Routing in Delay-Tolerant Networks under uncertain contact plans

Fernando D. Raverta, Juan Andrés A Fraire, Pablo G Madoery, Ramiro A Demasi, Jorge M Finochietto, Pedro R D'argenio

Accepted for publication in Ad Hoc Networks

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*

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 where uncertain contact plans can be exploited to enhance data delivery in many practical scenarios described by probabilistic schedules available *a priori*. Routing under uncertain contact plans (RUCoP) is next formulated as a multiple-copy Markov Decision Process and then exported to local-knowledge (L-RUCoP) and Contact Graph Routing extensions (CGR-UCoP) which can be implemented in the existing DTN protocol stack. RUCoP and its derivations are evaluated in a first extensive simulation benchmark for DTNs under uncertain contact plans comprising both random and realistic scenarios. Results confirm that RUCoP and L-RUCoP closely approach the ideal delivery ratio of an oracle, while CGR-UCoP improves state-of-the-art DTN routing schemes delivery ratio up to 25%.

Keywords: Delay-Tolerant Networks, Markov Decision Process, Uncertain Contact Plans

1. Introduction

The term Delay tolerant networking (DTN) was introduced by K. Fall in 2003 to designate time-evolving networks lacking of a continuous and instantaneous end-to-end connectivity [1, 2]. Since then, DTNs have drawn much attention from many researchers due to its applicability in very distinct domains including deep space [3] and near Earth communication net-

works [4], airborne networks [5], vehicular ad-hoc networks [6], mobile social networks [7], Internet of things [8] and underwater networks [9]. Indeed, delay and disruption conditions can be generated by long signal propagation time, regular node occlusion, high node mobility and reduced communication range and resources.

Although from diverse origins, partitions and delay in DTNs are tackled by a *bundle layer* that sits above specific layers of each network family [10]. The key feature of the bundle layer is a persistent storage on each DTN node to store-carry-and-forward *bundles of data* (or simply *bundles* as per DTN terminology) as transmission opportunities become available. Since data can propagate or rest in intermediate nodes for arbitrary amounts of time, DTN protocols and applications assume no immediate response from the receiver and tend to minimize end-to-end exchanges [11]. The time-evolving and partitioned nature of DTNs favor the representation of connectivity by means of *contacts*, a contact being an episode of time when a node is able to transfer data to another node.

Taxonomy The literature [2] classifies contacts in DTNs as:

- *Scheduled:* Contacts can be accurately predicted. Expected contacts can be imprinted in a *contact plan* comprising an exhaustive expression of the future network connectivity [12]. Such knowledge can be exploited to optimize resource utilization [13, 14, 15], medium access decisions [16] and routing calculations such as in Contact Graph Routing (CGR) algorithm [17, 18].

- *Probabilistic:* Contact patterns are dynamically inferred as network evolves in time. Routing is based on a topology model composed of probabilistic metrics accounting for the likelihood of meeting a given neighbour in the future [19, 20, 21]. In order to enhance delivery probability, multiple copies are sent through different paths, an approach that has also been considered for scheduled DTNs to forego the need of processing large contact plans [22].

- *Opportunistic:* No assumptions can be made on future contacts. Trivial flooding-based schemes have been used for opportunistic DTNs [23], as well as controlled flooding such as Spray-and-Wait (S&W) to reduce replication overhead [24, 25], among others opportunistic path models [26]. Also, previous research has extended scheduled routing approaches to cope with unpredictable opportunistic contacts [27].

In this paper, we claim the existence of DTN under *uncertain schedules* or *uncertain contact plans*, which are not properly covered by the existing DTN classification:

- *Uncertain:* Contacts whose materialization can differ from the original plan with a given probability available *a priori*. For example, expected contacts have a chance of being affected by well-known failure modes or by an incomplete or inaccurate (but bounded) knowledge of the system status by the time the schedule was computed. In other words, while in probabilistic DTNs the probability is assigned to a next-hop node (i.e, the probability of meeting a given node, based on contact history), uncertain DTNs under uncertain contact plans assign probabilities to forthcoming contacts (i.e., the probability of meeting a given node in a given time episode in the future).

Uncertain DTNs. Uncertain DTNs differ from perfectly scheduled DTNs in the nature of their contacts, which are no longer certain to occur (uncertain contacts have an associated probability of existing or failing). They also differ from probabilistic DTNs in the features of the model used to represent and reason about the network dynamics. Instead of relying on abstract node's visibility patterns (learned on the fly), uncertain DTNs exploit time-dependant probabilistic information of the forthcoming connectivity episodes encoded in the so-called uncertain contact plan (computed in advance). An uncertain contact plan is a probabilistic schedule that includes information regarding the probability of future contacts to diverge from the plan. The advantage of accounting for this knowledge in uncertain DTNs is that it can be used to make specific routing, forwarding and bundle replication decisions over the most reliable routes towards a destination, thus optimizing the data delivery chances.

The different nature of probabilistic and uncertain DTNs can also be appreciated in the route structure. Routes in probabilistic DTNs are expressed as a *sequence of nodes* through which the bundle shall be forwarded. There is no specific information on when the route hops will actually happen, just a time-averaged expectation based on inter-nodes visibility patterns. On the other hand, uncertain contact plans bring the notion of uncertain contact, which is also probabilistic, but encoding timing information is unavailable in traditional probabilistic schemes. Thus, and similarly to scheduled DTNs, routes in uncertain DTNs are constructed as a *sequence of uncertain contacts*,

which renders a delivery probability through each path, and thus, more granular and accurate (but also challenging) decision making opportunities.

Applications for uncertain DTNs include DTN networks based on a schedule of fault-prone nodes (unreliable space networks [28]), uncertain mobility patterns (public vehicle networks [29]), interference-sensitive communication links (cognitive radio [30]), or third-party carriers with limited availability (backbone links with known reliability [31]). Indeed, the uncertain contact plan including contacts probabilities can be computed by specific network models (i.e., fault-prone satellite trajectories), empirically estimated in a controlled environment (i.e., lab or simulation setup), or made available from existing statistics (i.e., interference reports). As a result, an uncertain contact plan can be conveniently pre-computed instead of dynamically learned by nodes as in probabilistic DTNs, removing the burden of a training phase, and benefiting from highly accurate routing schemes for uncertain DTNs as introduced in this paper.

Previous Works. Previous works have addressed the survivability properties of time-varying networks [32], as well as the problem of reliable topology design in DTN [33]. However, to the best of the authors' knowledge, the problem of reliable route determination based on uncertain contact plans has been overlooked. Authors have already studied how schedule-aware (i.e., CGR) and schedule-agnostic (i.e., S&W) routing schemes behave under uncertain contact plans in [28, 34, 35] (probabilistic routings such as MaxProp [20] and Prophet [19] were disregarded as they are based on learning phases during network operations). These papers essentially showed that existing routing schemes only perform well on their respective domains (perfectly scheduled or fully opportunistic), while significant room for improvement was identified for scenarios with uncertain schedules. In order to evaluate the potential improvement, the authors in [36] have approached the problem with a first theoretical formulation based on probabilistic model checking techniques [37, 38, 39], where the contact plan with its respective fault probabilities is modelled as a Markov Decision Process (MDP). Although this first approach provided a compelling optimal solution for single-copy routing, replication-based heuristics remained an open topic. Exception to this statement is a recent publication that addressed the multi-copy DTN routing problem by means of approximated simulations techniques based on distributed schedulers [40]. However, simulation techniques lack the required optimality guarantee that formal MDP models can provide.

Contributions. In this paper, we present Routing under Uncertain Contact Plans (RUCoP), a comprehensive framework to execute reliable routing under uncertain contact plans. RUCoP embraces single copy [36] and extends it to multiple-copy routing in an overcoming MDP model expression. As the fact of considering multiple copies renders the focus of [36] unsuitable, we propose a novel MDP formulation accompanied by a specific resolution algorithm. The fact of using MDP arises naturally since the Markov kernel corresponds to probabilistically quantified uncertainty on the contacts while the decisions (or the non-determinism) of the MDP correspond to the possibilities of routing decisions of each node at a given time. The RUCoP model is the first of its kind to consider *rerouting*, which models both the fault detection and reaction time of the DTN routing agent. Modeling this crucial and practical aspect allows us to introduce L-RUCoP (a variation that uses only local information available on each node) and CGR-UCoP (an extension to CGR that materializes routing under uncertain contact plans in existing DTN protocol stacks). We evaluate and compare the RUCoP, L-RUCoP and CGR-UCoP in an appealing benchmark comprising networks with random failures as well as realistic case studies of Low-Earth Orbit (LEO) satellite networks with uncertain inter-satellite and ground contacts. Results provide compelling evidence that RUCoP provides the adequate framework to route in uncertain DTNs.

To summarize, contributions in this paper are enumerated as follows:

1. We present a new uncertain DTN classification and model;
2. We introduce RUCoP to route on uncertain DTNs based on a theoretical MDP formulation;
3. We propose L-RUCoP and CGR-UCoP as concrete practical application approaches derived from RUCoP; and
4. We evaluate RUCoP, L-RUCoP and CGR-UCoP in realistic fault-prone LEO satellite networks.

The remaining of this paper is organized as follows. Section 2 presents the uncertain DTN network model which is used to construct the RUCoP model and derived L-RUCoP and CGR-UCoP in Section 3. A comparison benchmark and subsequent results are presented, analyzed and discussed in Section 4. Finally, the paper is concluded in Section 5.

2. Uncertain DTN Model

2.1. Uncertain Time-Varying Graph

In order to model a time-evolving and uncertain DTN network, the time-varying graph proposed in [32] is extended by uncertainty functions into an Uncertain Time-Varying Graph defined as follows.

Definition. An Uncertain Time Varying Graph $\mathcal{G} = (G, \mathcal{T}, p_f, \varsigma, f_{dd})$ is a Graph composed of the following components:

1. **Underlying (static) digraph** $G = (V, E)$. Represents the connectivity of the network that remains stable during a time slot.

2. **Time slot** $\mathcal{T} \subseteq \mathbf{T}$, where $\mathbf{T}$ is the time domain (e.g. the natural numbers). $\mathcal{T} = \{t_0, t_1, ..., t_T\}$ is a discrete and finite time span set, where T is an integer indicating the horizon of interest, measured in the number of slots. The slot length in $\mathcal{G}$ can be adjusted in order to capture (i) the topological changes, and (ii) the minimum period of time it takes a node to realize a link has failed to establish.

3. **Edge failure probability function** $p_f : E \times \mathcal{T} \to [0, 1]$. It indicates the probability an edge will not occur as expressed in the uncertain contact plan, i.e., a topology change respects the original schedule. Indeed, $p(e, t) = 1 - p_f(e, t)$, where $p(e, t)$ stands for the edge e success probability at the time slot t. A success probability of $p(e, t) = 0$ indicates no contact is present at this edge.

4. **Edge delay function** $\varsigma : E \times \mathcal{T} \to \mathcal{T}$. It models the time data spend on crossing an edge between two nodes. When $\varsigma(e, t) = 0$, the time is insignificant compared with the time slot duration, i.e., the data is delivered immediately. The value of the edge delay function stands for the number of time slots (i.e., $\varsigma(e, t)$ is an integer) required for the target node to receive the traffic.

