Optimal Routing in Satellite DTN through Markov Decision Processes

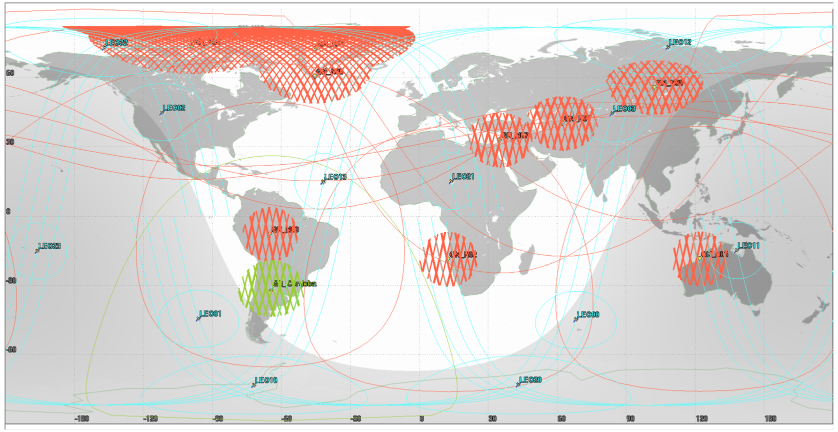
Pedro R. D'Argenio

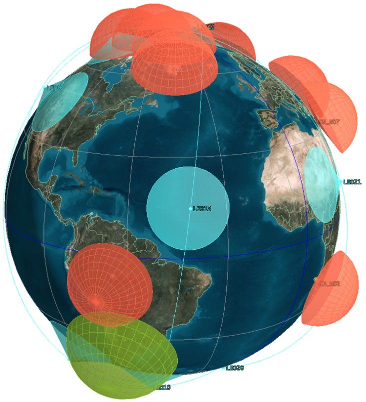
Joint work with
Juan Fraire, Arnd Hartmanns, Fernando Raverta,
Ramiro Demasi, Pablo Madhoery, Jorge Finochieto

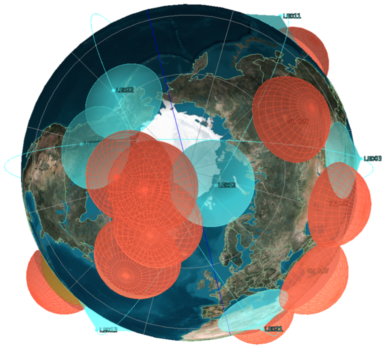
MISSION@INVAP - February 2022

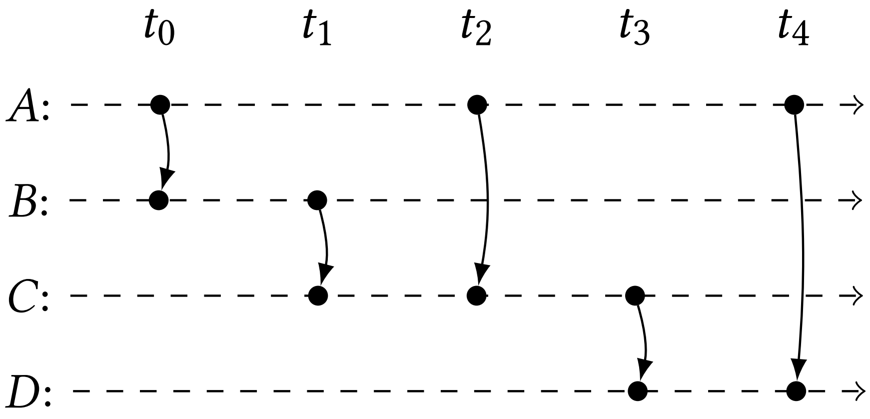
Satellite Delay Tolerant Networks

Contact Plan









Standard: Contact Graph Routing (CGR)

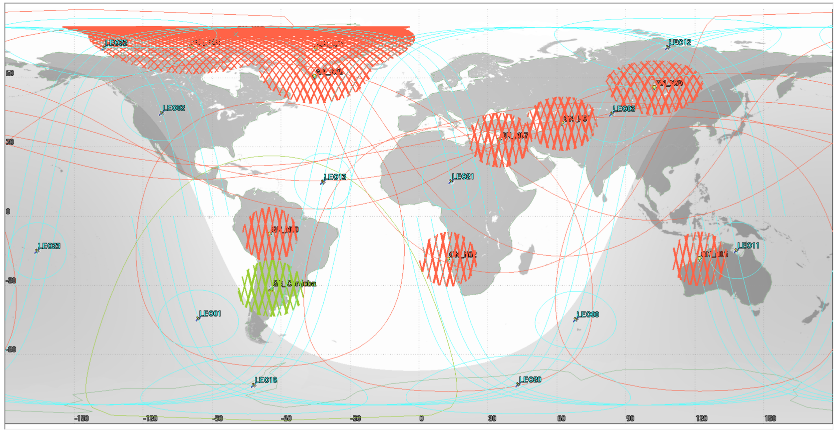
CONICET

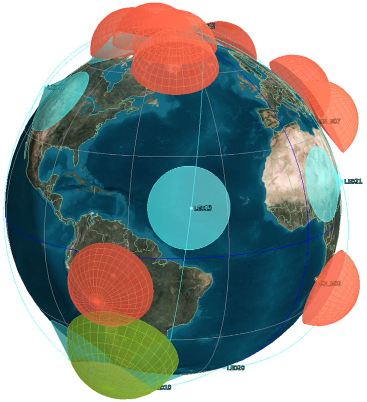
UNC

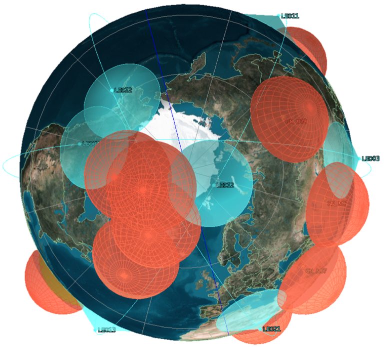
UNIVERSITÄT DES SAARLANDES

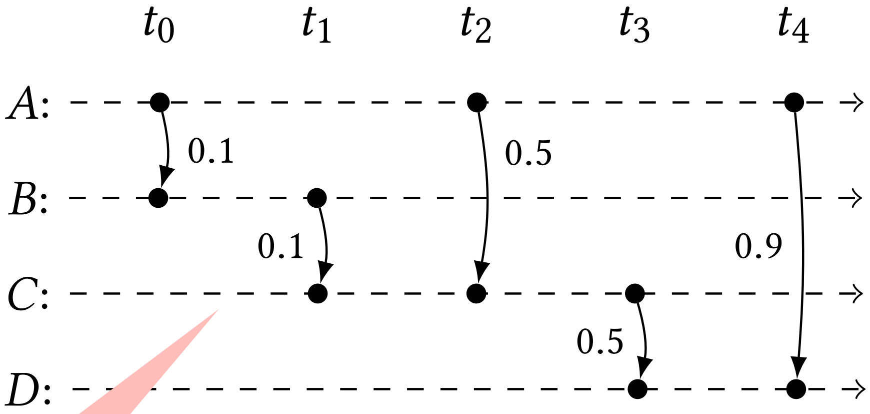
Satellite Delay Tolerant Networks

Contact Plan









Links may fail!

Standard: Contact Graph Routing (CGR)

Increase reliability: CGR with multiple copies

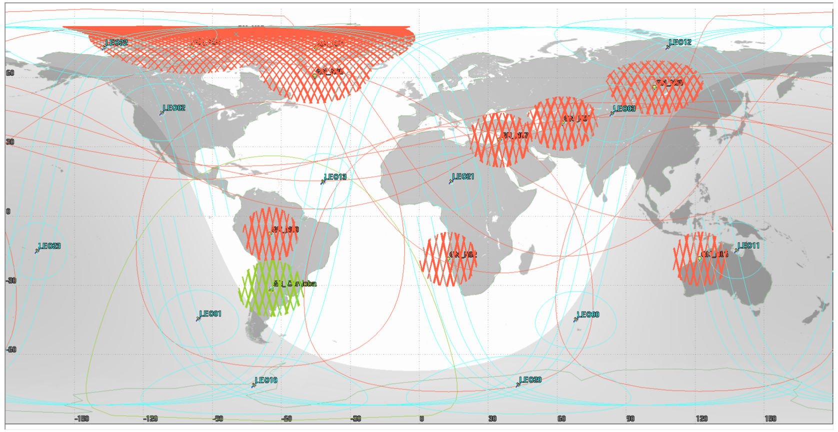
CONICET

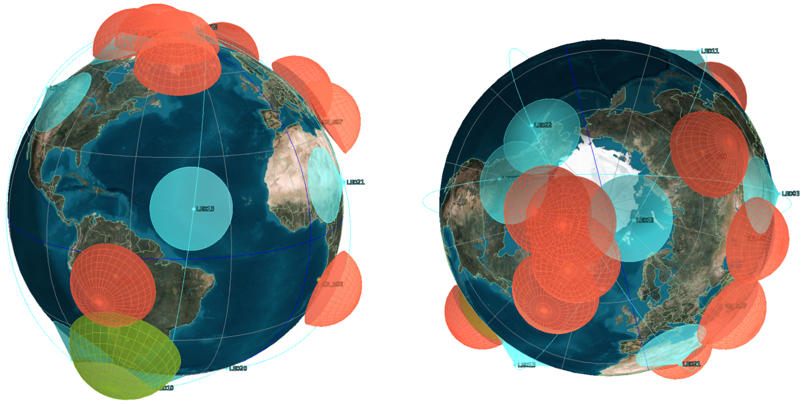
UNC

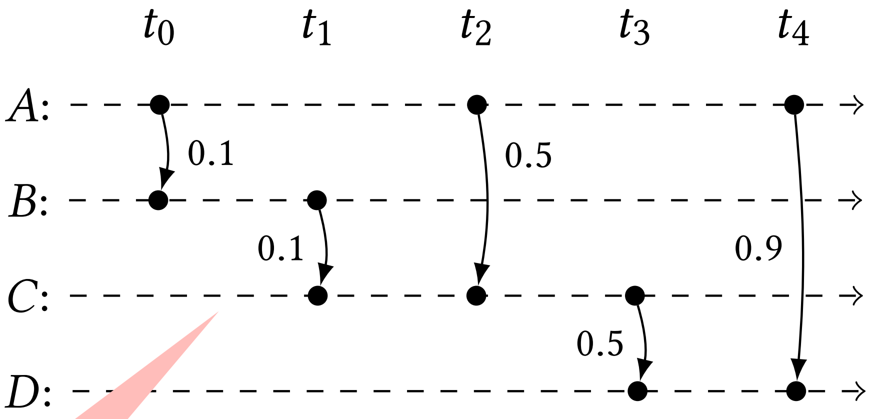
UNIVERSITÄT DES SAARLANDES

Satellite Delay Tolerant Networks

Contact Plan







Links may fail!

Not optimal!

Standard: Contact Graph Routing (CGR)

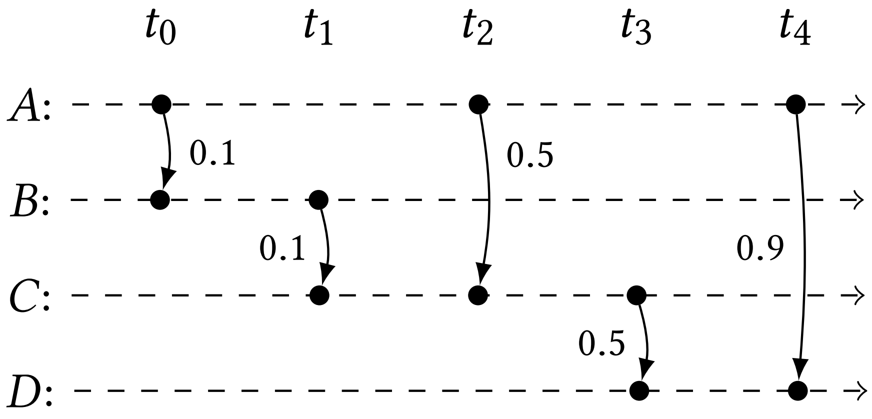
Increase reliability: CGR with multiple copies

