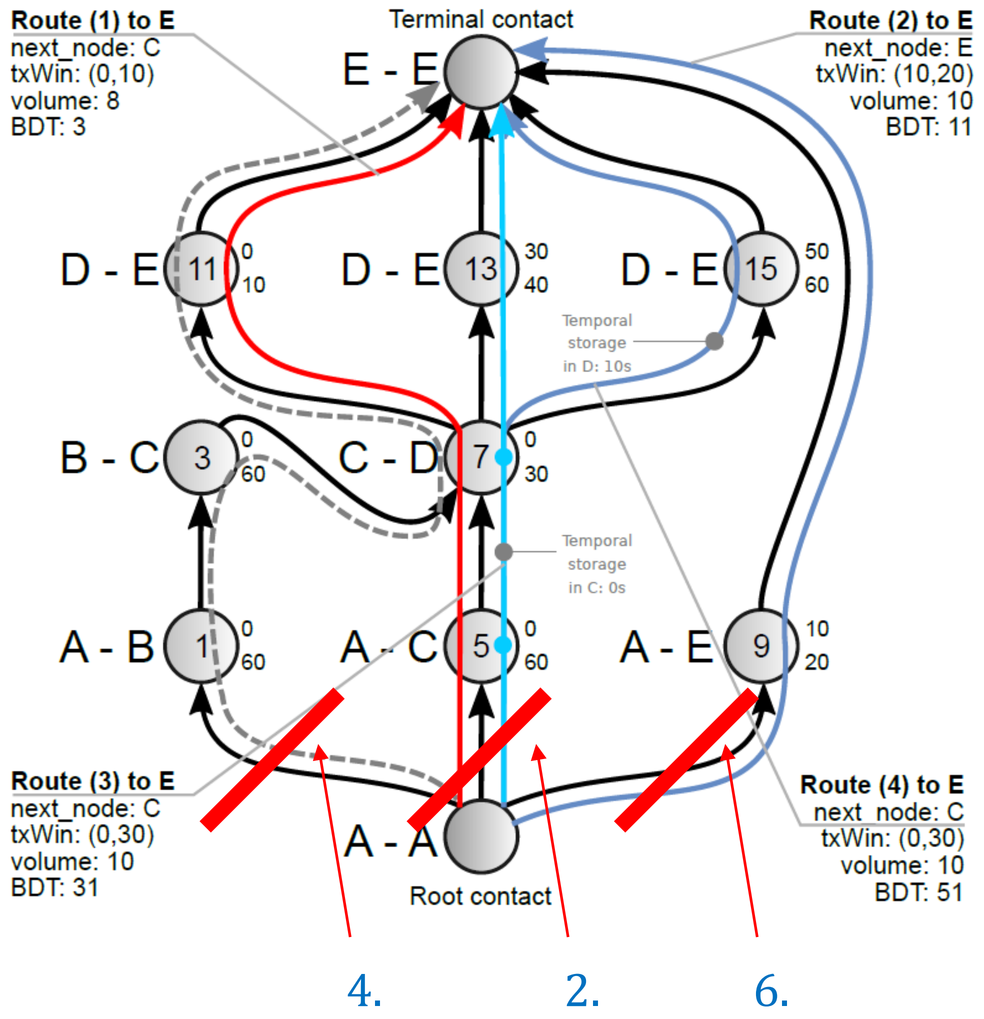
Multiple Routes

- A *list of routes* is needed for each destination
  - Route expires after $tx\_win.end$ $\rightarrow$ others will be needed afterwards
  - Limited volume $\rightarrow$ others might provide the extra capacity
  - Priority class volume $\rightarrow$ others might provide a means to reduce congestion
  - Uncertainties or failures $\rightarrow$ others might provide necessary redundancy