5. **Edge failure detection delay function** $f_{dd} : E \times \mathcal{T} \to \mathcal{T}$. It stands for the time it takes to detect a contact did not occur as expected. As with the edge delay function, $f_{dd}(e, t)$ is expressed as a number of time slots. In DTN protocol terminology, $f_{dd}(e, t)$ would represent the bundle custody acknowledge timeout. In general, $f_{dd}(e, t) \geq \varsigma(e, t)$.

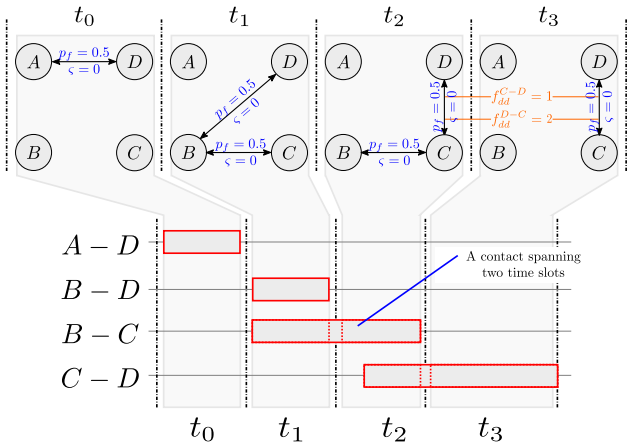


Figure 1: Uncertain time-varying graph model example with 4 nodes, 4 time slots $\mathcal{T}$ and 4 contacts.

Fig. 1 illustrates an example DTN graph modeled by an uncertain time-varying graph. All edges present in $G = (V, E)$ are configured with a failure probability function $p_f = 0.5$ and a delay function $\varsigma = 0$. In the model, a contact between two nodes can span several time slots, such as the $B-C$ case spanning t_1 and t_2. Also, a time slot can represent long and stable topological periods with the same underlying digraph, such as t_3 with an edge between $C-D$. At t_2, node C will be able to detect a failure on edge $C-D$ and react at the beginning of t_3, as its failure detection delay $f_{dd}^{C-D} = 1$. However, node D will not do so before t_3 terminates since $f_{dd}^{D-C} = 2$. Indeed, contacts in DTN are unidirectional and can have different properties on the forward and return link.

Failure probability p_f in $\mathcal{G}$, ς, and f_{dd} are expressed on a per-slot basis. Two modeling approaches with different interpretations are envisioned on this regard: coarse and fine grained slotting.

Coarse-grained slotting: When time-slots are designed to contain full contacts (i.e., $B - D$ contact in t_1 in Fig. 1), then p_f represents the failure probability of the whole contact. In other words, the whole contact exists or the whole contact fails. In such case, an $f_{dd} = 0$ would model the case where the failure of the contact is detected and reacted upon immediately at contact start time, while an $f_{dd} = 1$ would represent the case where the contact is declared as failed only once it is finalized. This approach is appropriate to model transient failures in nodes, for instance. Also, coarse-grained slotting

is particularly appealing for networks with sparse contacts, which can be bounded by a single time slot t_n in $\mathcal{T}$.

Fine-grained slotting: When a contact spans several smaller time slots (i.e., $B - C$ contact in t_1 and t_2 in Fig. 1), p_f is the probability of failure of each of the slotted episodes comprising the contact. In this case, a finer-grain slotting can be exploited to model independent transmission attempts within the contact. An $f_{dd} = 1$ would thus model a timeout equal to the bundle transmission duration and the round trip time delay for receiving a delivery confirmation. Fine-grained slotting can be used to model contacts where poor channel conditions or interference from other sources render a successful transmission uncertain.

2.2. Fault Detection and Rerouting

Rerouting after effective detection of a failed contact or transmission attempt is a fundamental practical aspect to model the overall data flow in DTNs under uncertain contact plans. Single route reliability estimations such as those in [32] can result inaccurate in practice when nodes detect and act upon unexpected failures. However, the phenomena is not trivial.

Consider the example of Fig. 2 in which all links have a failure probability $p_f = 0.5$ with the exception of $S \rightarrow B$ at t_0 and $C \rightarrow D$ at t_1 which have a failure probability of $p_f = 0.80$ and $p_f = 0.75$ respectively. The transmission delay $\varsigma = 0$ and failure detection delay is $f_{dd} = 1$ for all links and data flows from source S to destination D. Without considering rerouting, routes via node A ($S \rightarrow A \rightarrow B \rightarrow D$) and via node C ($S \rightarrow C \rightarrow D$) would be equally reliable because they both account for a *successful delivery probability* (SDP) of 0.125. However, rerouting after failure detection might challenge this calculation. If the link between $A \rightarrow B$ fails in the route via A, then the data will not reach the destination. But, if the contact between $C \rightarrow D$ fails, it is still possible to relay the data to node E after t_1, which has another route towards D. In a context where rerouting is possible with $f_{dd} <= 1$, the probability of a bundle to reach the destination via node C is 75% higher ($SDP = 0.219$). Otherwise, for $f_{dd} >= 2$, the delivery probability through C remains $SDP = 0.125$.

In the following section, we claim the rerouting effect in an uncertain time varying graph can be properly represented by means of Markov Decision Processes.

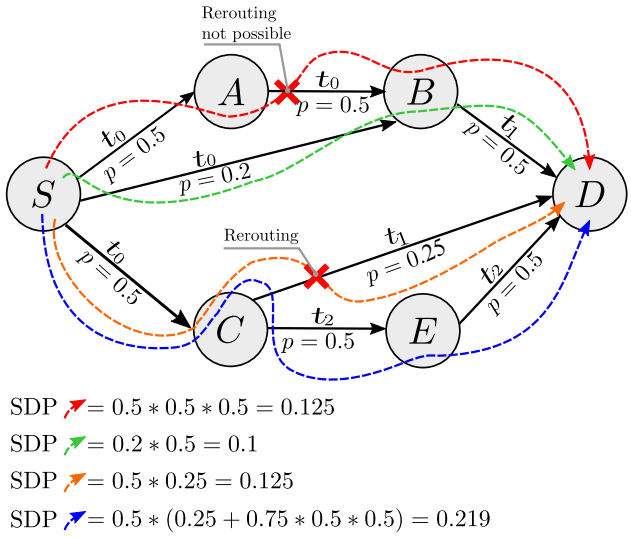


Figure 2: Rerouting is possible when node C detects a failure at the end of t_1 ($f_{dd} = 1$) and has an alternative route to D at t_2 that arrives on the same time slot ($\varsigma = 0$).

3. Routing Under Uncertain Contact Plans

3.1. Markov Decision Process

A Markov Decision Process (MDP) is a mathematical structure that allows for the modelling of discrete-time systems in which the interaction between non-deterministic and probabilistic behaviour is central [41, 42]. Thus, MDPs provide an appropriate framework for modelling decision making on systems under probabilistically quantified uncertainty. In its simplest form, a MDP $\mathcal{M}$ is a tuple $(S, Act, \mathbf{P}, s_0)$ where

- S is a finite set of states with initial state $s_0 \in S$,
- Act is a finite set of actions, and
- $\mathbf{P} : S \times Act \times S \to [0, 1]$ is a transition probability function such that $\sum_{s' \in S} \mathbf{P}(s, \alpha, s') \in \{0, 1\}$, for all $s \in S$ and $\alpha \in Act$.

If $\sum_{s' \in S} \mathbf{P}(s, \alpha, s') = 1$, α is said to be *enabled* in s. In this case, $\mathbf{P}(s, \alpha, \cdot)$ can be interpreted as the probability distribution of choosing the next state, conditioned to the fact that the system is in state s and action α has been chosen. We notice that it is usually required that at least one action is enabled in every state. Since the problem ahead is a reachability problem

(instead of a cost or reward problem), the usual reward function does not play any role and hence we have omitted it in the definition of MDPs.

The intuitive operational behaviour of the MDP $\mathcal{M}$ is as follows. The computation of $\mathcal{M}$ starts at the initial state s_0. Assume now the computation has taken n steps and reached state s_n. At this moment one of the enabled actions in s_n, say α_{n+1}, is chosen to resolve the non-determinism at this state. The next state s_{n+1} is now sampled randomly according to distribution $\mathbf{P}(s_n, \alpha_{n+1}, \cdot)$.

Different types of properties could be required to a MDP. The usual objective is to find a *policy* that maximizes or minimizes the likelihood of the given property. A *policy* is a function $\pi : S \to Act$ that defines the decision to be made in a possible resolution of the non-determinism[1]. Thus, limiting the MDP $\mathcal{M}$ to the choices of the policy π defines a Markov chain for which probabilities can be calculated.

We are particularly interested on maximizing the probability to reach a state in the set of *goal states* $B \subseteq S$ from the initial state s_0, say $Pr_{s_0}^{\max}(reach(B))$. (In our case, B is the set of states in which bundles have been successfully delivered). Moreover, we want to obtain the maximizing policy. This problem can be solved using the Bellman equations as follows [38]. Let $S^{=0} \subseteq S$ be the set of states whose probability of reaching a state in B is 0. ($S^{=0}$ could be calculated in $\mathcal{O}(|S|)$.) For each state $s \in S$, define a variable x_s which represents the maximum probability of reaching a goal state in B from s, that is $x_s = Pr_s^{\max}(reach(B))$. Then, precisely the vector $(x_s)_{s \in S}$ is the least solution of the following equation system:

$$
\begin{aligned}
x_s &= 1 && \text{if } \; s \in B \\
x_s &= 0 && \text{if } \; s \in S^{=0} \\
x_s &= \max_{\alpha \in Act(s)} \sum_{t \in S} \mathbf{P}(s, \alpha, t) \cdot x_t && \text{if } \; s \in S \backslash (S^{=0} \cup B)
\end{aligned}
$$

Besides, the maximizing policy $\pi^{\max}$ can be obtained as follows:

$$
\pi^{\max}(s) = \operatorname*{argmax}_{\alpha \in Act(s)} \sum_{t \in S} \mathbf{P}(s, \alpha, t) \cdot x_t \qquad \text{if } \; s \in S \backslash (S^{=0} \cup B)
$$

[1]Polices could be more complex, depending on the whole history rather than the current state, and selecting randomly among the enabled actions. The definition given here correspond to the so called *memoryless* and *deterministic* policies, which is sufficient for our purposes.

If $s \in S^{=0} \cup B$, $\pi^{\max}(s)$ is not interesting as s is already a goal state, or it cannot reach it.

Reachability properties are standard properties in probabilistic model checkers such as PRISM [43]. Indeed, we have successfully modeled single-copy routing in DTNs under uncertain contact plans in PRISM [36] and derived optimal routes in this case. Unfortunately, PRISM cannot deal with the size of models we required, specially when we consider DTNs with multiple copies.