CONICET

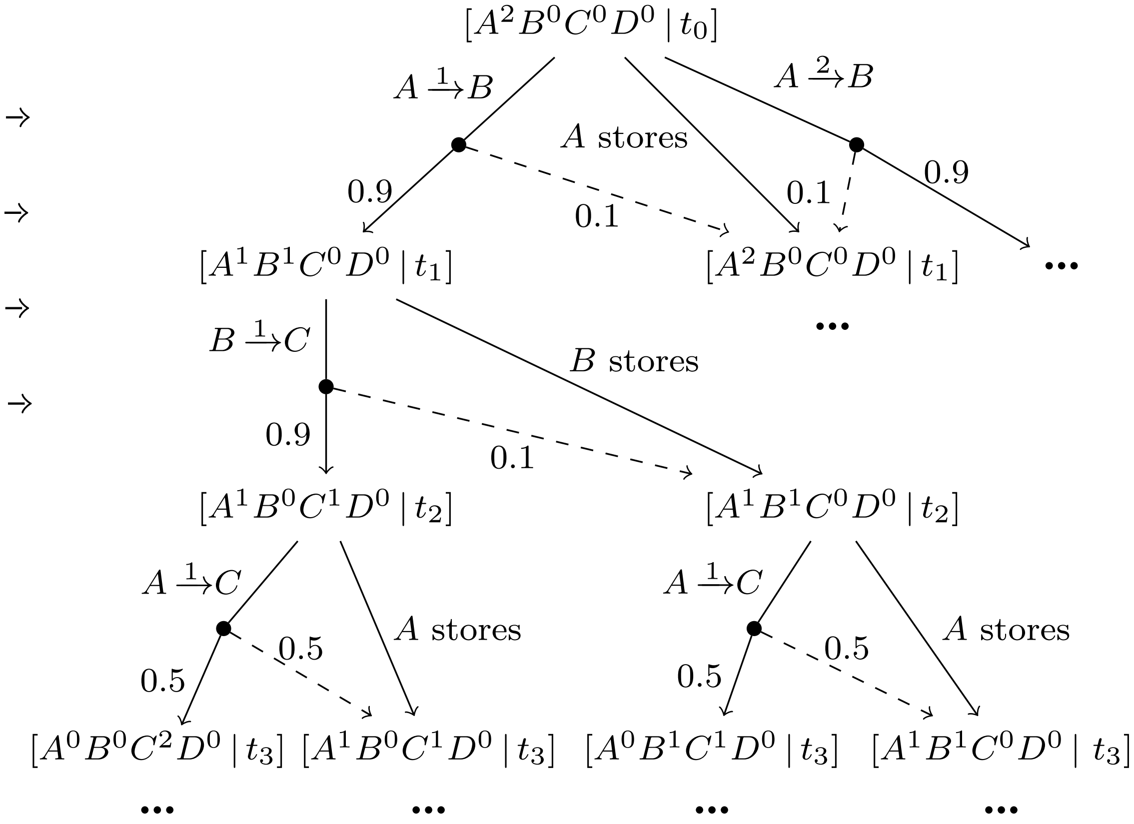
UNC

UNIVERSITÄT DES SAARLANDES

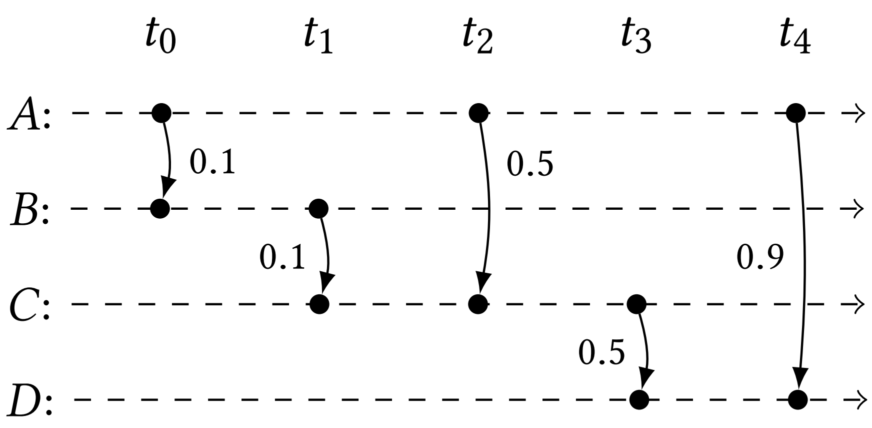
Optimality through Markov Decision Processes



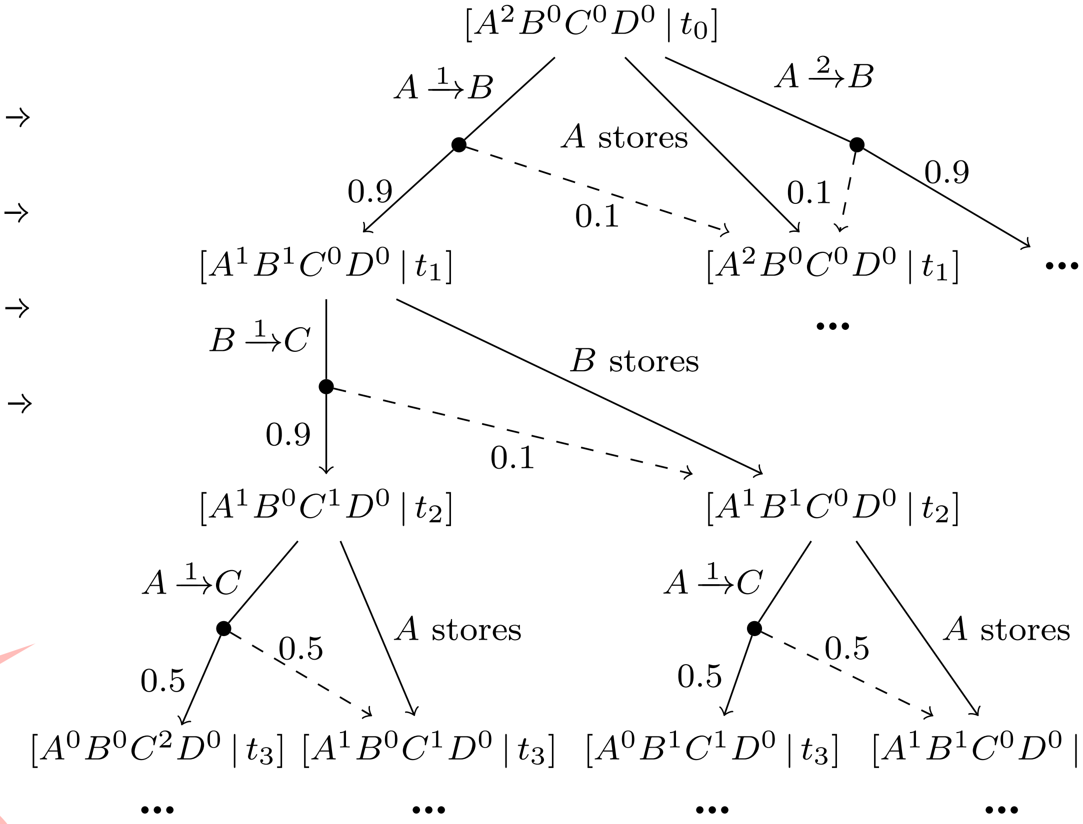
Assume 2 copies are sent



Optimality through Markov Decision Processes

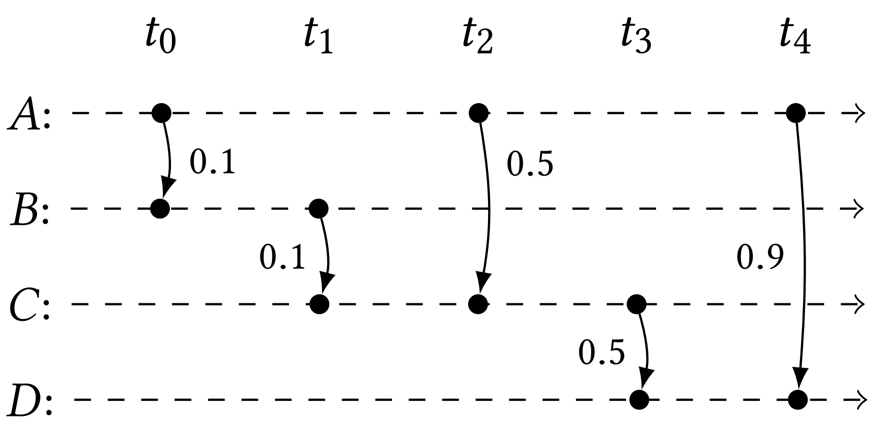


Assume 2 copies are sent

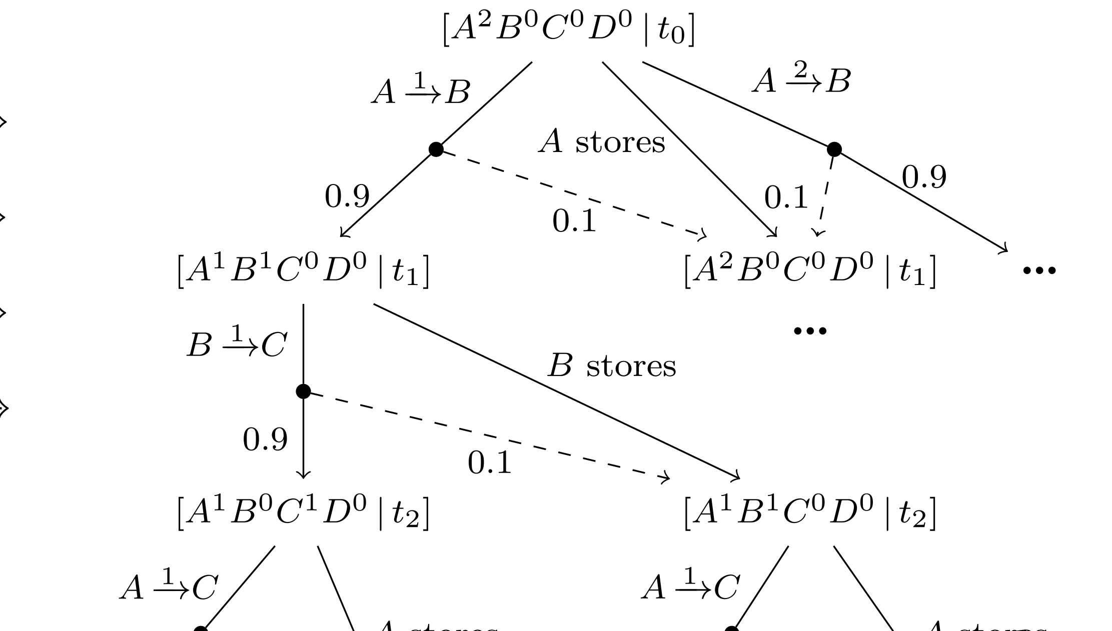


We have a reachability problem where goal states are those with a copy at target node

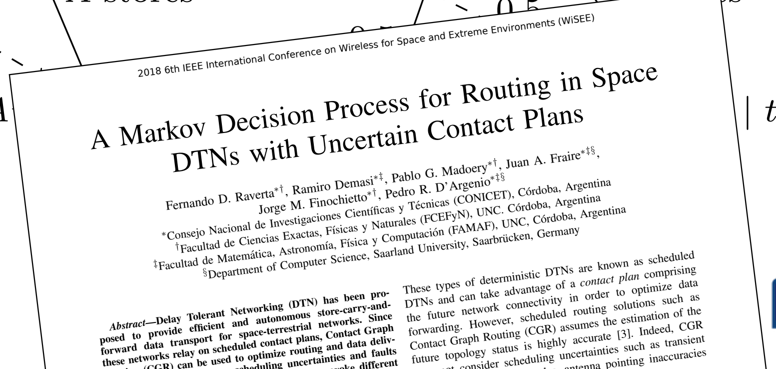
Optimality through Markov Decision Processes



Assume 2 copies are sent



We have a reachability problem where goal states are those with a copy at target node


2018 6th IEEE International Conference on Wireless for Space and Extreme Environments (WiSEE)

A Markov Decision Process for Routing in Space DTNs with Uncertain Contact Plans

Fernando D. Raverta*†, Ramiro Demasi*‡, Pablo G. Madoery*†, Juan A. Fraire*‡§, Jorge M. Finochietto*†, Pedro R. D'Argenio*‡§
*Consejo Nacional de Investigaciones Científicas y Técnicas (CONICET), Córdoba, Argentina
†Facultad de Ciencias Exactas, Físicas y Naturales (FCEFyN), UNC. Córdoba, Argentina
‡Facultad de Matemática, Astronomía, Física y Computación (FAMAF), UNC, Córdoba, Argentina
§Department of Computer Science, Saarland University, Saarbrücken, Germany

***Abstract*—Delay Tolerant Networking (DTN) has been proposed to provide efficient and autonomous store-carry-and-forward data transport for space-terrestrial networks. Since these networks relay on scheduled contact plans, Contact Graph ... (CGR) can be used to optimize routing and data deliv- ... scheduling uncertainties and faults ... different**

These types of deterministic DTNs are known as scheduled DTNs and can take advantage of a *contact plan* comprising the future network connectivity in order to optimize data forwarding. However, scheduled routing solutions such as Contact Graph Routing (CGR) assumes the estimation of the future topology status is highly accurate [3]. Indeed, CGR ... consider scheduling uncertainties such as transient ... antenna pointing inaccuracies

UNIVERSITÄT DES SAARLANDES

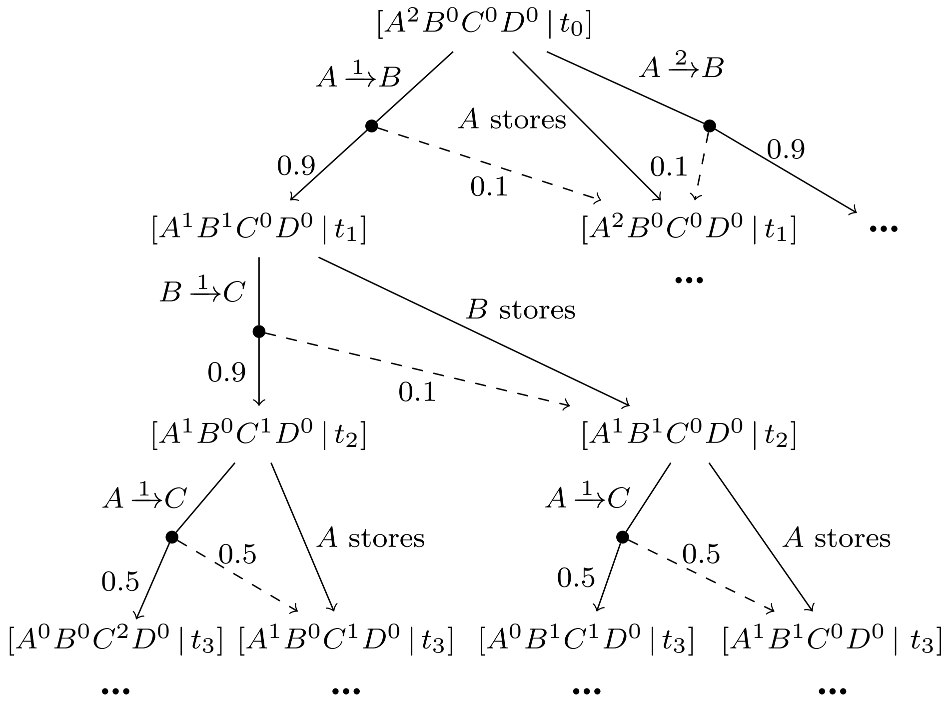
First technique

Routing under Uncertain Contact Plans (RUCoP)