Route Management

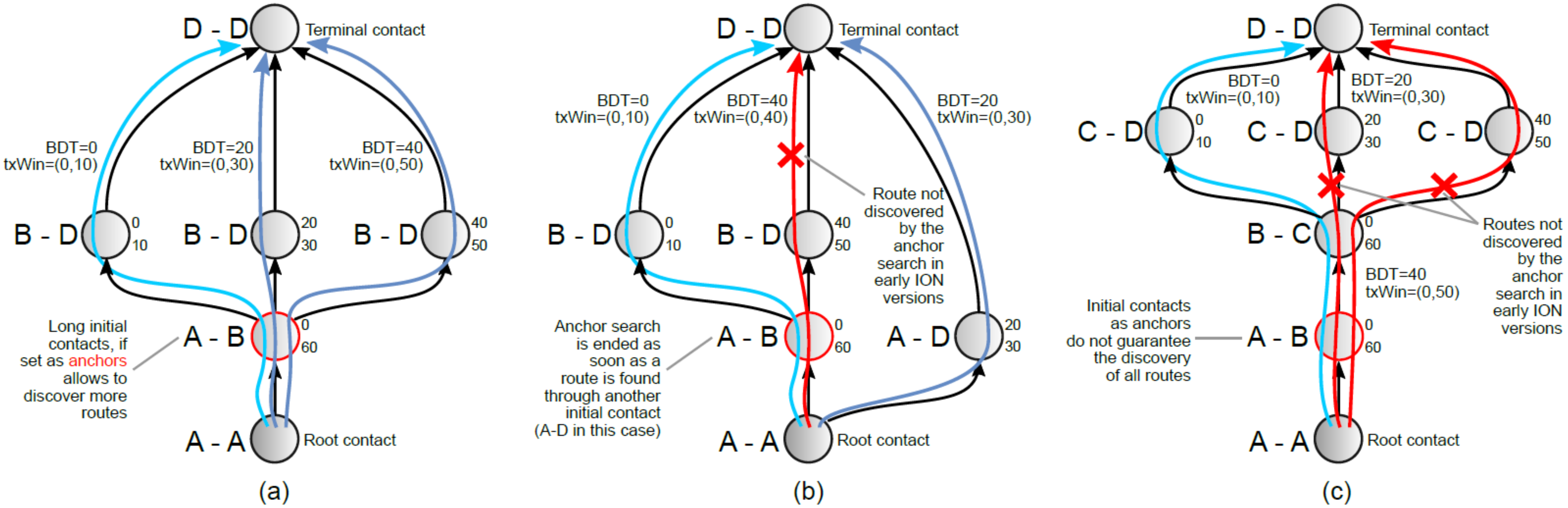
Multiple Routes: Trivial Solution

1. $R_1 \leftarrow dijkstra\_search(CP)$
2. $CP.remove\_contact(R_1.hops[0])$
3. $R_2 \leftarrow dijkstra\_search(CP)$
4. $CP.remove\_contact(R_2.hops[0])$
5. ...



Route Management

Multiple Routes: Trivial Solution Issues



Anchor on long initial contacts

Premature anchor termination

Looses routes further ahead

ION v3.6

Route Management

Multiple Routes: Yens' Algorithm

- Yen's algorithm
  - Dijkstra search in a nested loop
  - Delivers a set of the K best routes in $[R_S^D]$.
  - $[R_S^D]$ is a subset of a potential routes $[P_S^D]$
  - $[P_S^D]$ is populated with up to K routes on each iteration *originating* from the last best route found $[R_S^D][-1]$.