3.2. RUCoP

In order to determine the upper delivery probability bound for routing with N copies in a DTN, we have developed Routing under Uncertain Contact Plans (RUCoP). RUCoP is an MDP formulation which encodes all possible routing decisions for an uncertain DTN network based on its uncertain time-varying graph representation and traffic parameters, comprising source, target and number of copies allowed. This information is encoded in states and transitions. Table 1 summarizes the notation used throughout the remaining of this section.

Table 1: Notation reference

Symbol	Description
Uncertain DTN Model (Section 2)	
$p_f(e,t)$	Failure probability for link e at time slot t
$\varsigma(e,t)$	Delay for link e at time slot t
$f_{dd}(e,t)$	Failure detection delay for link e at time slot t
$\mathcal{T}$	Set of time slots
RUCoP Core Algorithm (Section 3.2)	
G_{t_i}	Underlying digraph G for time slot t_i
$\mathcal{S}_{t_{end}}$	Set of successful final states
$\mathcal{S}_{t_i}$	Set of states at time slot t_i
$cp(c)$	Number of copies at node c
$\mathcal{C}_{t_i}$	Set of nodes carrying copies in time slot t_i
$pred^{+}_{G_{t_i}}(c)$	Set of all nodes in G_{t_i} reaching c in at least one hop
$path_{G_{t_i}}(c',c)$	Set of directed path from c' to c in G_{t_i}
$\mathcal{P}_c$	Set of paths leading to c

R	Set of rules (i.e. pairs of nr. of copies and a path)
$\mathcal{R}_c$	Set of c-compatible sets of rules (i.e. set of rules transmitting exactly $cp(c)$ copies from c)
$Tr(s)$	Set of actions leading to state s (an action is a set of rules distributing exactly num_copies)
pr_R	Successful probability of action R
$Pr(s)$	Successful delivery probability of state s
$SDP(R, s, t)$	Successful probability for action R starting from state s at time slot t (Algorithm 2)
$get\_prev\_state(s, R)$	Returns the state from which action R leads to s
$best\_action(s)$	The action from s maximizing the delivery prob.
$RUCoP(G, c, T)$	Algorithm 1
RUCoP SDP Computation (Section 3.2)	
$\wp(X)$	Power set of X
$contacts(R)$	Set of links involved in action R
$state\_af\_fl(R, s, fs)$	Leading state when set of failures fs happen
pr_{fs}	Probability of all links in fs failing
pr_R	Successful delivery probability of action R
$SDP(s)$	Successful delivery probability of state s
L-RUCoP (Section 3.3)	
$Safe\_state(n, c, ts)$	State in which node n has all c copies available
$LTr_n(\_, \_, \_)$	Routing table for node n
$Post(LTr_n(ts, rc, ts'))$	The state known by node n after action $LTr_n(ts, rc, ts')$
CGR-UCoP (Section 3.4)	
$Rl_n(ts)$	Set of partial routes computed by CGR at node n for time slot ts
r	A partial route computed by CGR
$r[i]$	ith contact in the partial route r
$Pr_n(ts)$	Prob. of delivering a copy from n at time slot ts
$src(e)$	Source of link e
$tgt(e)$	Destination of link e
$SDP_{CGR}(r, ts)$	Bundle's delivery prob. through partial route r

States. Each state in RUCoP contains information of the number of copies

present on each node in the network at a given time slot. For example, in the network of Fig. 2, the initial state would be $s_{t_0} = [S^n A^0 B^0 C^0 D^0 E^0 | t_0]$ denoting that s_{t_0} has n copies of the bundle at time 0, the start time of t_0. A state $s_{t_3} = [S^s A^a B^b C^c D^d E^e | t_3]$ at the beginning of t_3 would represent a successful delivery of data to D as long as $d >= 1$, meaning at least one copy of the data arrived at D at the end of the time horizon. Since it is assumed copies cannot be created or deleted, $s + a + b + c + d + e = n$ in all states.

Transitions. Transitions between states in RUCoP are composed by actions, which can be of two types: (i) *transmission transitions* imply a node perform a non-deterministic transmission through one (single-hop) or more edges (multi-hop) in G, and (ii) *store transitions* model the case where a node decides to keep the bundle in memory during the time slot. Since state transitions imply a routing action on the nodes, the terms transitions and actions are used interchangeably in RUCoP.

Tree Construction. To build the state and transition tree, RUCoP starts from the desirable *successful states* where data was delivered to the destination. Next, it considers states from the previous time slot that can lead to the current state, whether by transmitting data through a path or by keeping it in storage. In order to determine which state of the previous time slot can arrive to the current state, a set of transmissions transition are constructed. Finally, between these transitions, the one which has the highest delivery probability is chosen and noted. The process repeats until the *initial state* is reached. In order to determine the probability of a given transition, all cases of failures and successful link establishments are considered: **(*i*)** when a contact fails, data remains stored in the transmitting node where new transmission transitions can be considered after f_{dd}, and **(*ii*)** when a link is established, the data is transmitted through it, and it can be sent again after ς.

For example, the RUCoP model in Fig. 3 corresponds to the network of Fig. 2, when a single copy is sent. The successful state $[S^0 A^0 B^0 C^0 D^1 E^0 | t_3]$ is at the last time slot t_3, which can be reached either by receiving data through $C \rightarrow E \rightarrow D$ (multi-hop transmission) or by having data already stored at D since t_2. In turn, these intermediate states can only be reached if a $C \rightarrow D$ transition or a $B \rightarrow D$ transition takes place on t_1. It can be observed that, if $C \rightarrow D$ fails, C can detect the failure ($f_{dd} = 1$) and store the data for further transmission transitions. However, if the contact $B \rightarrow D$ fails, data will remain in B leading to state $[S^0 A^0 B^1 C^0 D^0 E^0 | t_2]$, from which

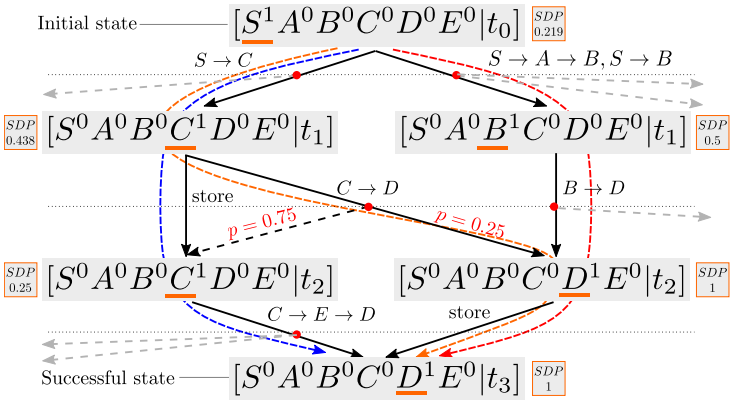


Figure 3: RUCoP MDP tree based on the network of Fig. 2 for 1 copy.

the successful state cannot be reached (i.e., delivery cannot occur). This is represented by the grey dotted arrow outgoing the red dot. A similar (but more involved) situation happens in transitions $S \to A \to B$ and $S \to B$ outgoing the initial state: if the $S \to A$, $A \to B$ or $S \to B$ contacts fail, data will remain in S leading to state $[S^1A^0B^0C^0D^0E^0|t_1]$ or in A leading to state $[S^0A^1B^0C^0D^0E^0|t_1]$. Both of these states are failure consequences of transitions $S \to A \to B$ and $S \to B$, which have no possibility of reaching the successful state (greyed-out arrows in the figure). In this simple example, all non-deterministic transmission transitions (red dots in the figure), except $C \to D$, lead to states unable to reach the successful state as long as some contact in the transition fails. Indeed, constructing the tree backwards avoids exploring such states. It is interesting to note that if detection delay would have been $f_{dd} = 2$ in $C \to D$ at t_1, the dashed line indicating failure path would lead to $[S^0A^0B^0C^1D^0E^0|t_3]$, which is also unable to reach the successful state. In other words, by the time when C detects the failure, the contact $C \to E$ would have already passed.

Successful delivery probability. While constructing the tree, RUCoP keeps track of the successful delivery probability. Indeed, $SDP = 1$ at the successful states, and is updated as the tree is built backwards in time following the Bellman equations. For each non-deterministic transmission transition, the probability of arriving to the successful state is computed. *SDP* is updated with the highest probability. Once the initial state is reached, the *SDP* will capture the maximum delivery probability possible. By navigating the tree top-down, the most reliable routing decisions (i.e., policy) can be

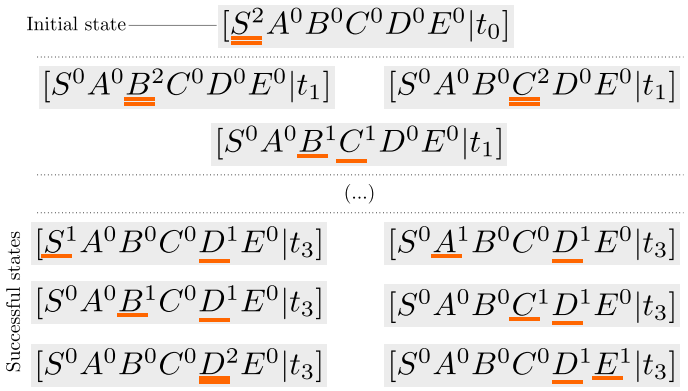


Figure 4: RUCoP MDP tree based on the network of Fig. 2 for 2 copies.

obtained by choosing transitions that lead to states with the best *SDP* metric. In the example, S should route the data to C at t_0 for an $SDP = 0.219$, and C should try to send data to D at t_1 for an $SDP = 1$, or to E at t_2 in case of failure.

Multiple copies. The proposed RUCoP expression is specifically designed to model the state of the network with multiple copies. Naturally, modeling multiple copies notably increases the number of transitions and states in the MDP. For example, when two instances of the bundle are considered, transmission transitions can involve the transmission of either one or two bundles of data, and transmission failures might occur in any of the used links. As illustrated in Fig. 4, six successful states are possible and should be considered with two copies on the example network. For instance, node S can choose to transmit one copy via A and one via C to maximize the delivery chances. However, for larger networks with several copies, constructing the model requires of the following formal expression of the RUCoP algorithm.

The algorithm: For simplicity, we present the algorithm limited to uncertain time varying graphs where the edge delay is insignificant and the edge failure detection delay is always one time slot (i.e. $\varsigma(e,t) = 0$ and $f_{dd}(e,t) = 1$ for all edge $e \in E$ and time slot $t \in \mathcal{T}$). At the end of this section, we hint the required modifications of the algorithm to deal with the general treatment of these delays. Algorithm 1 lists the formal steps required to construct and solve the RUCoP MDP for these type of networks with a maximum of *num_copies* number of copies.