Observe: MDP (almost) acyclic

RUCoP:

- follows Bellman equations backwardly (starting from goal states)
- only one pass required
- only maximizing subgraph (Markov chain!) is preserved



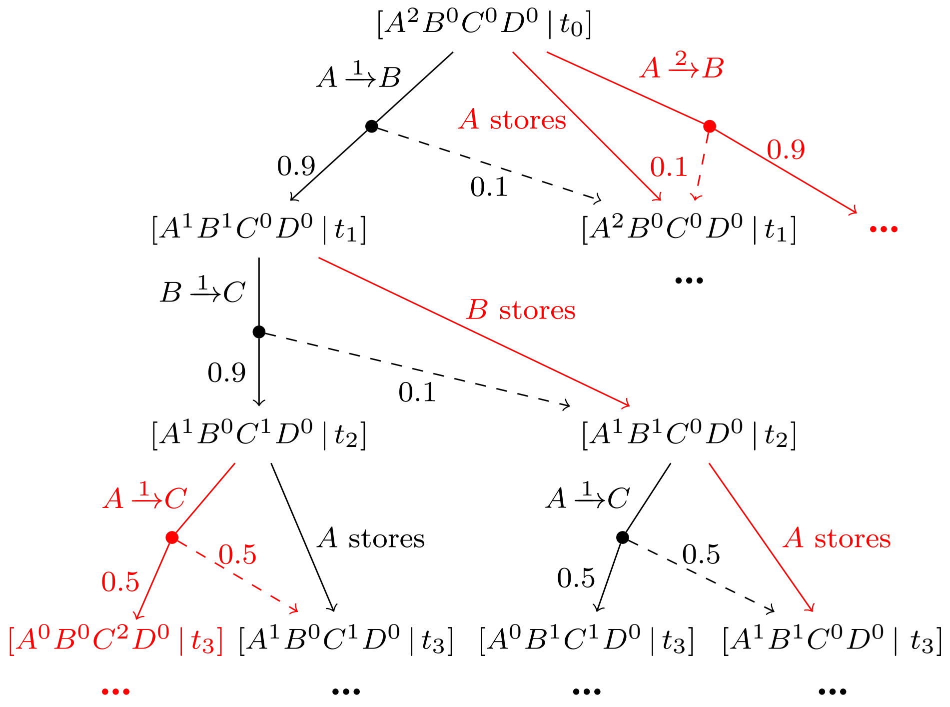
First technique

Routing under Uncertain Contact Plans (RUCoP)

Observe: MDP (almost) acyclic

RUCoP:

- follows Bellman equations backwardly (starting from goal states)
- only one pass required
- only maximizing subgraph (Markov chain!) is preserved



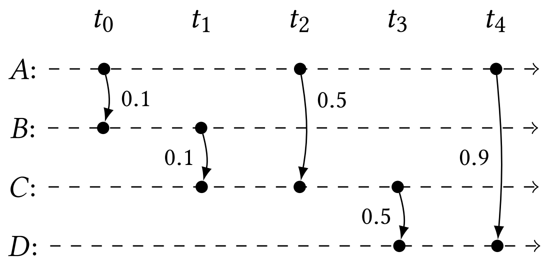
Second technique

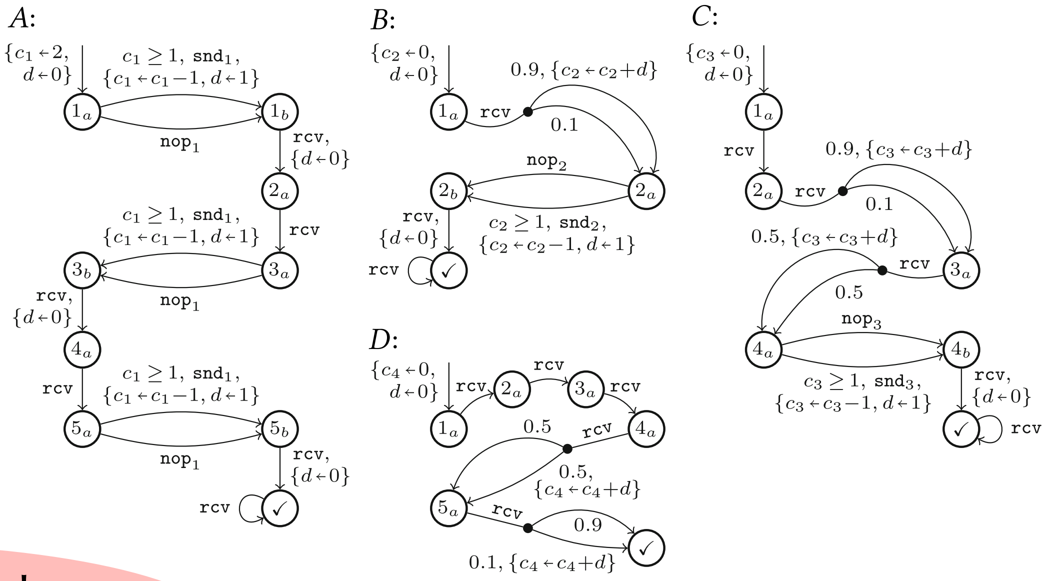
Simulation through Lightweight Smart Sampling (LSS)

SMC+LSS:

1. Select m 32-bit integer, each of them representing a scheduler identifier σ
2. For each σ, perform standard SMC letting σ resolve all non-determinism
3. Return the maximum (or minimum) and the corresponding σ

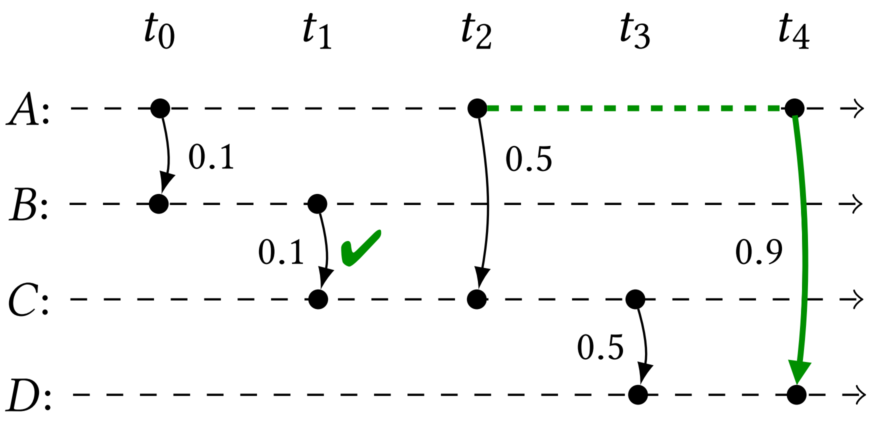
- SMC+LSS returns an underapproximation (or overapproximation) which we call near optimal
- The efficiency depends on m