ION v3.7+

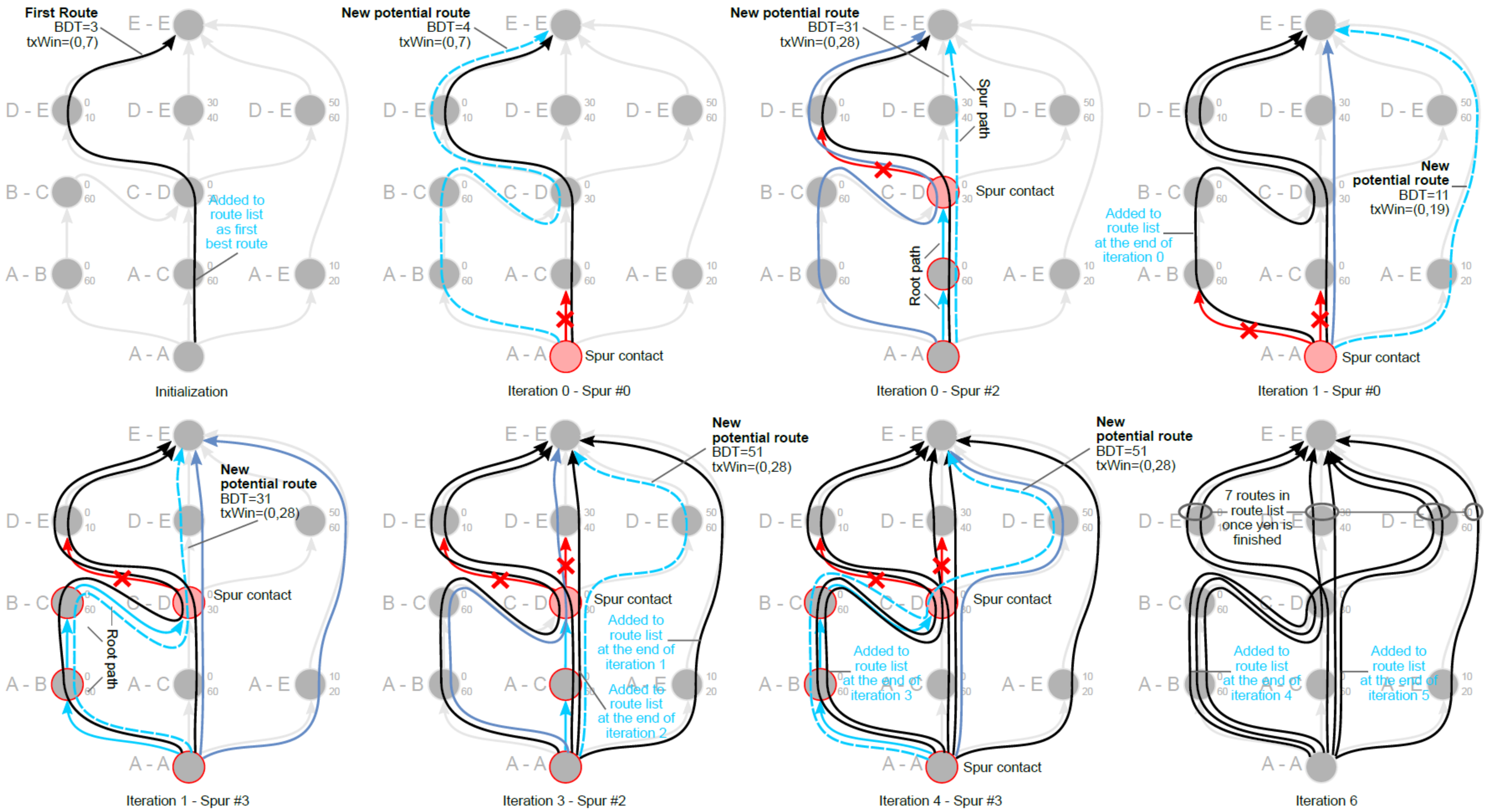
Algorithm 4: Contact Graph Yens' algorithm

Data: source-dest. S-D, contact plan CP, K routes
Result: Route list $[R_S^D]$ with K routes

Clear(CP)
$[R_S^D] \leftarrow C_{S,S}^{0,\infty} +$ Dijkstra($C_{S,S}^{t,\infty}$, D, CP)
$[P_S^D] \leftarrow \{\}$ // potential routes
for k **from** 1 **to** $K-1$ **do**
  for $C_{spur} \in [R_S^D].[-1]$ **do**
    /* root path from $C_{S,S}^{0,\infty}$ to spur contact */
    $P_{root} = [R_S^D].[-1].hops[0, C_{spur} - 1]$
    Clear(CP)
    /* suppress all contacts in root path */
    for $C \in P_{root}$ **do**
      $C.suppr = True$
    /* suppress C_{spur} edges in any R in R_S^D */
    for $R \in [R_S^D]$ **do**
      if $P_{root} = R.hops$ **then**
        $C_{spur}.suppr\_nh \leftarrow R.hops[len(P_{root})]$
    /* compute spur path from C_{spur} to D */
    $C_{spur}.arr\_time = P_{root}.arr\_time$
    $C_{spur}.visited\_n \leftarrow \forall P_{root}.hops.to$
    $P_{spur} =$ Dijkstra(C_{spur}, D, CP)
    /* if any, insert new potential route */
    if $P_{spur} \neq \{\}$ **then**
      $[P_S^D] \leftarrow \{P_{root} + P_{spur}\}$
      Sort $[P_S^D]$ by arrival time
  /* move best potential route to $[R_S^D]$ */
  if $[P_S^D]$ *is not empty* **then**
    $[R_S^D] \leftarrow P_S^D[0]$
  else
    finish // no more potential routes