Initially, a set of all possible *successful states* $\mathcal{S}_{t_{end}}$ are generated (line 1)

Algorithm 1: The RUCoP algorithm

Input: Uncertain time varying graph $\mathcal{G}$, *num_copies*, Target

Output: Explored states $\mathcal{S}$, Routing table Tr, Successful delivery probability Pr

determine *successful states* $\mathcal{S}_{t_{end}}$ for *num_copies*
$\mathcal{S} \leftarrow \mathcal{S}_{t_{end}}$
for all $t_i \in \mathcal{T}$, starting from t_{end-1} **do**
  $\mathcal{S}_{t_i} \leftarrow \varnothing$
  for all state $s \in \mathcal{S}_{t_{i+1}}$ **do**
    determine *carrier nodes* $\mathcal{C}_{t_i}$
    for all node $c \in \mathcal{C}_{t_i}$ **do**
      $\mathcal{P}_c \leftarrow \{c\} \cup \bigcup_{c' \in pred^+_{G_{t_i}}(c)} path_{G_{t_i}}(c', c)$
      $\mathcal{R}_c \leftarrow \left\{R \subseteq \{0, \dots cp(c)\} \times \mathcal{P}_c \mid \sum_{(k,\rho)\in R} k = cp(c)\right\}$
    end for
    $Tr(s) \leftarrow \left\{ \bigcup_{c\in\mathcal{C}_{t_i}} R_c \mid \forall c \in \mathcal{C}_{t_i} : R_c \in \mathcal{R}_c \right\}$
    for all $R \in Tr(s)$ **do**
      $s' \leftarrow get\_previous\_state(s, R)$
      $\mathcal{S}_{t_i} \leftarrow \mathcal{S}_{t_i} \cup \{s'\}$
      $pr_R \leftarrow SDP(R, s', t_i)$
      if $Pr(s')$ is undefined or $Pr(s') < pr_R$ **then**
        $Pr(s') \leftarrow pr_R$
        $best\_action(s') \leftarrow R$
      end if
    end for
    $\mathcal{S} \leftarrow \mathcal{S} \cup \mathcal{S}_{t_i}$
  end for
end for
return $\mathcal{S}$, Tr, Pr

and added to the set of explored states (line 2). A state is successful if at least one copy is in the target node and exactly *num_copies* are distributed among all nodes. RUCoP builds the MDP backwards from this set with the goal of arriving to the initial state. To this end, all reachable states $\mathcal{S}_{t_i}$ within each t_i in $\mathcal{T}$ are determined starting from an empty set (line 4 and loop starting at line 5). $\mathcal{S}_{t_i}$ is subsequently populated with all states that are able to reach some state in $\mathcal{S}_{t_{i+1}}$ by means of actions involving bundle transmissions, data storage or a combination of them when multiple copies are presented.

Thus, for each state $s \in \mathcal{S}_{t_{i+1}}$, the loop proceeds in two parts. The first one (lines 6-11) determines the set of actions $Tr(s)$ that successfully lead to state s. The second one (lines 12-20) calculates the predecessor states for each of these actions which are then included in the set of states $\mathcal{S}_{t_i}$ of the preceding time slot and for which its successful delivery probability (SDP) is calculated.

To obtain $Tr(s)$, the set of *carrier nodes* $\mathcal{C}_{t_i}$ in s is first determined (line 6). A carrier node is a node holding at least one copy of the bundle. An action in $Tr(s)$ is a set of *rules*. A rule is a tuple (k, ρ) where ρ is a valid single-hop or multiple-hop path (or route) in the underlying digraph G for the time slot t_i (G_{t_i}), and k is the number of copies transmitted through this path; thus, $k \leq cp(c)$, where $cp(c)$ is the number of copies the target carrier node c has in its buffer.

For each carrier node $c \in \mathcal{C}_{t_i}$, the set $\mathcal{P}_c$ of paths leading to c in the current contact digraph G_{t_i} is determined. This is calculated in line 8 where: (i) $pred^+_{G_{t_i}}(c)$ is the set of all nodes in G_{t_i} reaching c in at least one hop, and (ii) $path_{G_{t_i}}(c', c)$ is the set of all paths in G_{t_i} starting in node c' and ending in c containing all distinct vertices. In addition, $\mathcal{P}_c$ always contains the trivial path c which is intended to represent that data remains stored in the node c for the current time slot.

Notice that the different copies may arrive at node c through multiple paths. Thus $\mathcal{R}_c$ contains the set of all *compatible* sets of rules that indicate how the copies arrive to c (line 9). By compatible, we mean that the numbers of copies delivered by the rules in such set should add up to exactly $cp(c)$, i.e., $R \in \mathcal{R}_c$ whenever $\sum_{(k,\rho)\in R} k = cp(c)$.

Finally (line 11), an action $R \in Tr(s)$ is a set of rules so that, for each carrying node $c \in \mathcal{C}_{t_i}$, the subset of all rules in R leading to c is compatible (i.e., $R \cap (\mathbb{N} \times \mathcal{P}_c) \in \mathcal{R}_c$). A rule R never delivers more than *num_copies* in

total. This is guaranteed by the fact that $\sum_{c \in \mathcal{C}_{t_i}} cp(c) \leq num\_copies$.

To illustrate the exposed concepts, Fig. 5 lists carrier and predecessor nodes, paths, rules and transition for state $s = [S^0A^0B^2C^1E^0D^0|t_1]$ corresponding to the network in Fig. 2 when 3 copies are allowed. Since B carries two copies, each compatible set of rules leading to B may have up to two rules. The resulting transition includes all the possible transmissions of the copies succesfully reaching the evaluated state.

Each transition $R \in Tr(s)$ is considered individually to determine its corresponding previous state s' (line 13) which is added to the set of previous sates $\mathcal{S}_{t_i}$ (line 14). Notice that s' may already be present in $\mathcal{S}_{t_i}$ if it is the source of a previously analysed transition $\hat{R} \in Tr(\hat{s})$ for some previously selected state $\hat{s} \in \mathcal{S}_{t_{i+1}}$. In line 15, the probability induced by transition R is calculated calling function *SDP* (which we will shortly discuss). If this is the first time state s' is visited (hence its successful deliver probability $Pr(s')$ is not yet defined) or its previously assigned probability is smaller than the newly found pr_R (line 16) $Pr(s')$ is set to the new maximum pr_R (line 17) and indicated that this is achieved through transition R (line 18). (This is implementing the maximum of the Bellman equations.) Finally, all states explored at time slot t_i are added to the set of of explored states $\mathcal{S}$ (line 21). The next iteration will explore the new set of states $\mathcal{S}_{t_i}$ and so forth until t_0 is reached.

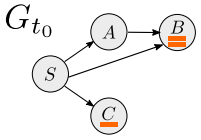
If the initial state s_{t_0} –where all copies are present at the source node– is part of the set of explored states $\mathcal{S}$, then there is a series of actions (stored in array *best_action*) that lead to a successful delivery of the data with an optimal SDP equal to $Pr(s_{t_0})$. If the initial state s_{t_0} is not present in $\mathcal{S}$, then $Pr(s_{t_0})$ is undefined and the SDP for the model is 0, implying no routing decision can be successful in delivering the bundle of data to the intended destination.

Calculating SDP. Algorithm 2 shows how SDP is computed for a transition R leaving a state s. We let $contacts(R)$ be a set containing every link involved in some path in R, and iterate for every possible combination of link failures (line 2). Thus, a failure set $fs \in \wp(contacts(R))$ stands for a set of links that failed to be established whereas $contacts(R) - fs$ are the links that successfully transmitted the data. Depending on fs, a transition comprising several hops can leave the bundle in different nodes in the path and thus lead to different states. The state *to_state* to which the network would evolve to if links in fs failed is computed (line 3). Notice that *to_state* may not be a

$$[S^0 A^0 \underline{B}^2 \underline{C}^1 D^0 E^0 | t_1]$$

Carrier nodes, G_{t_0}, and pred. nodes.

$$\mathscr{C}_{t_0} = \{B, C\}$$



$$pred^+_{G_{t_0}}(C) = \{S\}$$
$$pred^+_{G_{t_0}}(B) = \{S, A\}$$

$$\mathcal{P}_B = \{\mathrm{B}, A \to B,\ S \to B,\ S \to A \to B\}$$

$$\begin{aligned}\mathcal{R}_B = \{&\{(\underline{2}, B)\},\ \{(\underline{2}, A \to B)\},\ \{(\underline{2}, S \to B)\},\ \{(\underline{2}, S \to A \to B)\},\\ &\{(\underline{1}, B), (\underline{1}, A \to B)\},\ \{(\underline{1}, B), (\underline{1}, S \to B)\},\\ &\{(\underline{1}, B), (\underline{1}, S \to A \to B)\},\ \{(\underline{1}, A \to B), (\underline{1}, S \to B)\},\\ &\{(\underline{1}, A \to B), (\underline{1}, S \to A \to B)\},\ \{(\underline{1}, S \to B), (\underline{1}, S \to A \to B)\}\}\end{aligned}$$

$$\mathcal{P}_C = \{C, S \to C\}$$

$$\mathcal{R}_C = \{\{(\underline{1}, C)\}, \{(\underline{1}, S \to C)\}\}$$

$$\begin{aligned}Tr = \{&\{(\underline{2}, B), (\underline{1}, C)\},\ \{(\underline{2}, A \to B), (\underline{1}, C)\},\ \{(\underline{2}, S \to B), (\underline{1}, C)\},\\ &\{(\underline{2}, S \to A \to B), (\underline{1}, C)\},\ \{(\underline{1}, B), (\underline{1}, A \to B), (\underline{1}, C)\},\\ &\{(\underline{1}, B), (\underline{1}, S \to B), (\underline{1}, C)\},\ \{(\underline{1}, B), (\underline{1}, S \to A \to B), (\underline{1}, C)\},\\ &\{(\underline{1}, A \to B), (\underline{1}, S \to B), (\underline{1}, C)\},\\ &\{(\underline{1}, A \to B), (\underline{1}, S \to A \to B), (\underline{1}, C)\},\\ &\{(\underline{1}, S \to B), (\underline{1}, S \to A \to B), (\underline{1}, C)\},\\ &\{(\underline{2}, B), (\underline{1}, S \to C)\},\ \{(\underline{2}, A \to B), (\underline{1}, S \to C)\},\\ &\{(\underline{2}, S \to B), (\underline{1}, S \to C)\},\ \{(\underline{2}, S \to A \to B), (\underline{1}, S \to C)\},\\ &\{(\underline{1}, B), (\underline{1}, A \to B), (\underline{1}, S \to C)\},\\ &\{(\underline{1}, B), (\underline{1}, S \to B), (\underline{1}, S \to C)\},\\ &\{(\underline{1}, B), (\underline{1}, S \to A \to B), (\underline{1}, S \to C)\},\\ &\{(\underline{1}, A \to B), (\underline{1}, S \to B), (\underline{1}, S \to C)\},\\ &\{(\underline{1}, A \to B), (\underline{1}, S \to A \to B), (\underline{1}, S \to C)\},\\ &\{(\underline{1}, S \to B), (\underline{1}, S \to A \to B), (\underline{1}, S \to C)\}\}\end{aligned}$$

Figure 5: Nodes, rules and transition example in RUCoP

successfully delivering state in which case $SDP(to\_state)$ will not be defined and the probability of delivering of this particular combination of failing links is 0. The conditional statement of line 4 takes this into account. Thus, if to_state is a successful delivering state, the probability pr_{fs} of this failure set to happen is calculated (line 5) and the contribution to the total probability of successfully delivering the data when links in fs fail is added up (line 6).