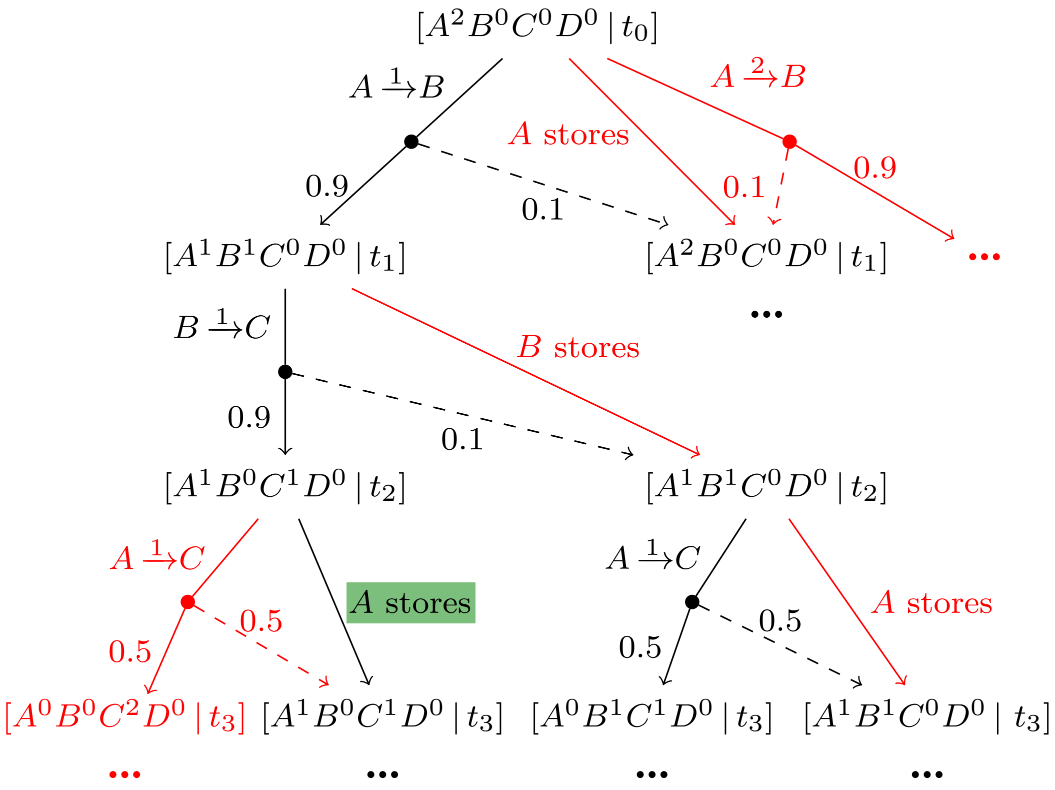




Implemented in the MODEST toolset

The problem of distributed information



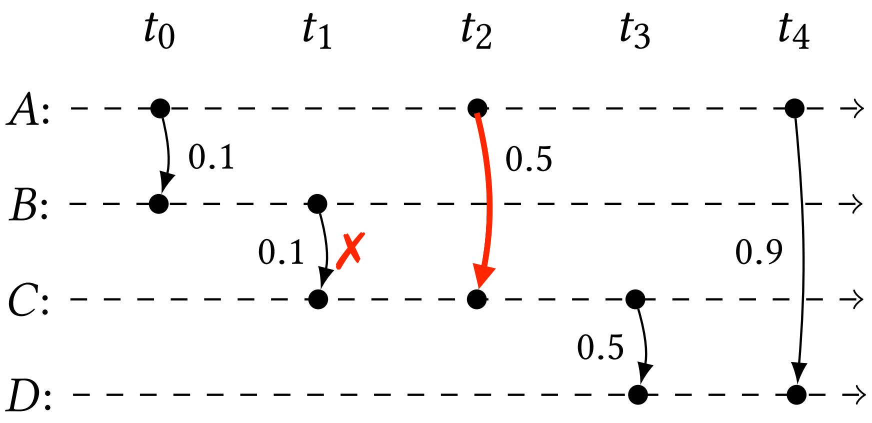


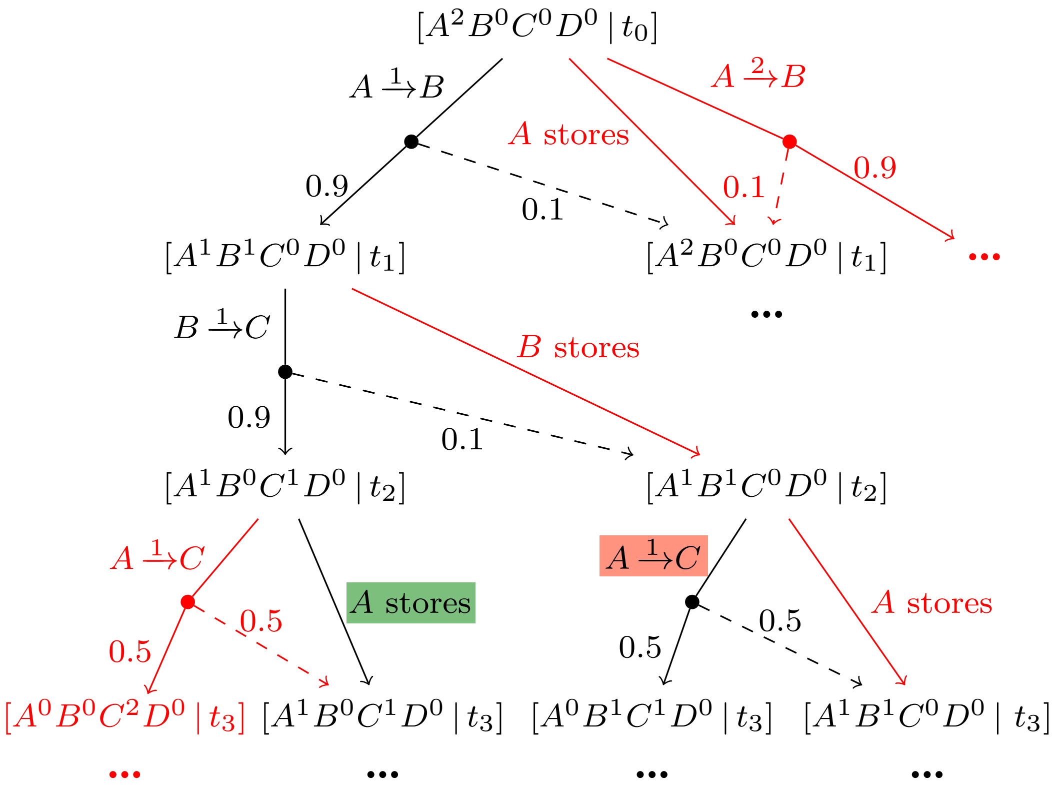
CONICET

UNC

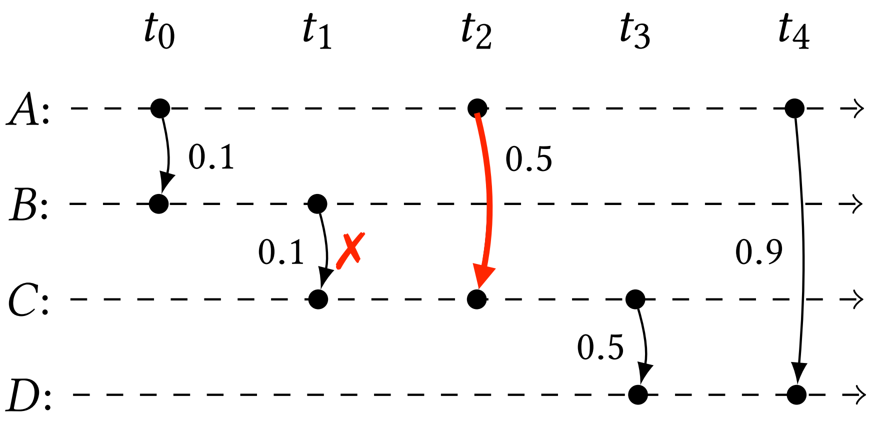
UNIVERSITÄT DES SAARLANDES

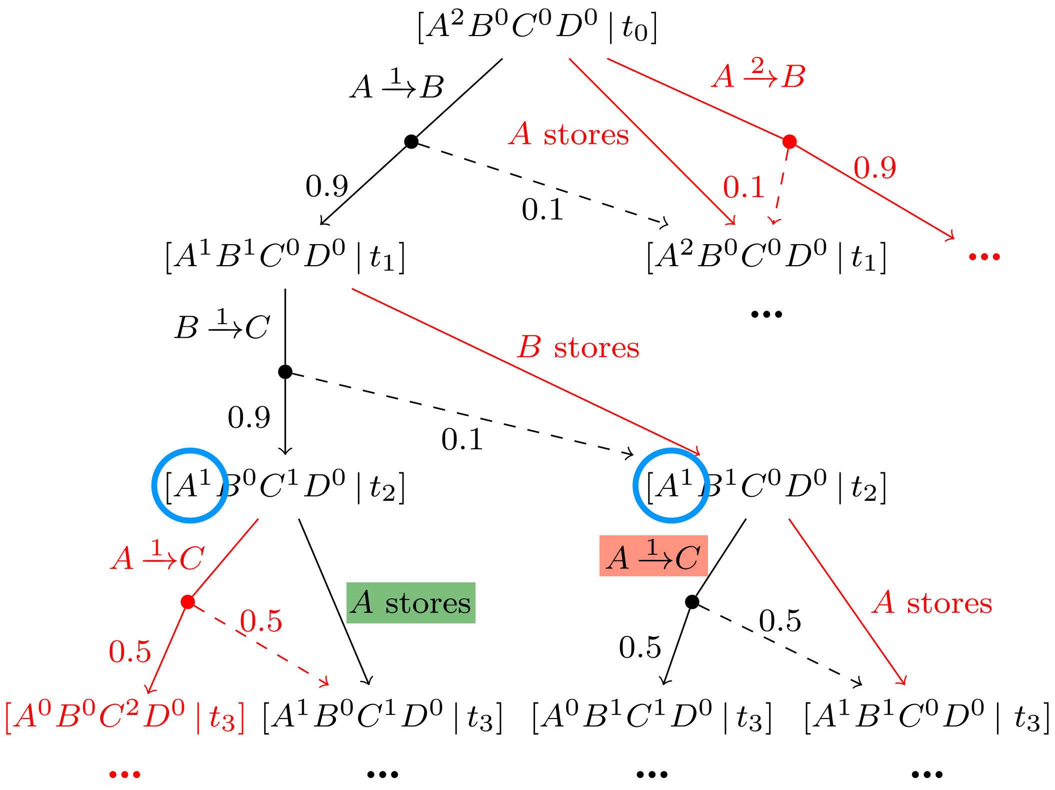
The problem of distributed information





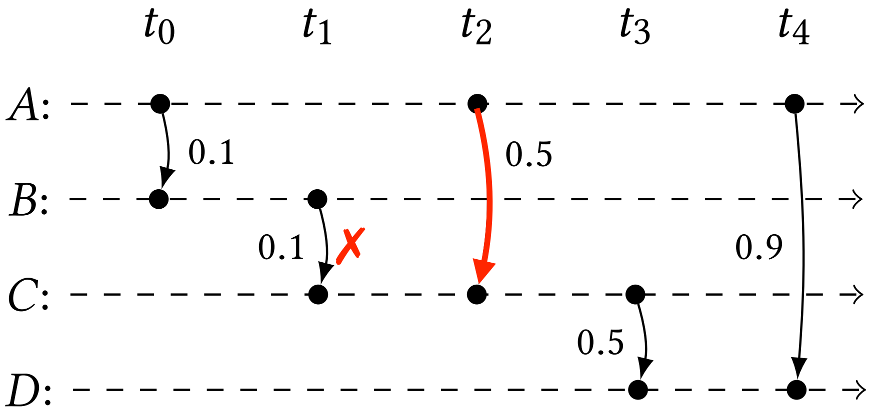
The problem of distributed information

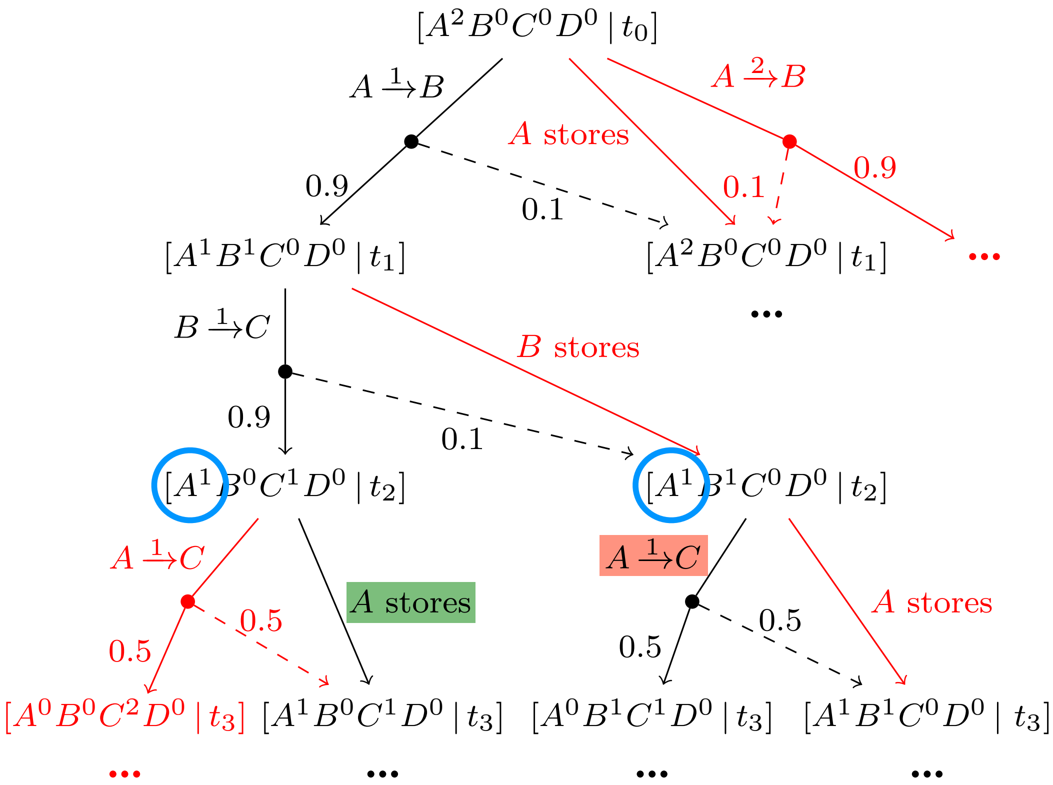


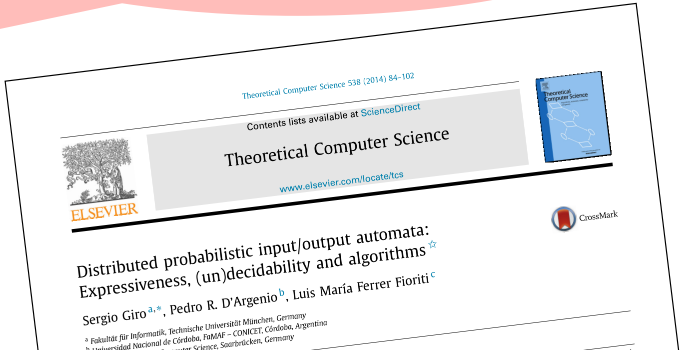


The decision has to be the same regardless the occurrences of locally unknown events

The problem of distributed information







Theoretical Computer Science 538 (2014) 84–102

Contents lists available at ScienceDirect

Theoretical Computer Science

www.elsevier.com/locate/tcs

Distributed probabilistic input/output automata: Expressiveness, (un)decidability and algorithms ☆

Sergio Giro [a,*], Pedro R. D'Argenio [b], Luis María Ferrer Fioriti [c]

[a] Fakultät für Informatik, Technische Universität München, Germany

[b] Universidad Nacional de Córdoba, FaMAF – CONICET, Córdoba, Argentina

CONICET

UNC

UNIVERSITÄT DES SAARLANDES

Third technique

Local decisions using RUCoP (L-RUCoP)

Input: number of copies N, target node T
Output: A routing table LTr_n for each node n

for all $c \leq N$ **do**
  $(S_c, Tr_c, Pr_c) \leftarrow RUCoP(G, c, T)$
end for
for all node n, time slot ts, and $c \leq N$ **do**
  $s \leftarrow Safe\_state(n, c, ts)$
  if $s \in S_c$ **then**
    $LTr_n(ts, c, ts) \leftarrow \{(k, r) \in Tr_c(s) \mid first(r) = n\}$
    $ts' \leftarrow ts$
    $rc \leftarrow (\exists\ (k, n) \in LT_r(n, ts, c, ts'))?\ k : 0$
    while $rc > 0$ **do**
      $s' \leftarrow Post(LTr_n(ts, rc, ts'))$
      $ts' = ts' + 1$
      if $s' \in S_{rc}$ **then**
        $LTr_n(ts, rc, ts') \leftarrow \{(k, r) \in Tr_{rc}(s') \mid first(r) = n\}$
      else
        break
      end if
      $rc \leftarrow (\exists\ (k, n) \in LTr_n(ts, rc, ts'))?\ k : 0$
    end while
  end if
end for
return LTr_n, for all node n.

UNC
UNIVERSITÄT DES SAARLANDES

Third technique

Local decisions using RUCoP (L-RUCoP)

Input: number of copies N, target node T

Output: A routing table LTr_n for each node n

Construct all RUCoP tables for $c \leq N$

for all $c \leq N$ **do**
    $(S_c, Tr_c, Pr_c) \leftarrow RUCoP(G, c, T)$
end for
for all node n, time slot ts, and $c \leq N$ **do**
    $s \leftarrow Safe\_state(n, c, ts)$
    if $s \in S_c$ **then**
        $LTr_n(ts, c, ts) \leftarrow \{(k, r) \in Tr_c(s) \mid first(r) = n\}$
        $ts' \leftarrow ts$
        $rc \leftarrow (\exists\ (k, n) \in LT_r(n, ts, c, ts'))?\ k : 0$
        while $rc > 0$ **do**
            $s' \leftarrow Post(LTr_n(ts, rc, ts'))$
            $ts' = ts' + 1$
            if $s' \in S_{rc}$ **then**
                $LTr_n(ts, rc, ts') \leftarrow \{(k, r) \in Tr_{rc}(s') \mid first(r) = n\}$
            else
                break
            end if
            $rc \leftarrow (\exists\ (k, n) \in LTr_n(ts, rc, ts'))?\ k : 0$
        end while
    end if
end for
return LTr_n, for all node n.

UNC
UNIVERSITÄT DES SAARLANDES

Third technique

Local decisions using RUCoP (L-RUCoP)

Input: number of copies N, target node T

Output: A routing table LTr_n for each node n

for all $c \leq N$ **do**
  $(S_c, Tr_c, Pr_c) \leftarrow RUCoP(G, c, T)$
end for
for all node n, time slot ts, and $c \leq N$ **do**
  $s \leftarrow Safe\_state(n, c, ts)$
  if $s \in S_c$ **then**
    $LTr_n(ts, c, ts) \leftarrow \{(k, r) \in Tr_c(s) \mid first(r) = n\}$
    $ts' \leftarrow ts$
    $rc \leftarrow (\exists\ (k, n) \in LT_r(n, ts, c, ts'))?\ k : 0$
    while $rc > 0$ **do**
      $s' \leftarrow Post(LTr_n(ts, rc, ts'))$
      $ts' = ts' + 1$
      if $s' \in S_{rc}$ **then**
        $LTr_n(ts, rc, ts') \leftarrow \{(k, r) \in Tr_{rc}(s') \mid first(r) = n\}$
      else
        break
      end if
      $rc \leftarrow (\exists\ (k, n) \in LTr_n(ts, rc, ts'))?\ k : 0$
    end while
  end if
end for
return LTr_n, for all node n.