Route Management

Multiple Routes: Yens' Algorithm



Forwarding

Candidate Routes

Forwarding

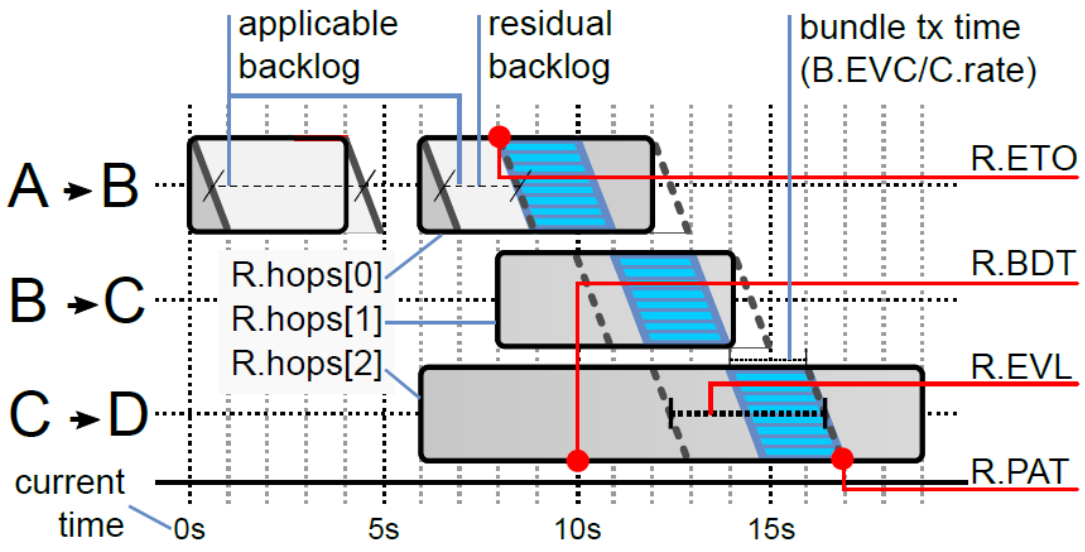
- Candidate route: subset of $[R_S^D]$ computed at forwarding time
  - Based on queue status at current time and the parameters of the bundle

Forwarding

Candidate Routes Computation

- Deadline checks
  - $R.BDT > B.deadline$
- Earliest *tx* opportunity (ETO)
  - Queue backlog ($B.EVC$) for B.p
- Projected arrival time (PAT)
  - B transmission time ∀ $R.hops$
- Effective volume limit (EVL)
  - Lowest value ∀ $C.EVL \in R.hops$

Alg. 5



Forwarding

Candidate Route Selection

- Best candidate route criteria
  1. Smallest value of projected arrival time (PAT)
  2. Fewest contacts ($hops$)
  3. Latest termination time (end of tx_win)
  4. Smallest first hop node ID number
- Critical bundles ($B.critical$),
  - A copy sent to every neighbor ($R.to\_node$) in the candidate route list