Algorithm 2: Successful Delivery Probability (SDP)

Input: Transition R, state s, current time slot t
Output: SDP of current action

1: $pr_R \leftarrow 0$
2: **for all** $fs \in \wp(contacts(R))$ **do**
3: $\quad to\_state \leftarrow state\_after\_failures(R, s, fs)$
4: $\quad$ **if** $SDP(to\_state)$ is defined **then**
5: $\quad\quad pr_{fs} \leftarrow \left(\prod_{e \in contacts(R)-fs} 1-p_f(e,t)\right) * \left(\prod_{e \in fs} p_f(e,t)\right)$
6: $\quad\quad pr_R \leftarrow pr_R + pr_{fs} * SDP(to\_state)$
7: $\quad$ **end if**
8: **end for**
9: **return** pr_R

Fig. 6 illustrates the calculation of the SDP for transition $\{(1, S \rightarrow A \rightarrow B), (1, S \rightarrow C)\}$ which is a transition from $[S^2A^0B^0C^0E^0D^0|t_0]$ (the initial state) to $[S^0A^0B^1C^1E^0D^0|t_1]$ when 2 copies are allowed and successfully transmitted. In other words, when no failure is observed ($\wp = \emptyset$), copies are successfully transmitted to B and other to C with a probability of $p = 5^3 = 0.125$. However, different failures can lead to 5 possible alternative states with an accumulated probability of $1 - 0.125$. Two of these have an undefined SDP, implying they have no further possibility of delivering the data to the destination. This particular transition is the one with the highest SDP for $[S^2A^0B^0C^0E^0D^0|t_0]$ so that it stands for the optimal decision for forwarding two copies from S to D in the example network.

Complexity analysis. First of all, notice that, if $N_c = |pred^+_{G_{t_i}}(c)|$, then $|\mathcal{P}_c| \leq N_c! \cdot \sum_{i=0}^{N_c} \frac{1}{i!} < eN_c!$ and hence $|\mathcal{R}_c| \leq \binom{|\mathcal{P}_c|+cp(c)}{cp(c)} < \binom{eN_c!+cp(c)}{cp(c)}$.

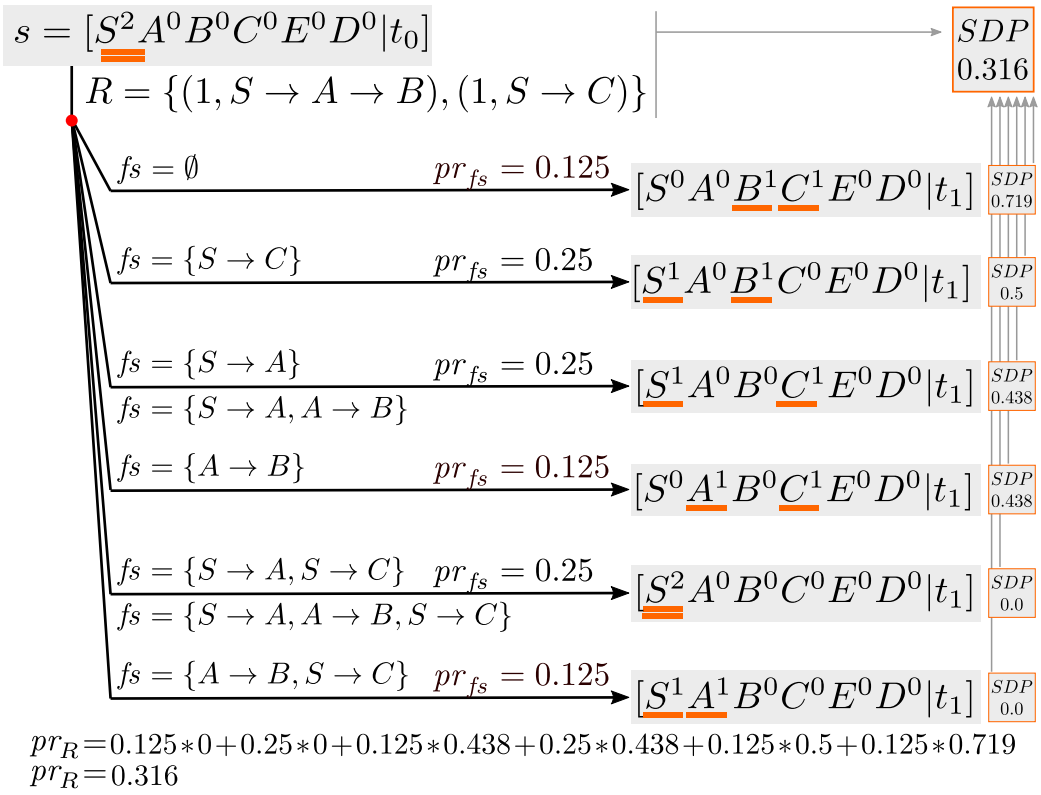


Figure 6: RUCoP updates for transition $\{(1, S \to A \to B), (1, S \to C)\}$ in-going state $[S^0 A^0 B^1 C^1 E^0 D^0 | t_1]$

From this, we have that

$$|\mathit{Tr}(s)| \;=\; \prod_{c\in\mathcal{C}_{t_i}} |\mathcal{R}_c| \;<\; \prod_{c\in\mathcal{C}_{t_i}} \binom{eN_c! + cp(c)}{cp(c)} \;\leq\; \binom{eN! + K}{K}^K.$$

The last inequality follows from taking the worst case values, knowing that $cp(c) \leq \mathit{num\_copies}$ and $|\mathcal{C}_{t_i}| \leq \mathit{num\_copies}$ (there can never be more carrier nodes than allowed copies), and letting $N = \max_{t\in\mathcal{T}} \max_{c\in\mathcal{C}_t} N_c$ and $K = \mathit{num\_copies}$. The calculation of $\mathcal{P}_c$ is done by a search algorithm of complexity $O(N_c!)$, and the construction of $\mathcal{R}_c$ and $\mathit{Tr}(s)$ are by enumeration. Thus, the complexity of lines 5-10 in Algorithm 1 is $O\left(\binom{eN!+K}{K}^K\right)$.

Focusing now in Algorithm 2, notice that $\mathit{contacts}(R)$ can contain, in the worst case, all edges present in G_{t_i}; therefore $|\mathit{contacts}(R)| \leq N_c^2 \leq N^2$. Calculation in line 5 involves a multiplication of $|\mathit{contacts}(R)|$ terms. Hence, taking into account that the loop repeats $|\wp(\mathit{contacts}(R))|$ times, the complexity of this algorithm is $O\left(N^2 2^N\right)$.

From the previous observation, we see that the body of loop in lines 4-20 in Algorithm 1 is $O\left(N^2 2^N \binom{eN!+K}{K}^K\right)$. By observing that that $|\mathcal{S}_{t_i}| = \binom{|V|+K}{K}$, we can finally conclude that the complexity of Algorithm 1 is:

$$O\left(N^2 \cdot 2^N \cdot \binom{eN!+K}{K}^K \cdot \binom{|V|+K}{K} \cdot |\mathcal{T}|\right)$$

Where V is the set of all nodes in the network and $\mathcal{T}$ is the time span under consideration. We remark that, although in the worst case $N = |V|$, we normally expect N —the maximum number of nodes reaching a carrier node in a single time slot— to be significantly smaller than the number of nodes in V.

Taking into account Stirling's aproximation to factorials, we finally notice that the algorithm is in 2-EXPTIME. However, in practice, we manage to have a satisfactory performance in practical use cases as it can be seen in Section 4.3.

Link and failure detection delays. Algorithm 1 is presented for networks with insignificant link delays and one time slot failure detection delay in all cases. In the general case, for networks where $\varsigma(e,t) > 0$ or $f_{dd}(e,t) > 1$, for some link $e \in E$ and time slot $t \in \mathcal{T}$, additional bookkeeping is necessary. In particular, it is not possible to only count copies of bundles. In this case, it will be necessary to distinguish each copy and annotate it with the time

slot in which it is available for transmission (either because of the delay after transmission, or because of the delay after failure). This will have to be carefully considered, especially, when calculating $path_{G_{t_i}}(c, c')$ (line 7 in Algorithm 1) or the target state in $state\_after\_failures(R, s, fs)$ (line 3 in Algorithm 2). In addition, this modification will have an impact on the (already high) complexity of the algorithm.

3.3. L-RUCoP

RUCoP is based on a global view of the system: decisions are taken based on the current state of the network. This implies that distributed nodes need to know where all copies are in the network at any moment, including remote and potentially disconnected nodes. Although optimal, this is impossible to achieve in highly partitioned DTNs where delays and disruptions force nodes to decide based on partial local knowledge [44, 45, 46]. A simple example of this phenomenon is presented in Fig. 7. Two decisions are possible at node A in t_2, it can *store* the copy or forward it to C. However, which is optimal, might depend on weather the other copy is on B or C at t_2 (and also on p_{f_4} and p_{f_5}). Nonetheless, because A was out of reach of B and C, or because the contact $A-C$ is unidirectional or highly delayed, node A may not be able to know which is the global status of the system nor which is the optimal action in t_2. The aim of this section is to propose a derivation of RUCoP that can be implementable in DTNs where knowledge is restricted to each node's local view. We coin this practical approach *local* RUCoP (L-RUCoP).

L-RUCoP takes routing decisions on each local node n using a pre-filled routing matrix $LTr_n(t_s, c, t_i)$. In this entry, t_s indicates the "safe" time slot and it is normally the next one after the copies have been received, c is the current number of copies that n holds, and $t_i \geq t_s$ is the current time slot. $LTr_n(t_s, c, t_s)$ will contain the best decision n can take assuming no knowledge of the network. This is the same as if assuming that n holds all copies and no other copy is in the system. Therefore $LTr_n(t_s, c, t_s)$ contains exactly all routing decisions made by RUCoP for the state in which n contains all c copies and no copies are in the other nodes. Nonetheless, if n decides to keep some copies $rc < c$ and only send $c - rc$ copies, in the following time slots n has certain knowledge of the previously distributed copies that may be handy to improve the decision on the routing of the remaining rc copies. We illustrate this peculiarity using the contact plan in Fig. 7 assuming that $p_{f_1} = p_{f_2} = 0.1$, $p_{f_3} = p_{f_4} = 0.5$ and $p_{f_5} = 0.9$. The optimizing route for $LTr_A(t_2, 1, t_2)$, in which A has no knowledge of the past, is to deliver the only