Start from a safe state for node *n* with *c* copies at time slot *ts*

$$Safe\_state(A, 2, t_0) = [A^2 B^0 C^0 D^0 \mid t_0]$$

$$Safe\_state(A, 1, t_2) = [A^1 B^0 C^0 D^0 \mid t_2]$$

UNC

UNIVERSITÄT DES SAARLANDES

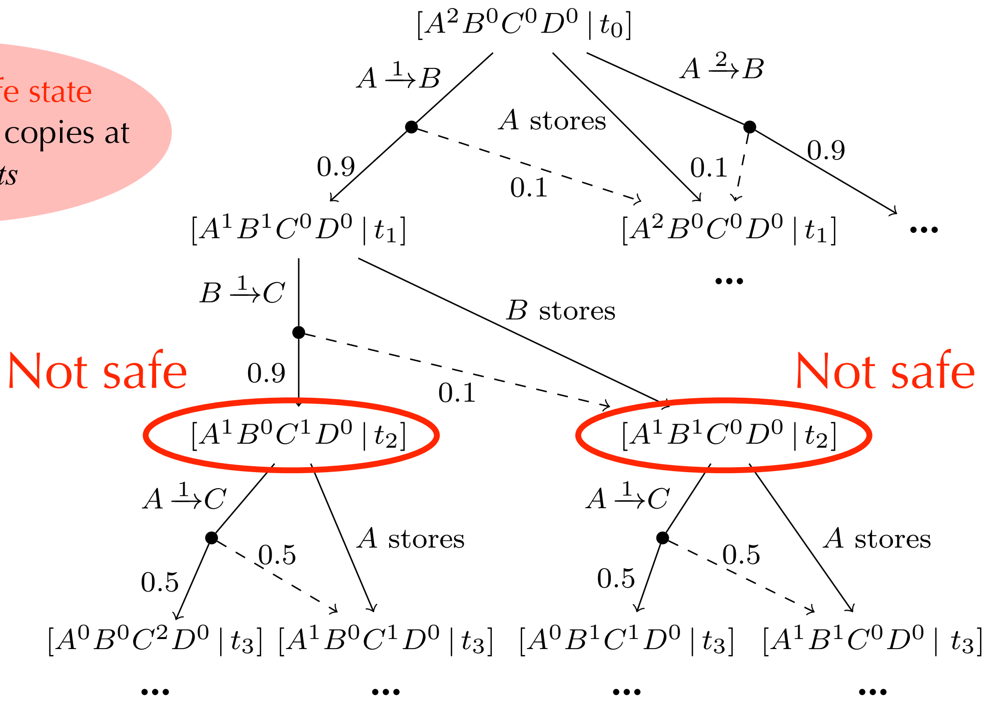
Third technique

Local decisions using RUCoP (L-RUCoP)

Input: number of copies N, target node T
Output: A routing table LTr_n for each node n

for all $c \leq N$ **do**
  $(S_c, Tr_c, Pr_c) \leftarrow RUCoP(G, c, T)$
end for
for all node n, time slot ts, and $c \leq N$ **do**
  $s \leftarrow Safe\_state(n, c, ts)$
  if $s \in S_c$ **then**
    $LTr_n(ts, c, ts) \leftarrow \{(k, r) \in Tr_c(s) \mid first(r) = n\}$
    $ts' \leftarrow ts$
    $rc \leftarrow (\exists\, (k, n) \in LT_r(n, ts, c, ts'))?\ k : 0$
    while $rc > 0$ **do**
      $s' \leftarrow Post(LTr_n(ts, rc, ts'))$
      $ts' = ts' + 1$
      if $s' \in S_{rc}$ **then**
        $LTr_n(ts, rc, ts') \leftarrow \{(k, r) \in Tr_{rc}(s') \mid first(r) = n\}$
      else
        break
      end if
      $rc \leftarrow (\exists\, (k, n) \in LTr_n(ts, rc, ts'))?\ k : 0$
    end while
  end if
end for
return LTr_n, for all node n.

Start from a safe state for node *n* with *c* copies at time slot *ts*



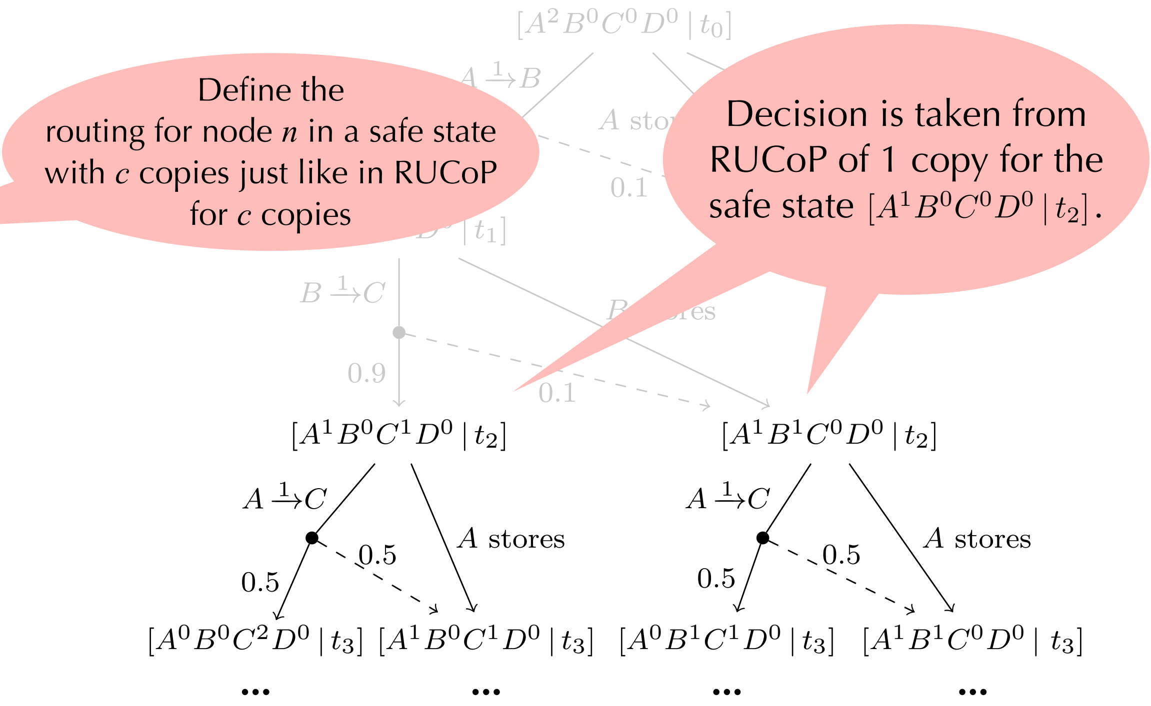
UNC
UNIVERSITÄT DES SAARLANDES

Third technique

Local decisions using RUCoP (L-RUCoP)

Input: number of copies N, target node T
Output: A routing table LTr_n for each node n

```
for all c ≤ N do
    (S_c, Tr_c, Pr_c) ← RUCoP(G, c, T)
end for
for all node n, time slot ts, and c ≤ N do
    s ← Safe_state(n, c, ts)
    if s ∈ S_c then
        LTr_n(ts, c, ts) ← {(k, r) ∈ Tr_c(s) | first(r) = n}
        ts' ← ts
        rc ← (∃ (k, n) ∈ LT_r(n, ts, c, ts'))? k : 0
        while rc > 0 do
            s' ← Post(LTr_n(ts, rc, ts'))
            ts' = ts' + 1
            if s' ∈ S_rc then
                LTr_n(ts, rc, ts') ← {(k, r) ∈ Tr_rc(s') | first(r) = n}
            else
                break
            end if
            rc ← (∃ (k, n) ∈ LTr_n(ts, rc, ts'))? k : 0
        end while
    end if
end for
return LTr_n, for all node n.
```

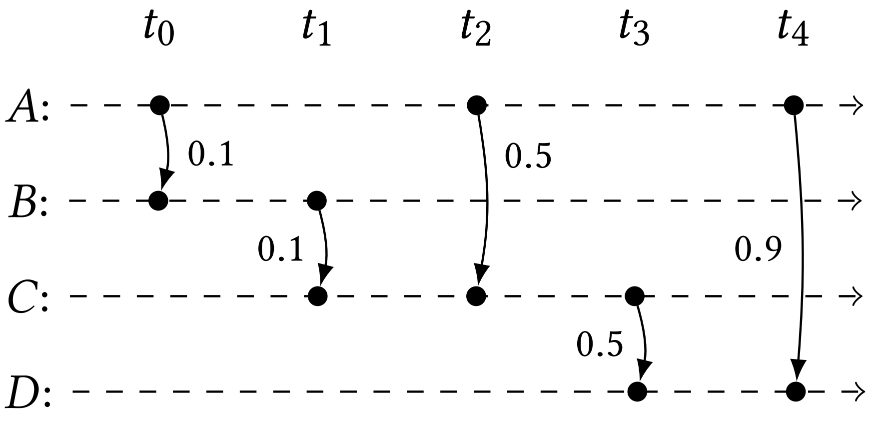


Third technique

Local decisions using RUCoP (L-RUCoP)

Input: number of copies N, target node T
Output: A routing table LTr_n for each node n

for all $c \leq N$ **do**
  $(S_c, Tr_c, Pr_c) \leftarrow RUCoP(G, c, T)$
end for
for all node n, time slot ts, and $c \leq N$ **do**
  $s \leftarrow Safe\_state(n, c, ts)$
  if $s \in S_c$ **then**
    $LTr_n(ts, c, ts) \leftarrow \{(k, r) \in Tr_c(s) \mid first(r) = n\}$
    $ts' \leftarrow ts$
    $rc \leftarrow (\exists\ (k, n) \in LT_r(n, ts, c, ts'))?\ k : 0$
    while $rc > 0$ **do**
      $s' \leftarrow Post(LTr_n(ts, rc, ts'))$
      $ts' = ts' + 1$
      if $s' \in S_{rc}$ **then**
        $LTr_n(ts, rc, ts') \leftarrow \{(k, r) \in Tr_{rc}(s') \mid first(r) = n\}$
      else
        break
      end if
      $rc \leftarrow (\exists\ (k, n) \in LTr_n(ts, rc, ts'))?\ k : 0$
    end while
  end if
end for
return LTr_n, for all node n.



Sometimes a node has some information about other nodes (e.g. when it just sent a message)

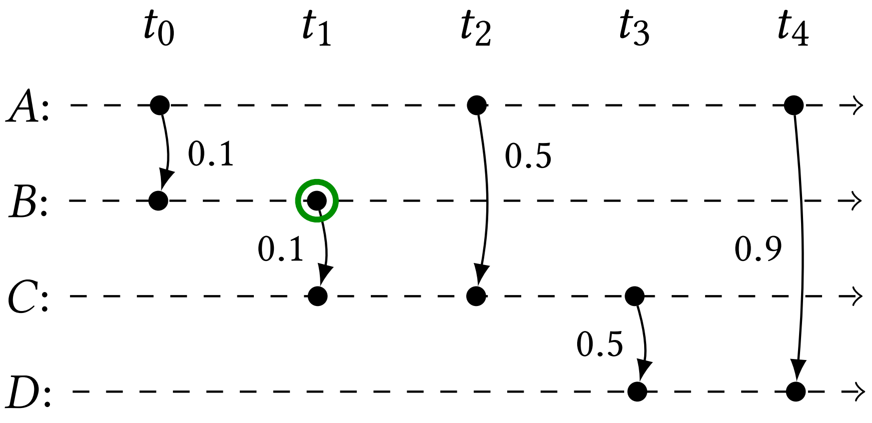
Third technique

Local decisions using RUCoP (L-RUCoP)

Input: number of copies N, target node T

Output: A routing table LTr_n for each node n

for all $c \leq N$ **do**
  $(S_c, Tr_c, Pr_c) \leftarrow RUCoP(G, c, T)$
end for
for all node n, time slot ts, and $c \leq N$ **do**
  $s \leftarrow Safe\_state(n, c, ts)$
  if $s \in S_c$ **then**
    $LTr_n(ts, c, ts) \leftarrow \{(k, r) \in Tr_c(s) \mid first(r) = n\}$
    $ts' \leftarrow ts$
    $rc \leftarrow (\exists\ (k, n) \in LT_r(n, ts, c, ts'))?\ k : 0$
    while $rc > 0$ **do**
      $s' \leftarrow Post(LTr_n(ts, rc, ts'))$
      $ts' = ts' + 1$
      if $s' \in S_{rc}$ **then**
        $LTr_n(ts, rc, ts') \leftarrow \{(k, r) \in Tr_{rc}(s') \mid first(r) = n\}$
      else
        break
      end if
      $rc \leftarrow (\exists\ (k, n) \in LTr_n(ts, rc, ts'))?\ k : 0$
    end while
  end if
end for
return LTr_n, for all node n.



t_1: B sends a copy to C who ack reception

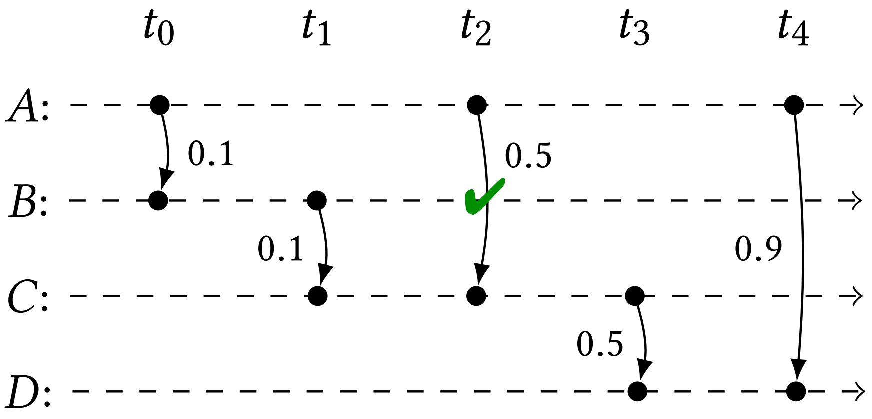
Sometimes a node has some information about other nodes (e.g. when it just sent a message)