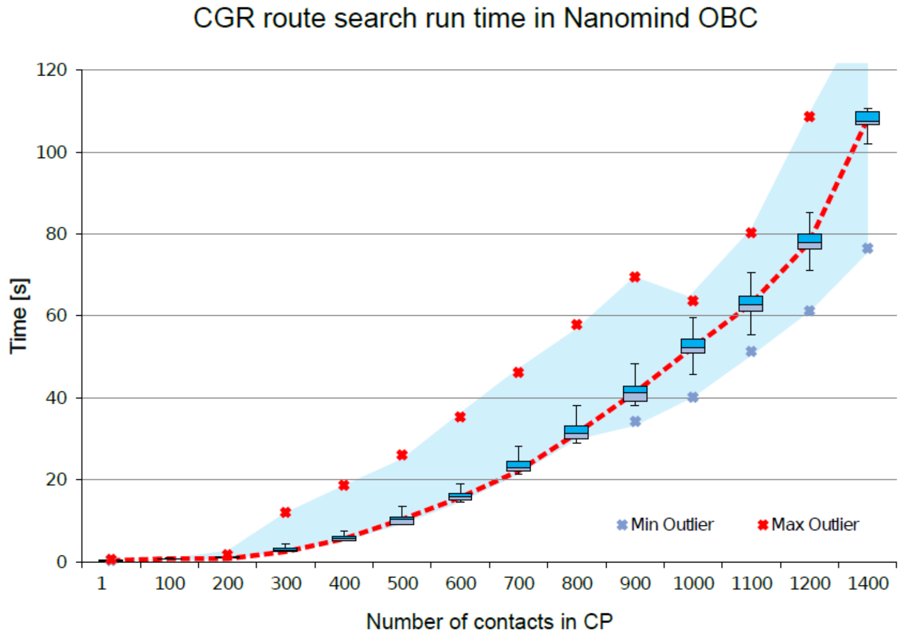
Forwarding

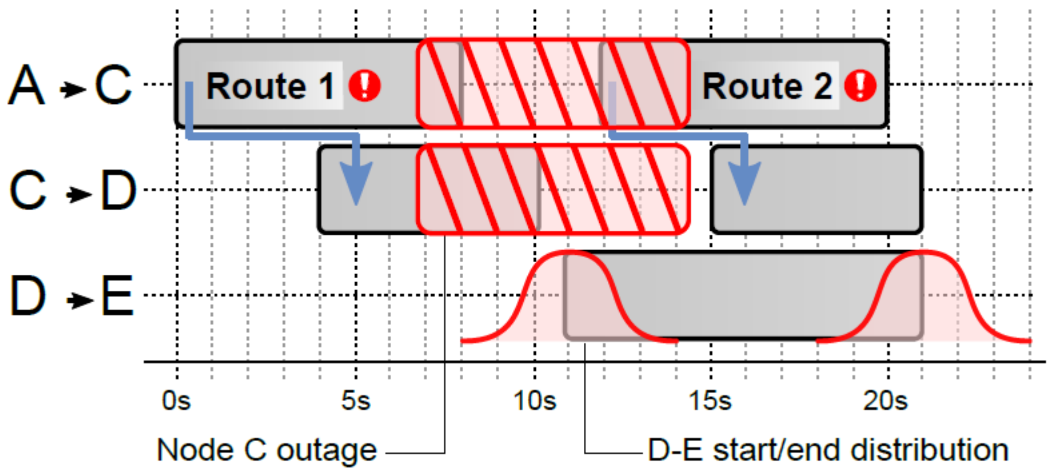
Post-Forwarding Actions

- After forwarding
  - Update $C.MAV(p)\ \forall\ C \in R.hops[]\ \forall\ p < B.p$
  - If $R.RVL(p) = R.volume \mid p < B.p$ (subscribed) $\rightarrow$ overbooking management
  - If the first contact did not occur as expected, B shall be re-forwarded

Trends Outlook

Scalability and Uncertainty





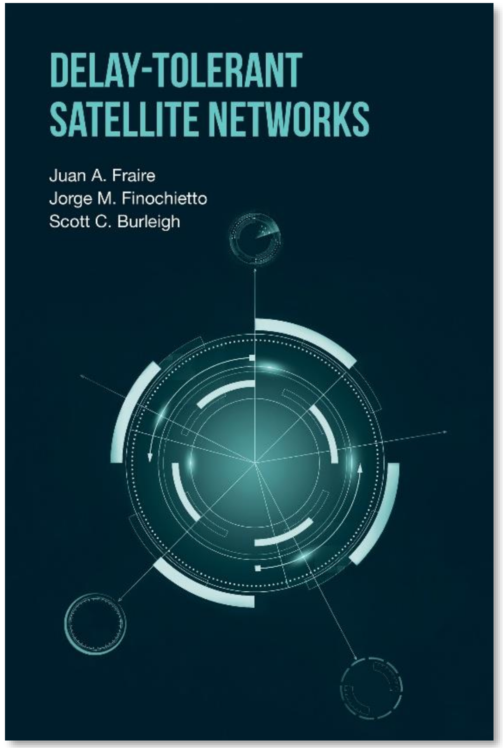
Scalability
(Hierarchical inter-regional routing)

Uncertainty
(Uncertain and Opportunistic routing)