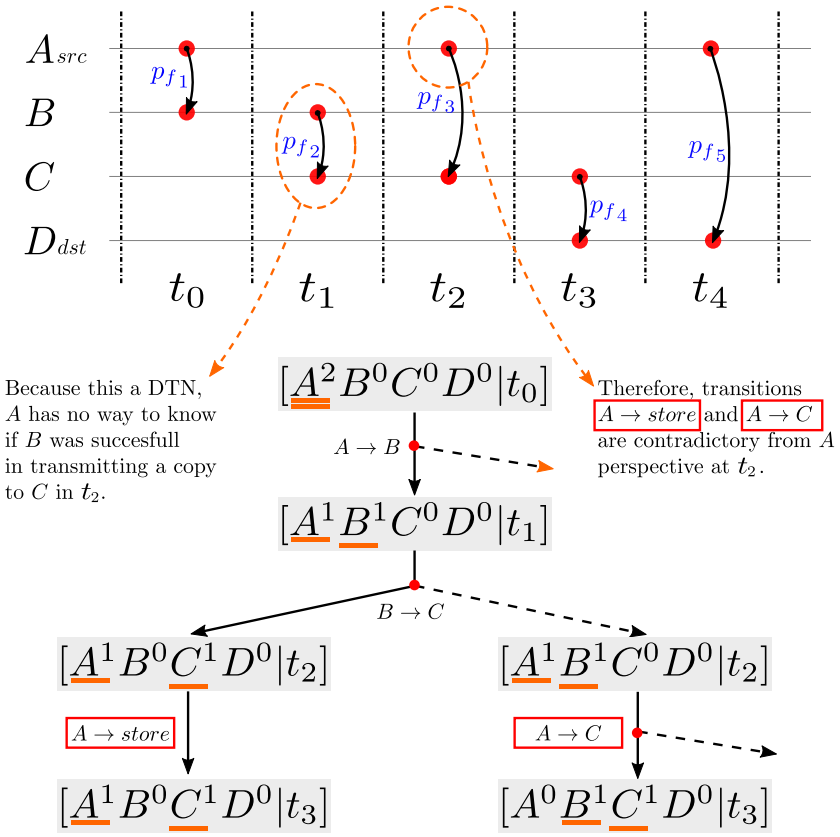


Figure 7: An example where local knowledge on A is not enough to determine the global status of the system.

copy through node C with a probability of success of 0.25 (the probability of success if delivering later directly to D is 0.1). However, if A had delivered a copy at time slot t_0 and preserved a second copy, the optimizing route for $LTr_A(t_0, 1, t_2)$ would be to keep the copy and deliver it later through D (with probability 0.4645, against 0.4525 if the second copy is delivered through C instead).

L-RUCoP considers this peculiarity to optimize the decisions. This means that populating the matrix requires N different executions of RUCoP. Since nodes in DTN networks may not have powerful on-board computers, a centralized node, such as the mission operation and control (MOC) center in the case of satellite networks, should be responsible for computing $LTr_n(t_s, c, t_i)$ and providing it to the network nodes in advance.

The construction of the L-RUCoP matrix is detailed in Algorithm 3. First, RUCoP is executed for all possible $c \leq N$ copies, storing the resulting states, transitions and delivery probabilities (S_c, Tr_c, Pr_c) (lines 1-2).

Notice that at this point all possible optimizing decisions have been calculated. So, what remains of the algorithm, is to construct all tables LTr_n by properly searching on the results calculated with RUCoP. Thus the algorithm nests two loops. The outer loop (lines 4-21) iterates on every node n, time slot ts, and number of copies $c \leq N$ in order to first calculate the "safe" decision $LTr_n(ts, c, ts)$. If needed, it then iterates on the inner loop (lines 10-19) to populate the table entries $LTr_n(ts, rc, ts')$ on the following time slots $ts' > ts$ for the distribution of the copies that have been held by the node.

So, the first step of the outer loop is to define the state s in which the node n has all copies c in time slot ts (line 5) and no other copy is in the network. Thus $Safe\_state(n, c, ts) = [A_0, B_0, ..., n_c, ...|ts]$. This is the "safe" state in which n has no knowledge of the network. If this state exists in S_c (i.e. the corresponding RUCoP found a likely successful route to the target node), node n has a route to target and its routing decisions (calculated through $Tr_c(s)$) are saved in $LTr_n(ts, c, ts)$ (line 7). At this point, the number of copies rc that are not distributed in this routing action is calculated (line 9) and the current time slot ts' is set to ts (line 8). If some copy remains in the node, the inner loop takes action (line 10). Firstly, the state s' known by node n after taking the last routing decision (namely, $LTr_n(ts, rc, ts')$) is calculated (line 11). More precisely $Post(LTr_n(ts, rc, ts'))$ delivers the state at time slot $ts' + 1$, in which node n contains the copies remaining after routing action $LTr_n(ts, rc, ts')$, any node n' that is in direct contact with n –according to

Algorithm 3: L-RUCoP Route table construction

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.

$LTr_n(ts, rc, ts')$– contains exactly the number of copies that n delivered to it, and any other node does not contain any copy. Also, the next time slot is calculated (line 12). If state s' exists in S_{rc} (i.e. the corresponding RUCoP found a likely successful route to the target node), the routing decision is saved (lines 14). Instead, if s' was not marked as explored by RUCoP, then no path to the successful state is possible from s', the action for that table entry is left undefined (line 16) and the inner loop is finished. While there is a successful route to the target node, the number of remaining copies rc for the next step are calculated (line 18) and the inner loop repeats until no further copies rc remains in n.

It is worth to recall that $LTr_n(ts, c, ts)$ is always the *safe entry* to look up for the local node. This means that whenever new copies arrive, or a routing decision fails to be accomplished in node n, it should take the current time slot ts as a safe place and look up the table at entry $LTr_n(ts, c, ts)$ (assuming c is the current number of copies held by n). Because of this fact of returning to the "safe entry" each time of uncertainty, in which the node assumes no copies are present in remote nodes, L-RUCoP accounts for a pessimistic-case knowledge from the local node perspective. Nevertheless, we show in Section 4 that L-RUCoP is a valuable routing approach for uncertain contact plan implementable in realistic DTN nodes constrained to localized knowledge.

3.4. RUCoP-enhanced CGR

To easily exploit the RUCoP method in existing DTN protocol stacks with minimal modifications, we also propose an alternative CGR formulation (a single-copy DTN routing scheme). We base the approach on a RUCoP-based SDP metric to achieve reliably delivery of bundles over an uncertain contact plan. CGR is a Dijkstra-based distributed routine that runs on each DTN node to determine the best routes to a given destination based on a pre-provisioned contact plan (the interested reader can refer to [18], [17] and [28] for an in-depth description of CGR). We propose CGR-UCoP as a simple means of extending CGR to operate with uncertain contact plans based on the outcomes of RUCoP. The idea is that CGR-UCoP selects the route that optimizes the successful delivery probability (SDP) instead of optimizing the time to destination as it is normally done in CGR.

In CGR-UCoP, we let CGR calculate the list of possible routes to a given destination using its modified Dijkstra contact plan search. In other words, route computation is left unchanged from legacy CGR. Also, the resulting

route list for each destination is constructed and consulted on forwarding time by the DTN node. However, instead of choosing the best route from the list based on the best delivery time metric, CGR-UCoP decides considering a custom SDP-based metric. CGR-UCoP metric is built around the Pr table constructed in Algorithm 1 for only 1 copy. More precisely, for each node n and time slot ts, we take $Pr_n(ts) = Pr(Safe\_state(n, 1, ts))$ ($Safe\_state$ is defined as in Sec. 3.3). That is $Pr_n(ts)$ is the probability of successfully delivering a single copy from node n at time ts. Similarly to L-RUCoP, the values of $Pr_n(t_s)$ can be pre-computed and provisioned to the DTN nodes together with the contact plan required by CGR to operate.

For the calculations, we assume that, after running CGR, a node n is left with a table $Rl_n : \mathcal{T} \to \wp(E^*)$ that, given a time slot ts, returns a set of partial routes $Rl_n(ts)$. Each $r \in Rl_n(ts)$ is a sequence of contacts –recall that each contact is an edge $e \in E$ of the uncertain timed-varying graph– representing a partial route to destination, more precisely, the fragment of the route that starts in node n at time slot ts and contains all hops that take place only during the same time slot. Thus, for instance, considering the graph of Fig. 2, $r = (S \to A)\,(A \to B)$ is a possible route in $Rl_S(t_0)$, but $(S \to A)\,(A \to B)\,(B \to D)$ is not, as it expands through two time slots (t_0 and t_1), nor is $(S \to A)$, since it does not contains all the hops in time slot t_0. We let $r[i]$ indicate the ith contact in the sequence and $|r|$ the length of r (in the example $r[0] = S \to A$ and $|r| = 2$). In addition, $src(e)$ and $tgt(e)$ indicate the source and target of contact e respectively.

Based on Pr, a SDP for a partial route $r \in Rl_n(ts)$ can be computed as follows.

$$
\begin{aligned}
SDP_{CGR}(r, ts) = \\
&\left(\prod_{k=0}^{|r|-1}(1 - p_f(r[k], ts)\right) \cdot Pr_{tgt(r[|r|-1])}(ts + \varsigma(r[|r| - 1], ts)) \\
+ \sum_{k=0}^{|r|-1} &\left(\prod_{i=0}^{k-1}(1 - p_f(r[i], ts))\right) \cdot p_f(r[k], ts) \\
&\cdot Pr_{src(r[k])}(ts + f_{dd}(r[k], ts))
\end{aligned}
$$

The first summand of the equation corresponds to the successful transmission of the message through all hops in r. This probability is estimated as the

product of the probability of successfully transmitting in each contact –the probability of success in the ith hop is $(1 - p_f(r[k]))$– times the likelihood (according to RUCoP) that the message is succesfully transmitted to destination from the last node of the partial route r (i.e. $Pr_{tgt(r[|r|-1])}(ts + \varsigma)$). Notice that this last probability should be considered at the moment that the message is available in the node, which can only be after the transmission delay $\varsigma(r[|r| - 1], ts)$. The second summand estimates the probability of successfully transmitting the message given that some hop in r failed to transmit at time slot ts. The kth summand here corresponds to the likelihood of successfully transmitting given that the hop k is the first to fail. This is calculated as the product of the probability of succsesfully transmitting in the first $k - 1$ hops (i.e. $\left(\prod_{i=0}^{k-1}(1 - p_f(r[i], ts))\right)$), times the probability of failing in the kth hop ($p_f(r[k], ts)$), times the likelihood (according to RUCoP) that the message is succesfully transmitted to destination from the node that failed to transmit in the kth hop (i.e. $Pr_{src(r[k])}(ts + f_{dd}(r[k], ts))$). Notice this last probability should be considered at the moment that such node detects that the communication has failed, i.e. at $ts + f_{dd}(r[k], ts)$.

The resulting metric SDP_{CGR} indicates the delivery probability of each route in $Rl_n(t_s)$ computed by CGR, which can be used to decide on a reliable proximate node to forward the bundle with a simple modification to existing implementations. It is worth noting that RUCoP might have explored more routes (potentially more reliable) than those in $Rl_n(t_s)$, the construction of which is guided by best delivery time as per CGR's internal Dijkstra searches. Nevertheless, in Section 4 we show that the RUCoP-based SDP metric outperforms baseline CGR and approximates the theoretical outcome of RUCoP and L-RUCoP in random and realistic application scenarios.

4. Result Analysis

In this section, we propose a benchmark ecosystem to evaluate the proposed routing schemes for DTNs under uncertain contact plans, and use it to analyze the network performance when applying RUCoP, L-RUCoP and CGR-UCoP.