Third technique

Local decisions using RUCoP (L-RUCoP)

Input: number of copies N, target node T
Output: A routing table LTr_n for each node n

for all $c \leq N$ **do**
  $(S_c, Tr_c, Pr_c) \leftarrow RUCoP(G, c, T)$
end for
for all node n, time slot ts, and $c \leq N$ **do**
  $s \leftarrow Safe\_state(n, c, ts)$
  if $s \in S_c$ **then**
    $LTr_n(ts, c, ts) \leftarrow \{(k, r) \in Tr_c(s) \mid first(r) = n\}$
    $ts' \leftarrow ts$
    $rc \leftarrow (\exists\ (k, n) \in LT_r(n, ts, c, ts'))?\ k : 0$
    while $rc > 0$ **do**
      $s' \leftarrow Post(LTr_n(ts, rc, ts'))$
      $ts' = ts' + 1$
      if $s' \in S_{rc}$ **then**
        $LTr_n(ts, rc, ts') \leftarrow \{(k, r) \in Tr_{rc}(s') \mid first(r) = n\}$
      else
        break
      end if
      $rc \leftarrow (\exists\ (k, n) \in LTr_n(ts, rc, ts'))?\ k : 0$
    end while
  end if
end for
return LTr_n, for all node n.



t_2: B knows C has a copy

Sometimes a node has some information about other nodes (e.g. when it just sent a message)

UNC

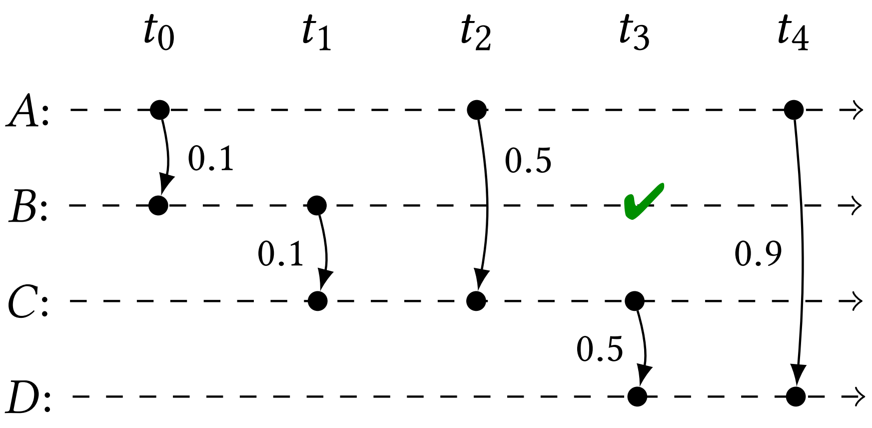
UNIVERSITÄT DES SAARLANDES

Local decisions using RUCoP (L-RUCoP)

Input: number of copies N, target node T

Output: A routing table LTr_n for each node n

for all $c \leq N$ **do**
    $(S_c, Tr_c, Pr_c) \leftarrow RUCoP(G, c, T)$
end for
for all node n, time slot ts, and $c \leq N$ **do**
    $s \leftarrow Safe\_state(n, c, ts)$
    if $s \in S_c$ **then**
        $LTr_n(ts, c, ts) \leftarrow \{(k, r) \in Tr_c(s) \mid first(r) = n\}$
        $ts' \leftarrow ts$
        $rc \leftarrow (\exists\ (k, n) \in LT_r(n, ts, c, ts'))?\ k : 0$
        while $rc > 0$ **do**
            $s' \leftarrow Post(LTr_n(ts, rc, ts'))$
            $ts' = ts' + 1$
            if $s' \in S_{rc}$ **then**
                $LTr_n(ts, rc, ts') \leftarrow \{(k, r) \in Tr_{rc}(s') \mid first(r) = n\}$
            else
                break
            end if
            $rc \leftarrow (\exists\ (k, n) \in LTr_n(ts, rc, ts'))?\ k : 0$
        end while
    end if
end for
return LTr_n, for all node n.



t_3: B knows C has a copy

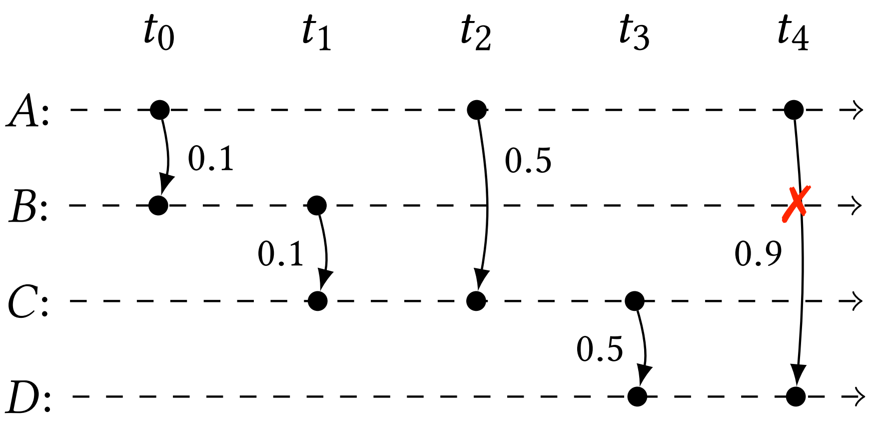
Sometimes a node has some information about other nodes (e.g. when it just sent a message)

Third technique

Local decisions using RUCoP (L-RUCoP)

Input: number of copies N, target node T
Output: A routing table LTr_n for each node n

for all $c \leq N$ **do**
  $(S_c, Tr_c, Pr_c) \leftarrow RUCoP(G, c, T)$
end for
for all node n, time slot ts, and $c \leq N$ **do**
  $s \leftarrow Safe\_state(n, c, ts)$
  if $s \in S_c$ **then**
    $LTr_n(ts, c, ts) \leftarrow \{(k, r) \in Tr_c(s) \mid first(r) = n\}$
    $ts' \leftarrow ts$
    $rc \leftarrow (\exists\ (k, n) \in LT_r(n, ts, c, ts'))?\ k : 0$
    while $rc > 0$ **do**
      $s' \leftarrow Post(LTr_n(ts, rc, ts'))$
      $ts' = ts' + 1$
      if $s' \in S_{rc}$ **then**
        $LTr_n(ts, rc, ts') \leftarrow \{(k, r) \in Tr_{rc}(s') \mid first(r) = n\}$
      else
        break
      end if
      $rc \leftarrow (\exists\ (k, n) \in LTr_n(ts, rc, ts'))?\ k : 0$
    end while
  end if
end for
return LTr_n, for all node n.



t_4: B does not know if C has a copy

Sometimes a node has some information about other nodes (e.g. when it just sent a message)

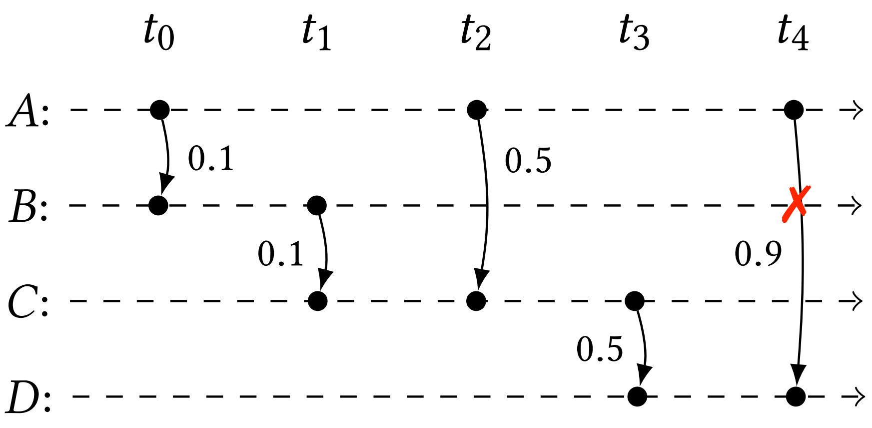
UNC
UNIVERSITÄT DES SAARLANDES

Third technique

Local decisions using RUCoP (L-RUCoP)

Input: number of copies N, target node T
Output: A routing table LTr_n for each node n

for all $c \leq N$ **do**
  $(S_c, Tr_c, Pr_c) \leftarrow RUCoP(G, c, T)$
end for
for all node n, time slot ts, and $c \leq N$ **do**
  $s \leftarrow Safe\_state(n, c, ts)$
  if $s \in S_c$ **then**
    $LTr_n(ts, c, ts) \leftarrow \{(k, r) \in Tr_c(s) \mid first(r) = n\}$
    $ts' \leftarrow ts$
    $rc \leftarrow (\exists\ (k, n) \in LT_r(n, ts, c, ts'))?\ k : 0$
    while $rc > 0$ **do**
      $s' \leftarrow Post(LTr_n(ts, rc, ts'))$
      $ts' = ts' + 1$
      if $s' \in S_{rc}$ **then**
        $LTr_n(ts, rc, ts') \leftarrow \{(k, r) \in Tr_{rc}(s') \mid first(r) = n\}$
      else
        break
      end if
      $rc \leftarrow (\exists\ (k, n) \in LTr_n(ts, rc, ts'))?\ k : 0$
    end while
  end if
end for
return LTr_n, for all node n.



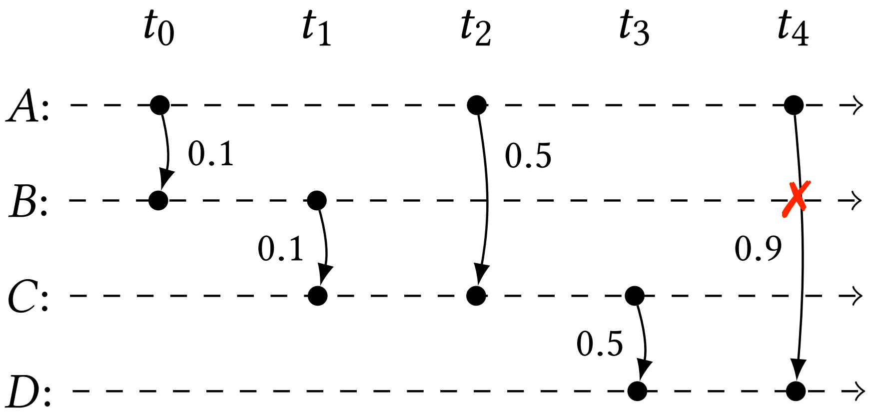
t_4: B does not know if C has a copy

Third technique

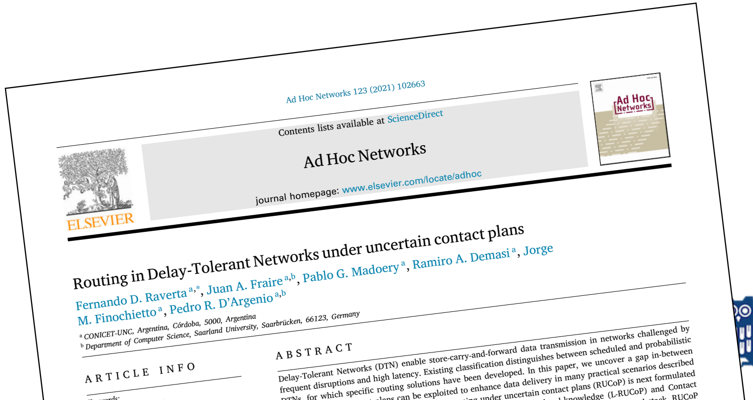
Local decisions using RUCoP (L-RUCoP)

Input: number of copies N, target node T
Output: A routing table LTr_n for each node n

for all $c \leq N$ **do**
  $(S_c, Tr_c, Pr_c) \leftarrow RUCoP(G, c, T)$
end for
for all node n, time slot ts, and $c \leq N$ **do**
  $s \leftarrow Safe\_state(n, c, ts)$
  if $s \in S_c$ **then**
    $LTr_n(ts, c, ts) \leftarrow \{(k, r) \in Tr_c(s) \mid first(r) = n\}$
    $ts' \leftarrow ts$
    $rc \leftarrow (\exists\ (k, n) \in LT_r(n, ts, c, ts'))?\ k : 0$
    while $rc > 0$ **do**
      $s' \leftarrow Post(LTr_n(ts, rc, ts'))$
      $ts' = ts' + 1$
      if $s' \in S_{rc}$ **then**
        $LTr_n(ts, rc, ts') \leftarrow \{(k, r) \in Tr_{rc}(s') \mid first(r) = n\}$
      else
        break
      end if
      $rc \leftarrow (\exists\ (k, n) \in LTr_n(ts, rc, ts'))?\ k : 0$
    end while
  end if
end for
return LTr_n, for all node n.



t_4: B does not know if C has a copy



Ad Hoc Networks 123 (2021) 102663
Contents lists available at ScienceDirect
Ad Hoc Networks
journal homepage: www.elsevier.com/locate/adhoc

Routing in Delay-Tolerant Networks under uncertain contact plans

Fernando D. Raverta [a,*], Juan A. Fraire [a,b], Pablo G. Madoery [a], Ramiro A. Demasi [a], Jorge M. Finochietto [a], Pedro R. D'Argenio [a,b]
[a] CONICET-UNC, Argentina, Córdoba, 5000, Argentina
[b] Department of Computer Science, Saarland University, Saarbrücken, 66123, Germany

ARTICLE INFO

ABSTRACT

Delay-Tolerant Networks (DTN) enable store-carry-and-forward data transmission in networks challenged by frequent disruptions and high latency. Existing classification distinguishes between scheduled and probabilistic DTNs, for which specific routing solutions have been developed. In this paper, we uncover a gap in-between ...