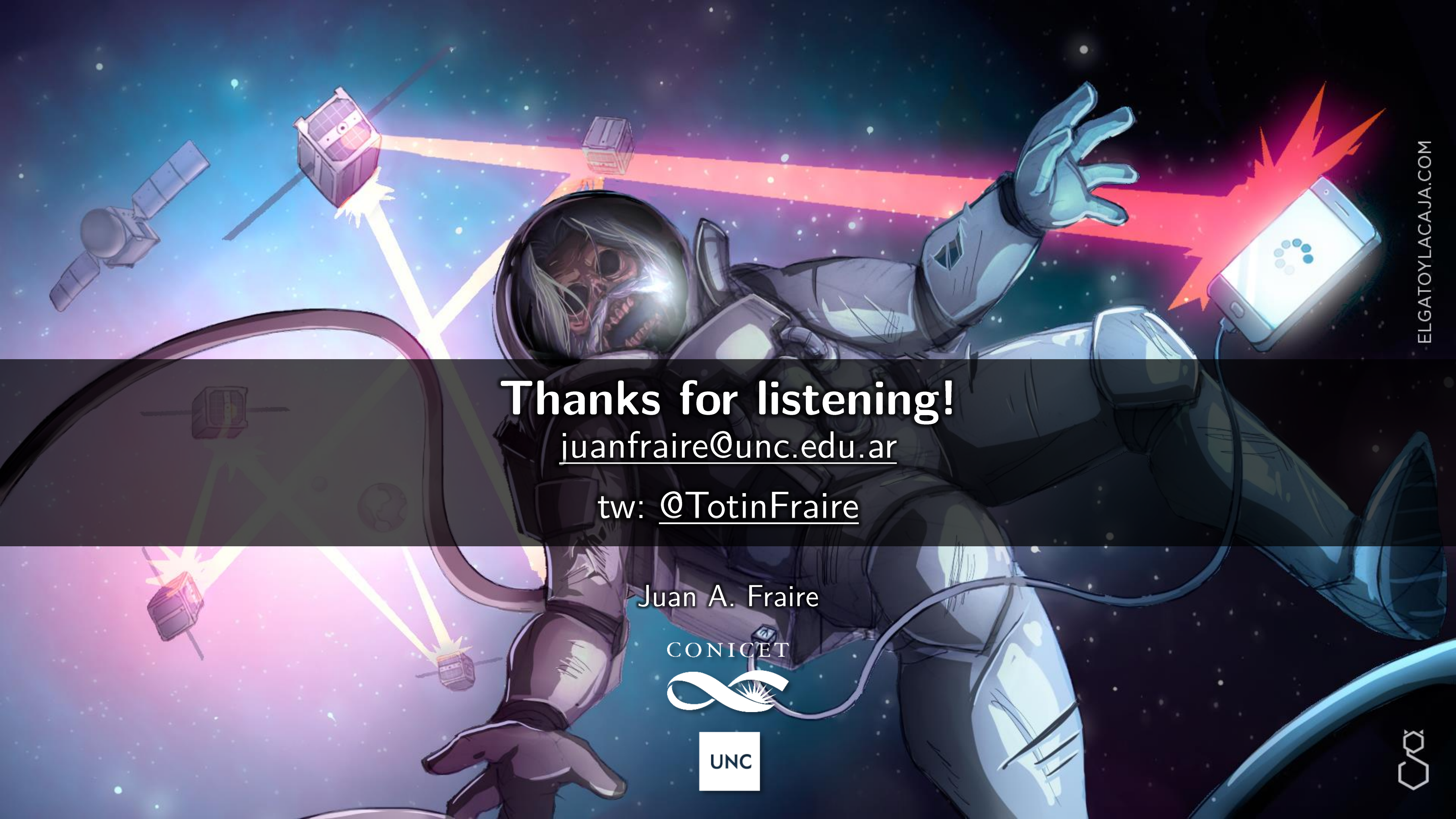
References

Materials and Further Reading

- Software
  - PyCGR library: https://bitbucket.org/juanfraire/pycgr.git
  - DtnSim: https://bitbucket.org/lcd-unc-ar/dtnsim/src
- Outreach
  - https://elgatoylacaja.com/dame-una-senal/
  - https://elgatoylacaja.com/rastis-satelitales/
- Papers
  - https://scnl.diten.unige.it/publications/CommMag-2015-CG.pdf
  - https://www.hindawi.com/journals/jcnc/2017/2830542/abs/
- Algorithms 2, 3 & 5
  - https://drive.google.com/file/d/12zpEwKnipiq_qqIh0kwchBbih88l8dpV/view?usp=sharing



https://www.amazon.com/-/es/Juan-Fraire-ebook/dp/B079C3W62L



Thanks for listening!

juanfraire@unc.edu.ar

tw: @TotinFraire

Juan A. Fraire

CONICET

UNC