4.1. Benchmark

A benchmark for DTNs under uncertain contact plans needs to comprise all possible routing solutions that can be considered for such scenarios.

In particular, CGR, sought for fully scheduled DTNs and S&W, sought for fully unpredictable DTNs sit at the edges of the uncertain DTNs classification. Other intermediate schemes present in the literature are also considered. Table 2 summarizes and compares the routing schemes present in the benchmark. We briefly recapitulate them as follows.

- Upper bound reference:
  - **CGR-FA**: CGR-FA is an oracle-based fault-aware (FA) scheme. It leverages the same single-copy implementation than CGR, but uses a contact plan where contacts that will fail are removed. By being able to know where and when faults will occur, CGR-FA is used as a theoretical upper bound providing the best achievable performance (delivery ratio and energy consumption).
- Single-copy, certain contact plan:
  - **CGR**: Current implementation of CGR [17] in ION v3.5.0 [47] which forwards a bundle using the first contact of the route which has the *best delivery time* among all to the given destination. CGR assumes all contacts in the contact plan will occur as planned.
- Single-copy, uncertain contact plan:
  - **CGR-HOP**: A variant of CGR which forwards a bundle on the first contact or hop of the route which has the *least hop count* among all to the given destination. As discussed in [48], reducing the hops increases the delivery probability in uncertain contact plans, at the expense of delivery delay.
  - **CGR-UCoP**: The RUCoP-enhanced CGR formulation presented in Section 3.4 that enables a straightforward implementation to leverage RUCoP model features in DTN nodes based on ION protocol stack.
- Multi-copy, uncertain contact plan:
  - **RUCoP**: Static routing rules are sent to each node in the network. These routes are computed using the RUCoP model in Algorithm 1. To determine the current state and decide on the subsequent action, nodes have access to a global view of the copy

distribution on the network, which is not necessarily feasible in reality. The benchmark considers RUCoP with 1, 2, 3 and 4 copies.

  - **L-RUCoP**: Static routing rules are sent to each node in the network by means of the *LTr* table. The table comprises a set of specific routing decisions, based on RUCoP model computed for each node, destination and number of copies. For each bundle, nodes decide routing based on the number of local copies. The benchmark considers L-RUCoP with 1, 2, 3 and 4 copies.
  - **CGR-2CP**: Another variant of CGR where two-copies (2CP) are generated at the source [48]. Copies are forwarded via both the best delivery time and the least hop count routes, when different. CGR-2CP provides equal or better delivery ratio than CGR-HOP with improved delivery delay.

- Multi-copy, no contact plan knowledge:
  - **S&W**: Spray-and-wait routing provides similar performance metrics than flooding with less overhead [24]. The traffic source spreads a limited number of copies to the first contacted neighbors and then wait until one of those copies reaches the destination. We evaluate S&W with 2, 3 and 4 copies.

For each routing scheme, the benchmark considers and evaluates the following routing metrics.

- **Delivery Ratio:** number of bundles successfully delivered over number of bundles generated, excluding copies. This is the main metric of the benchmark.
- **Delivery Delay:** mean delay per bundle successfully delivered to the destination. Non delivered bundles are not considered in the metric; thus, this metric should be considered after the delivery ratio.
- **Energy Efficiency:** number of bundles successfully delivered over the total number of transmissions in the network. Also observed after the delivery ratio, as good efficiency might come at the expense of poor delivery.

Routing Algorithms	Contact plan	Encoded probability	Encoded failures (oracle)	Implementable (local view)	Copies	Main optimization metric
CGR-FA	Yes	Yes	Yes	No	1	Delivery
RUCoP	Yes	Yes	No	No	1-4	Delivery
L-RUCoP	Yes	Yes	No	Yes	1-4	Delivery
CGR-UCoP	Yes	Yes	No	Yes	1	Delivery
CGR	Yes	No	No	Yes	1	Delay
CGR-HOP	Yes	No	No	Yes	1	Delivery
CGR-2CP	Yes	No	No	Yes	2	Delivery & Delay
S&W	No	No	No	Yes	2-4	Delivery & Delay

Table 2: Routing Schemes in the Benchmark

We analyze the results obtained from two benchmark scenarios: random networks and ring-road networks (RRN). The former renders a highly connected network with several route paths, while the latter comprises two realistic and simple topologies where satellites can contact ground spots (RRN-A and RRN-B). In all cases, bundles sizes are set small enough to avoid congestion biases. Also, channels are configured as error-free (i.e., no packet drop) in order to focus the analysis only on the uncertainty phenomena.

- **Random Networks:** Composed of 10 random topologies with 8 nodes and a duration of 100 seconds. Time is fragmented in episodes of 10 seconds. In each episode, the connectivity between nodes (i.e., presence of contacts) is decided based on a contact density parameter of 0.2, similar to [35]. An all-to-all traffic pattern is assumed. Each routing algorithm is simulated 100 times on each of the 10 networks and then averaged.

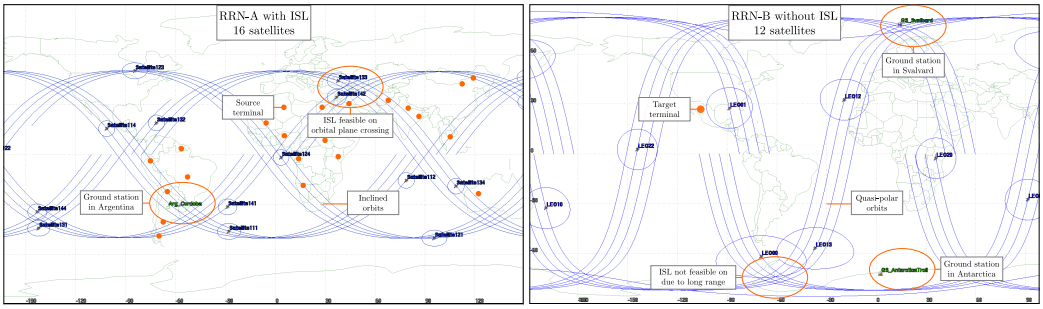
- **RRN-A with ISL:** The RRN-A is based on a realistic low-Earth orbit Walker constellation of 16 satellites proposed and described in [28]. Satellites act as data-mules by receiving data from 22 isolated ground

terminals, store it and deliver it to a ground station placed in Argentina. This is an all-to-one traffic pattern. In this case, satellites are equipped with Inter-Satellite Links (ISLs) implying contacts are also possible in-orbit [28]. Routes can thus involve multiple hops between satellites and ground terminals. The scenario is propagated for 24 hours and sliced into 1440 time slots, each of 60 seconds. Within a time slot, a contact is considered feasible if a communication opportunity of more than 30 seconds exists. This corresponds to a fine-grained model.

- **RRN-B without ISL:** A different Walker constellation topology of 12 satellites on polar orbits where no close-distance crossing is present. Not having ISL implies the routes to a target ground spot destination use at most one data-mule satellite. In this case, the routing decision is taken by a centralized mission control for data flowing from Internet to the isolated terminals. This is a one-to-one traffic pattern where routing implies deciding which ground station will be used to upload the data to which satellite. Two ground stations are configured as gateways in Antarctica and Svalbard. This scenario considers a coarse-grain model: time slots are defined in such a way that contacts start and terminate within the time slot duration.

It is worth mentioning that orbital paths[2] are calculated from STK [49] and encoded into contact plans with contact plan designer [50]. For the sake of simplicity, contact failure probabilities p_f are configured homogeneously in all links, ranging between [0,1]. Indeed, p_f s is the independent variable in the benchmark. As a result, it is expected that certain contact plan routing provide good metrics when $p_f \approx 1$, while non contact plan based solutions on $p_f \approx 0$. The hypothesis is that uncertain contact plan approaches outperform both in intermediate values of p_f. By running a large routing simulation campaign using DtnSim [51], we are able to determine on which ranges of p_f the hypothesis holds.

[2]STK scenarios, visualizations, orbital parameters and ground locations as well as resulting contact plans for the proposed benchmark are publicly available at `https://sites.google.com/unc.edu.ar/dtsn-scenarios`



RRN-A with ISL - 16 LEO satellites - 1 ground station (dst) - 22 ground terminals (src)					
Name	**T. Anomaly** [deg]	**Altitude** [km]	**Arg. Perigee** [deg]	**Inclination** [deg]	**RAAN** [deg]
Satellite111,12,13,14	0, 90, 180, 270	500	0	50	0
Satellite121,22,23,24	23, 113, 203, 293	500	0	50	90
Satellite131,32,33,34	45, 135, 225, 315	500	0	50	180
Satellite141,42,43,44	68, 158, 248, 338	500	0	50	270
Name	**Latitude** [deg]	**Longitude** [deg]	**Altitude** [km]		
Arg_Cordoba	-31.5242	-64.4636	0.724		

RRN-B without ISL - 12 LEO satellites - 2 ground stations (src) - 1 ground terminals (dst)					
Name	**T. Anomaly** [deg]	**Altitude** [km]	**Arg. Perigee** [deg]	**Inclination** [deg]	**RAAN** [deg]
LEO00, 01, 02, 03	0, 90, 180, 270	600	0	98	360
LEO10, 11, 12, 13	30, 120, 210, 300	600	0	98	240
LEO20, 21, 22, 23	60, 150, 240, 330	600	0	98	120
Name	**Latitude** [deg]	**Longitude** [deg]	**Altitude** [km]		
GS_AntarcticaTroll	-72.00194	2.53389	1.285103231		
GS_Svalbard	78.065041	14.214635	0.033170608		

Figure 8: RRN satellite constellation topologies, parameters and orbital tracks. On the left, RRN-A with ISL shows the 22 ground nodes (sources of data) as well as the target ground station in Argentina (many-to-one traffic). On the right, RRN-B without ISL shows the two ground station that can be used as gateways to reach a single target spot (one-to-one traffic).

4.2. Results

The benchmark results[3] are summarized in Fig. 9. To facilitate the comparison with state-of-the-art solutions, metrics are plotted with respect to CGR. CGR-FA is plotted as maximum theoretical bound in dotted lines. Because the RRN satellite networks offer simpler (and less) routes (i.e., less hop count) than the random networks, the potential improvement evidenced by CGR-FA in these scenarios is significant towards cases with higher failure probabilities (right hand-side of the curves).

4.2.1. Delivery Ratio

When contact failure probabilities are close to 0, the contact plan occurs as expected (i.e., no uncertainties). In this context, and for all studied scenarios, RUCoP, L-RUCoP and CGR-UCoP provide the same delivery ratio performance than CGR. Being based on CGR calculations, CGR-2CP and CGR-HOP also provide the same delivery ratio metric. On the other hand, S&W algorithms offer limited relative performance in these cases as they

[3]The RUCoP implementation in Python3 as well as the scripts used to obtain the results presented in this sections are publicly available at `https://bitbucket.org/fraverta/experiments-paper-ieee-tmc-2020`.