UNIVERSITÄT DES SAARLANDES

Fourth technique

SMC + LSS of distributed schedulers

- Resolving non-determinism in SMC+LSS

$$\mathcal{H}(\sigma.s) \bmod n$$

32-bit hash function

state as a bit vector

number of choices at s

Fourth technique

SMC + LSS of distributed schedulers

- Resolving non-determinism in SMC+LSS

$$\mathcal{H}(\sigma.s) \bmod n$$

- Resolving non-determinism in SMC+LSS+DS

$$\mathcal{H}(\sigma.(s{\downarrow}_{M_i})) \bmod n_i$$

bit vector limited to component *i*

number of choices of component *i* at *s*

Fourth technique

SMC + LSS of distributed schedulers

- Resolving non-determinism in SMC+LSS

$$\mathcal{H}(\sigma.s) \bmod n$$

- Resolving non-determinism in SMC+LSS+DS

$$\mathcal{H}(\sigma.(s{\downarrow_{M_i}})) \bmod n_i$$

bit vector limited to component *i*

number of choices of component *i* at *s*

Input: Network of VMDP $M = \|_{SV}(M_1, \ldots, M_n)$ with $[\![M]\!] = \langle S, s_I, A, T\rangle$, goal set $G \subseteq S$, $\sigma \in \mathbb{Z}_{32}$, $\mathcal{H}$ uniform deterministic, PRNG $\mathcal{U}_{\mathrm{pr}}$.

```
s := s_I
while s ∉ G do                                          // break on goal state
    if ∀ s -a-> μ: μ = { s ↦ 1 } then break             // break on self-loops
    C := { j | T(s) ∩ I_t(M_j) ≠ ∅ }                    // get active components
    i := U_pr({ j ↦ 1/|C| | j ∈ C })                    // select component uniformly
    T_i := T(s) ∩ I_t(M_i)                              // get component's transitions
    ⟨a, μ⟩ := (H(σ.s↓_{M_i}) mod |T_i|)-th element of T_i   // schedule local transition
    s := U_pr(μ)                                        // select next state according to μ
return s ∈ G
```

Fourth technique

SMC + LSS of distributed schedulers

- Resolving non-determinism in SMC+LSS

$$\mathcal{H}(\sigma.s) \bmod n$$

- Resolving non-determinism in SMC+LSS+DS

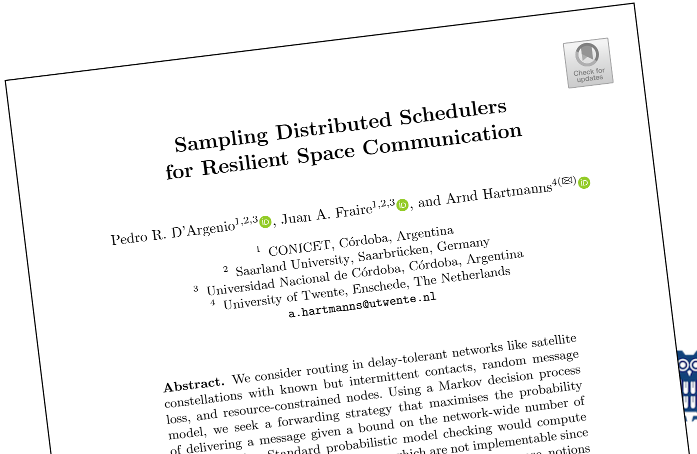
$$\mathcal{H}(\sigma.(s{\downarrow}_{M_i})) \bmod n_i$$

bit vector limited to component *i*

number of choices of component *i* at *s*

Input: Network of VMDP $M = \|_{SV}(M_1, \ldots, M_n)$ with $[\![M]\!] = \langle S, s_I, A, T \rangle$, goal set $G \subseteq S$, $\sigma \in \mathbb{Z}_{32}$, $\mathcal{H}$ uniform deterministic, PRNG $\mathcal{U}_{\text{pr}}$.

$s := s_I$
while $s \notin G$ **do** *// break on goal state*
  if $\forall\, s \xrightarrow{a} \mu \colon \mu = \{\, s \mapsto 1 \,\}$ **then break** *// break on self-loops*
  $C := \{\, j \mid T(s) \cap I_t(M_j) \neq \emptyset \,\}$ *// get active components*
  $i := \mathcal{U}_{\text{pr}}(\{\, j \mapsto \frac{1}{|C|} \mid j \in C \,\})$ *// select component uniformly*
  $T_i := T(s) \cap I_t(M_i)$ *// get component's transitions*
  $\langle a, \mu \rangle := (\mathcal{H}(\sigma.s{\downarrow}_{M_i}) \bmod |T_i|)$-th element of T_i *// schedule local transition*
  $s := \mathcal{U}_{\text{pr}}(\mu)$ *// select next state according to μ*
return $s \in G$



Sampling Distributed Schedulers for Resilient Space Communication

Pedro R. D'Argenio[1,2,3], Juan A. Fraire[1,2,3], and Arnd Hartmanns[4(✉)]

[1] CONICET, Córdoba, Argentina
[2] Saarland University, Saarbrücken, Germany
[3] Universidad Nacional de Córdoba, Córdoba, Argentina
[4] University of Twente, Enschede, The Netherlands
a.hartmanns@utwente.nl

Abstract. We consider routing in delay-tolerant networks like satellite constellations with known but intermittent contacts, random message loss, and resource-constrained nodes. Using a Markov decision process model, we seek a forwarding strategy that maximises the probability of delivering a message given a bound on the network-wide number of ...

CONICET

UNIVERSITÄT DES SAARLANDES

Experiments (delivery probability)

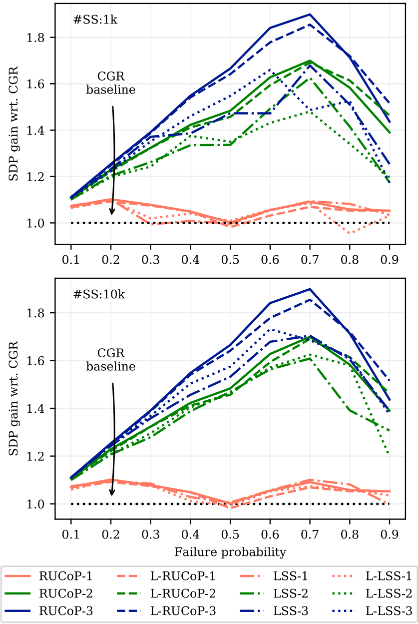


Figure 5: SDP gain over CGR in random networks.

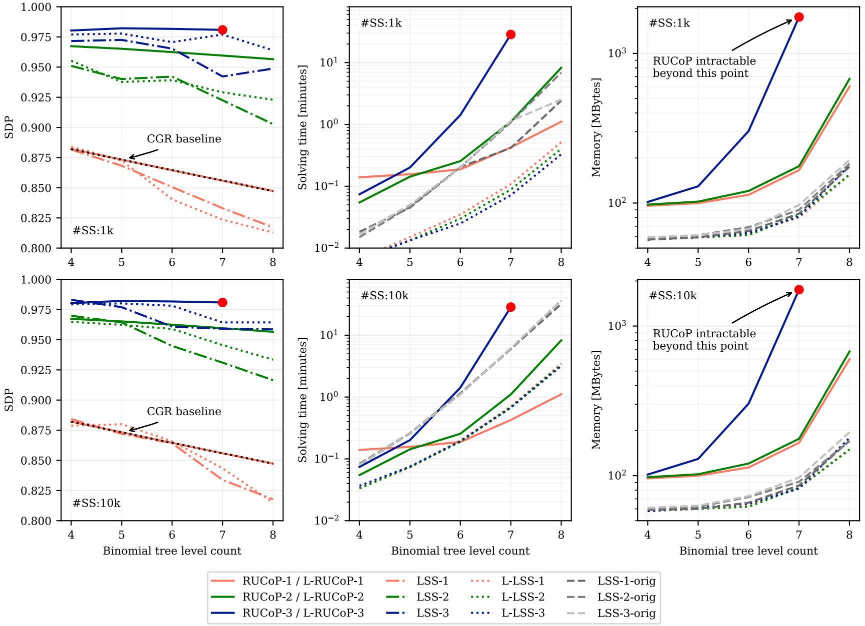


Figure 6: SDP, solving time, and memory for binomial networks with varying complexity (i.e., levels).

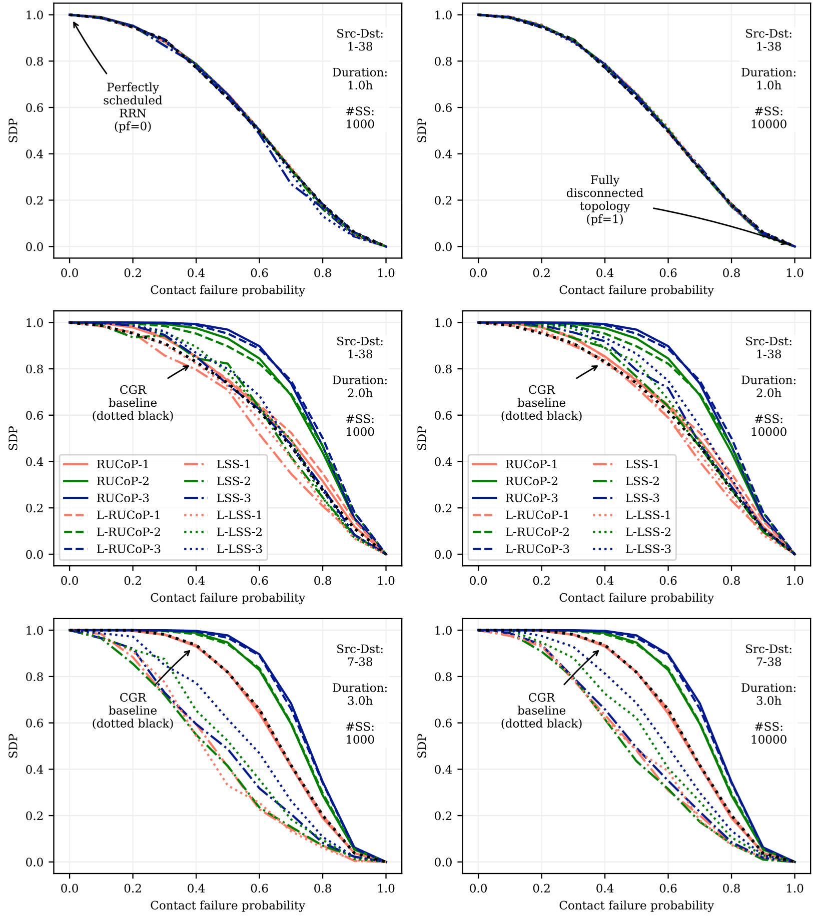


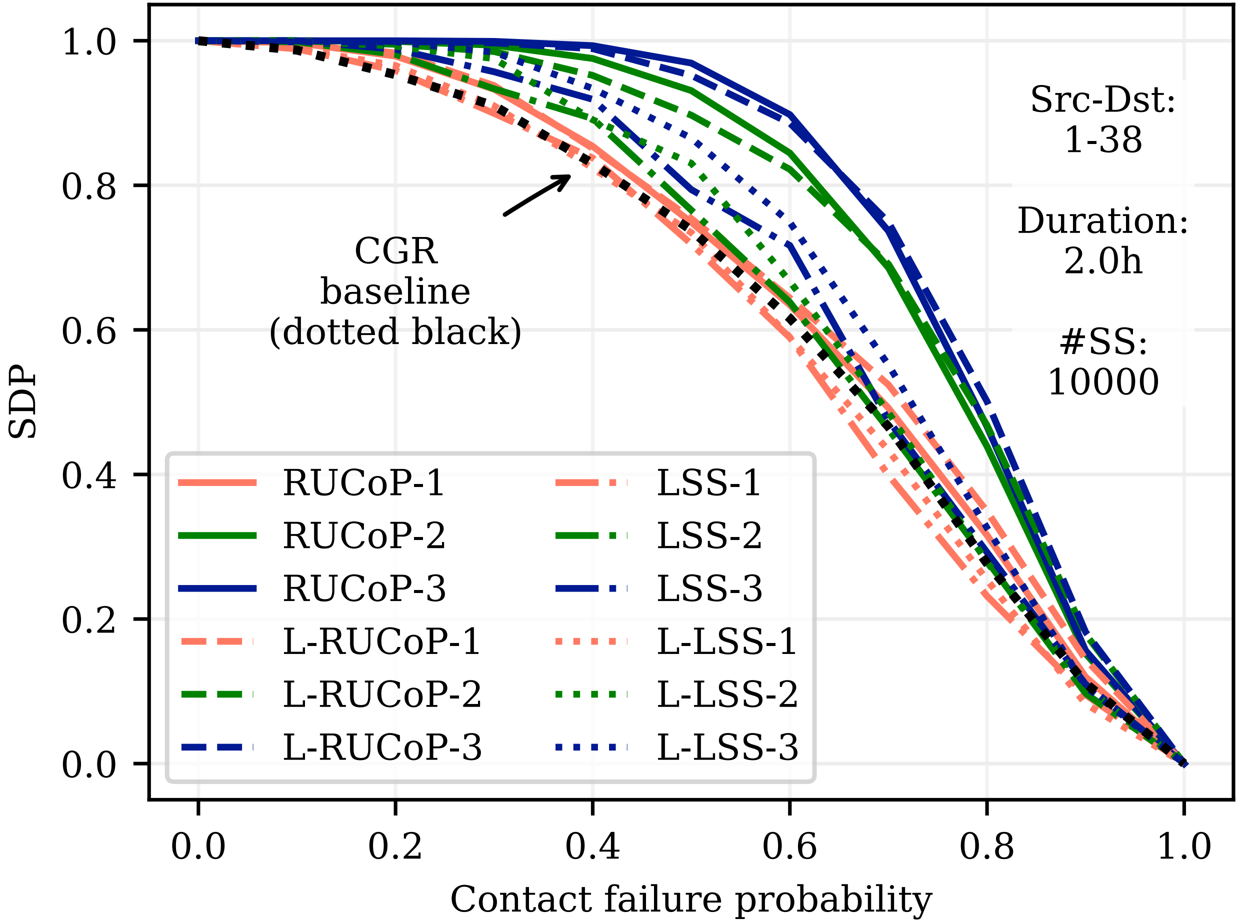
Figure 7: SDP for RRN for different source-target nodes, contact plan duration, and scheduler sampling.

CONICET

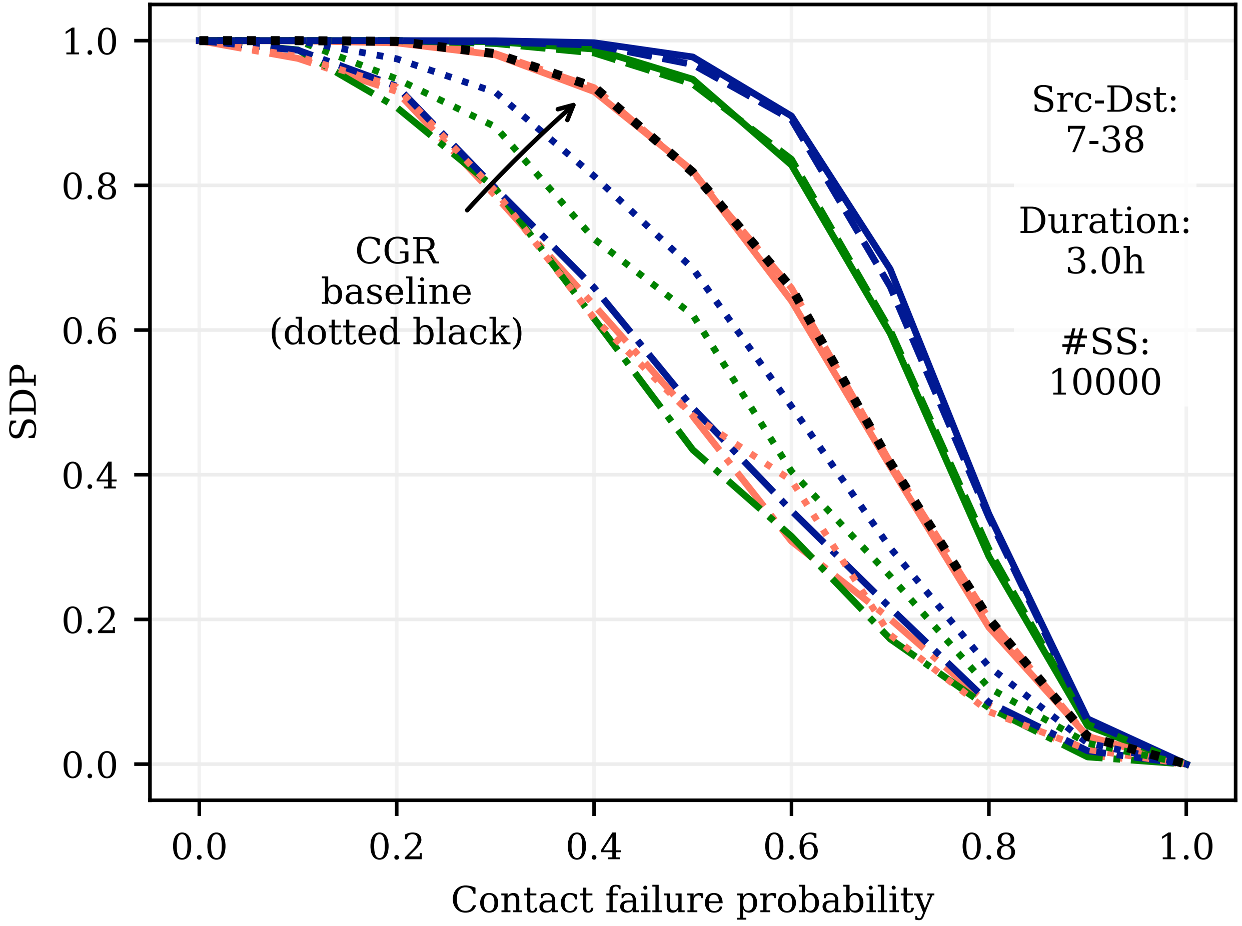
UNC

UNIVERSITÄT DES SAARLANDES

Experiments (delivery probability)



Experiments (delivery probability)



CONICET

UNC

UNIVERSITÄT DES SAARLANDES

Experiments (time & memory)

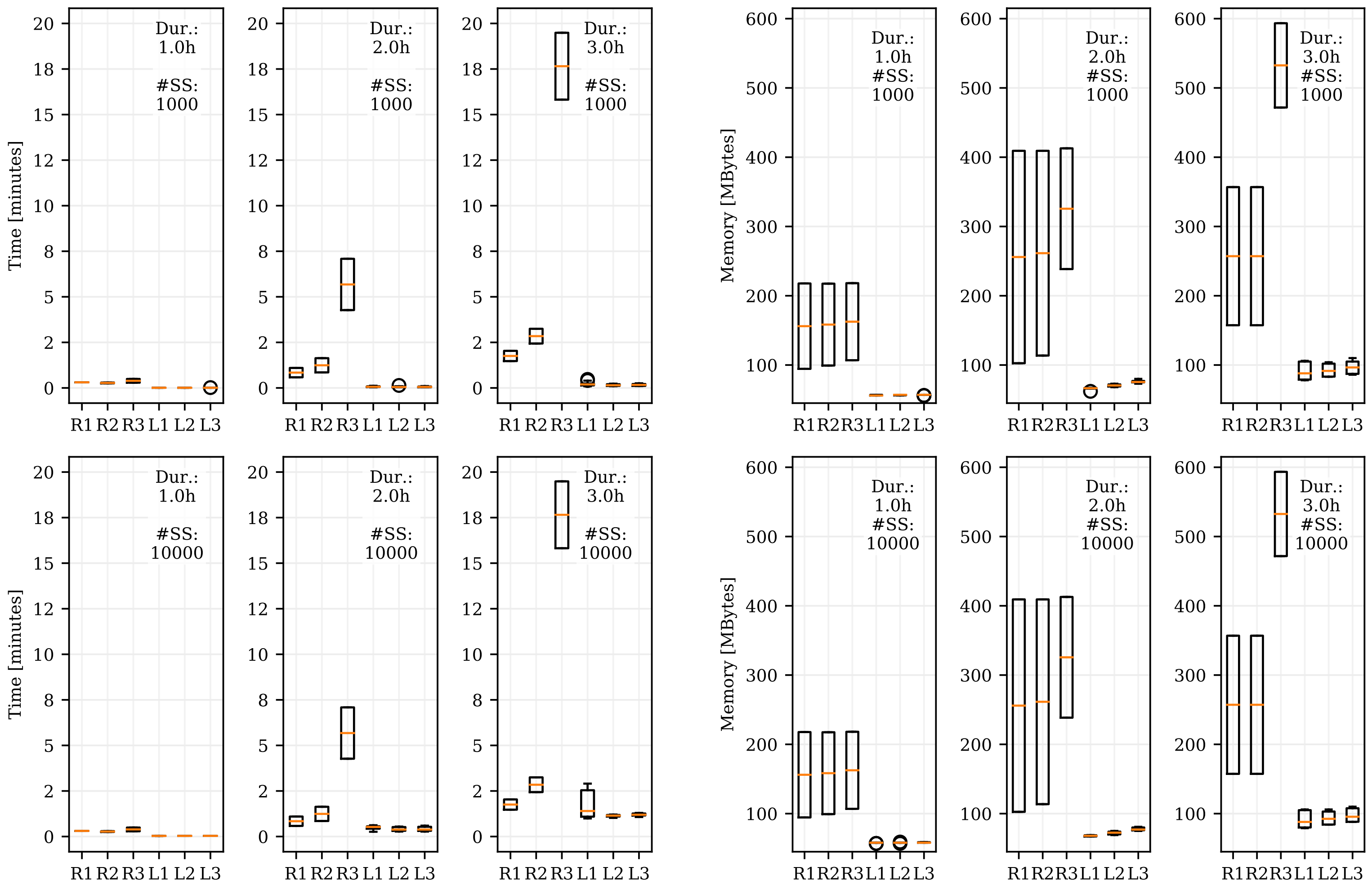


Figure 8: Solving time (left) and memory (right) for RRN for different source-target nodes, contact plan duration, and scheduler sampling (R = RUCoP, L = LSS).

CONICET

UNC

UNIVERSITÄT DES SAARLANDES

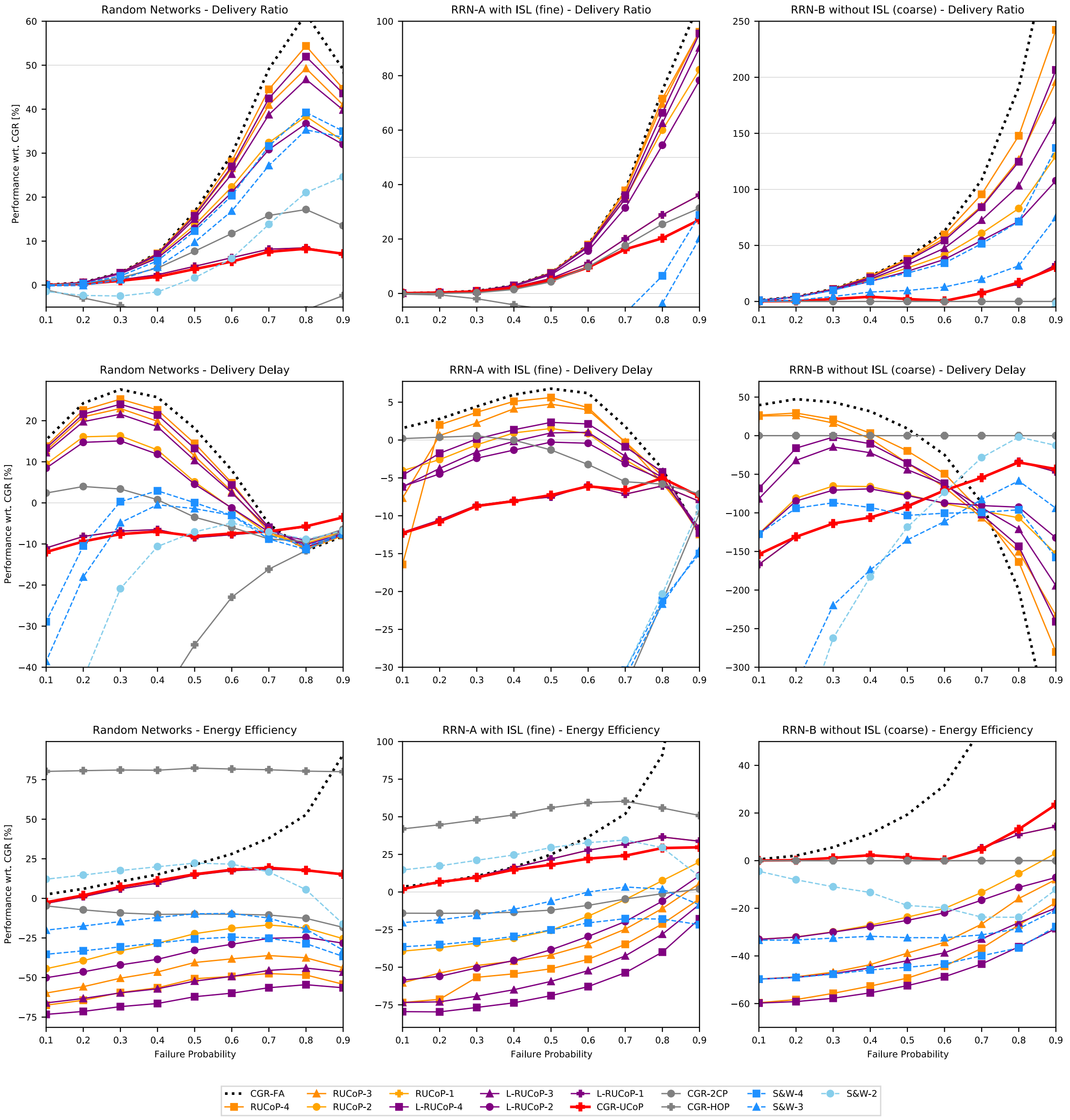
Experiments (routing efficiency)

Probability

Latency

Energy

(Only RUCoP & L-RUCoP)



Concluding remarks

- Clear increase of reliability (particularly L-RUCoP & CGR-UCoP)
- Comparison on latency is mixed. It very much depends on probability of link failure
- Particularly, (L-)RUCoP-1 & CGR-UCoP are more energy efficient than CGR
- All algorithms are demanding:
  - Routing tables need to be calculated on ground and uploaded to the satellites
  - (CGR requires uploading the contact plan, routing decisions are made on flight)
  - CGR-UCoP requires uploading an annotated contact plan, routing decisions are made on flight. However, RUCoP is needed to annotate.

Optimal Routing in Satellite DTN through Markov Decision Processes

Pedro R. D'Argenio

Universidad Nacional de Córdoba – CONICET – Universität des Saarlandes
https://cs.famaf.unc.edu.ar/~dargenio/

MISSION@INVAP - February 2022