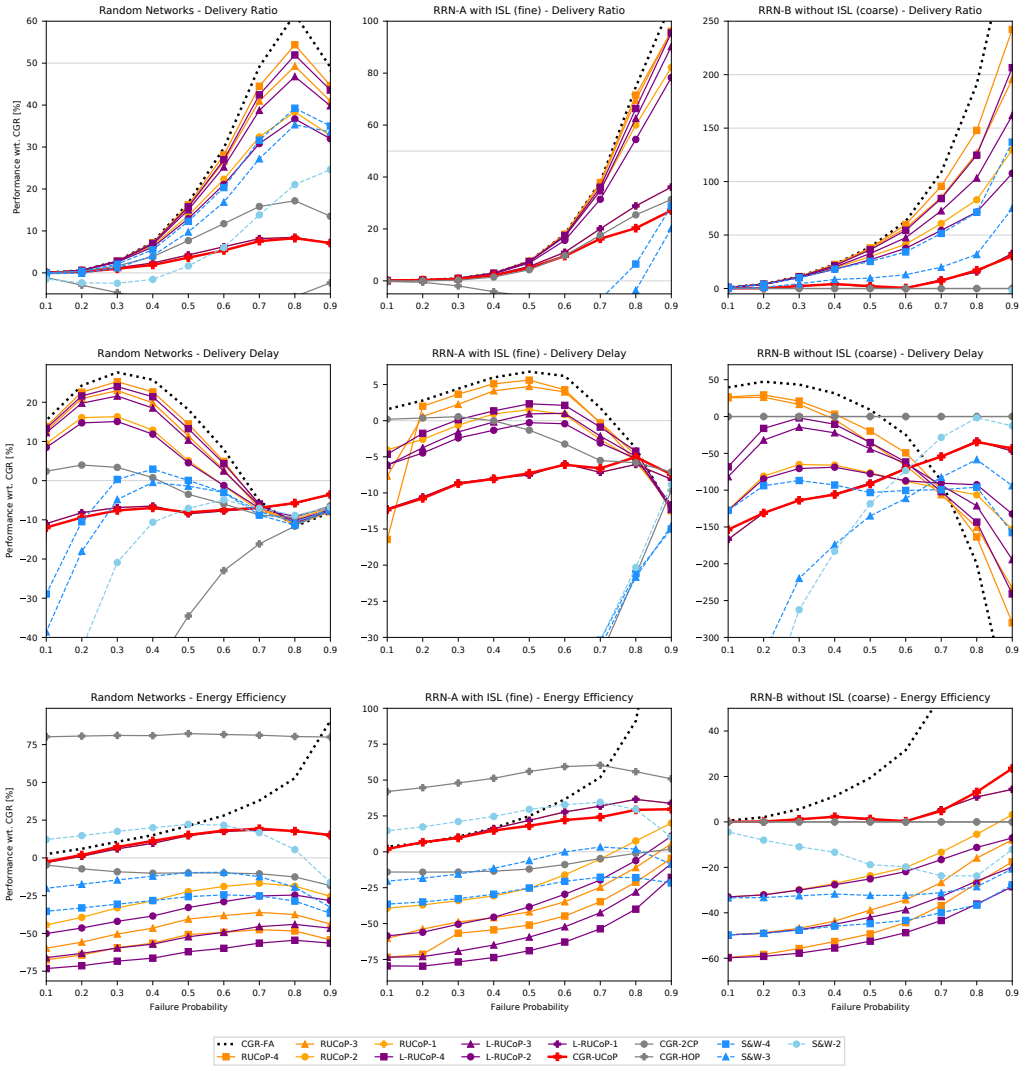


Figure 9: Routing for DTNs under uncertain contact plan benchmark. From left to right, the different scenarios: random networks, RRN-A, and RRN-B. From top to bottom, the different metrics: delivery ratio, delivery delay, energy efficiency. Delivery delay and energy efficiency have to be considered after delivery ratio, as they are computed from delivered bundles only. Curves includes CGR-FA (oracle), RUCoP (1 to 4 copies), L-RUCoP (1 to 4 copies), CGR-UCoP (adapted CGR), CGR-2CP (two-copies), CGR-HOP (lowest hop count metric), and S&W (2 to 4 copies).

have no consideration of the topological knowledge imprinted in the contact plan.

As the probability of failure increases, the delivery ratio diverges for most techniques. In all scenarios, and for each number of copies, RUCoP model provides the best delivery ratio results, improving as the number of allowed copies increases. This improvement becomes more evident for larger p_f. L-RUCoP follows RUCoP closely, with a delta of performance explained by the fact of solely relaying on (a pessimistic) local node's knowledge. Also, as expected, S&W improves the delivery ratio on scenarios with higher uncertainty. Depending on the number of copies, S&W schemes can even outperform CGR baseline in particular cases, as already indicated in [35]. In random networks, S&W provides good two-copies results, in comparison with CGR-2CP; however, the latter behaves better in simpler networks such as RRN (delivery ratio for S&W-2 in RRN-A and B is always worst than CGR baseline and thus not plotted). Nevertheless, L-RUCoP offers the best single-copy implementable routing solution, closely followed by CGR-UCoP, both improving CGR delivery ratio in cases with medium and high failure probabilities. Moreover, in practical RRN scenarios, CGR-UCoP also provides better performance than S&W with two copies, and even better than S&W-3 in RRN with ISL. Indeed, L-RUCoP with one copy provides the same outcomes than RUCoP-1, and remarkably, CGR-UCoP (also single-copy) almost always delivers the same performance than both (notice cross markers of RUCoP and L-RUCoP are behind CGR-UCoP in most of the plots). This is compelling evidence that the practical applicability of CGR-UCoP can provide great value at minimum implementation costs. In particular, under high uncertainty, CGR-UCoP outperforms CGR by 9% in random networks, 22% in RRN-A with ISL and 25% in RRN-B without ISL.

4.2.2. Delivery Delay

Although not specifically optimized for delivery delay, RUCoP and L-RUCoP models exhibit a reasonable performance with respect to CGR in this metric, especially in random networks. This can be explained by the fact that RUCoP-based models consider all possible paths and can determine the optimal one, which is not always the case of CGR as already discussed in [52]. As p_f increases, the delivery delay of RUCoP decreases with respect to CGR, but with a much larger deliver ratio. That is, the few bundles that arrive with CGR do so in a shorter time on routes whose contacts do not present failures, while RUCoP is able to deal with failures and deliver a greater

number of bundles, some of which take longer to arrive thereby increasing the average delay value. On the implementable side, CGR-UCoP delivery delay performance approaches CGR as the failure probabilities increases. In realistic RRN scenarios, CGR-UCoP is consistently better than S&W routing as well as CGR-HOP which honors low hops and potentially higher latency routes (delivery delay for CGR-HOP is the lowest of all schemes not reaching the scale of RRN-A and B plots). Notably, CGR-2CP offers very similar performances than plain CGR as one of the two copies follows the same lowest delivery delay route than CGR.

4.2.3. Energy efficiency

On the energy efficiency side, we care about the transmission effort required to deliver the bundles. Naturally, single copies schemes offer the least effort, especially CGR-HOP which also minimises the overall hops and thus, transmissions. On the other hand, multiple copy solutions including RUCoP-4, L-RUCoP-4 and S&W-4 demand the largest energy effort, being the latter consistently better, at the expense of a lower delivery ratio. Remarkably, and being a single copy scheme, CGR-UCoP always offer the same or better energy efficiency than CGR, and is only outperformed by the less performing CGR-HOP and by S&W-2 in some cases.

To wrap up, RUCoP model proved to approach the ideal fault-aware case of CGR-FA by leveraging the presented MDP formulation, especially with larger number of copies. While RUCoP model can serve as a routing solution with global view, L-RUCoP obtains similar results based on a reduced local view in practical DTNs, and implemented in existing protocol stacks by means of CGR-UCoP. Indeed, CGR-UCoP has shown that the consideration of the adapted SDP calculation of RUCoP enables a very appealing performance over the whole failure probability range in DTNs under uncertain contact plan.

4.3. Discussion

To properly frame the benefits and applicability of RUCoP and L-RUCoP models and CGR-UCoP algorithm, we discuss some considerations.

Multiple Senders: Although RUCoP model, as presented in Section 2, takes one sender and one destination as arguments, multiple senders can be considered in a single MDP if they seek to reach the same destination. Indeed, this was already accounted for in the RRN-A case (all-to-one traffic

shape), where the same RUCoP was solved for each of the 22 senders. Indeed, a policy was derived for each data flow from a single execution of the MDP. This can be achieved because the MDP tree for each case is exactly the same except the initial state at $\mathcal{T}_0$. In general, this approach can be generalized as long as different data flows do not compete for a same limited channel resource (i.e., congestion).

Congestion: In general, congestion is an open research issue in DTN [53]. In this context, RUCoP-based models have been sought for and evaluated in scenarios where congestion is not present. This means that when a route is determined for a bundle, it is assumed that there will be enough capacity to allocate such data transmission (i.e., sizes of the bundles is by far smaller than the contact capacity). While this can be the case for unsaturated networks, congested networks would need to rely on simulations analysis that validates if the RUCoP routing assumptions holds.

Scalability: Table 4.3 summarizes the scalability metrics of the evaluated scenarios when using RUCoP. In particular, the execution time on an Intel i7 processor with 16 GB of RAM running an Ubuntu 19.10 was measured for a Python3 implementation of the RUCoP routine. The explored states and evaluated transitions were listed to observe their increment with larger scenarios and required copies. Results show that RUCoP is well suited to solve realistic cases in reasonable time. Indeed, less than an hour is required for the more complex case of RRN-A with ISL and four copies of the data. As already explained, a coarse model of the network offers significant gain in processing time, at the expense of less accurate results.

Compared with the computation time required by RUCoP, calculating L-RUCoP routing matrix demands a reduced overhead. The specific processing time for each of the case studies is reported in the L-RUCoP Time[sec] column, in Table 4.3. In particular, the time required for computing L-RUCoP-2 for the RRN-A scenario is the sum of those for RUCoP-1 and RUCoP-2 (i.e., $258 + 291 = 549$ seconds), plus the cost of building the L-RUCoP routing matrix (37.49 seconds), adding up for a total of 586.49 seconds. As RUCoP computation can be done in parallel, the time can be significantly reduced.

5. Conclusion

Delay Tolerant Networks (DTNs) classification has biased the research of routing algorithms to fit either fully scheduled or dynamically-learned

Table 3: Scalability Metrics

Copies	1	2	3	4
Random Networks				
Time [sec]	2	6	107	2416
States	74	318	1056	2915
Transitions	391	9491	179797	2804864
L-RUCoP Time[sec]	+0.15	+0.42	+0.85	+1.51
RRN-A with ISL (fine grain)				
Time [sec]	258	291	657	3290
States	6091	76428	646152	4126765
Transitions	6973	99742	969861	7147805
L-RUCoP Time[sec]	+12.92	+37.49	+107.19	+426.96
RRN-B without ISL (coarse grain)				
Time [sec]	18	21	38	134
States	898	8568	49774	220745
Transitions	1020	11133	73566	369689
L-RUCoP Time [sec]	+1.75	+4.38	+9.02	+21.16

probabilistic use cases. In this paper, we have uncovered that routing under uncertain contact planning deserves a different classification. Uncertain DTNs have not only applicable relevance but also can serve as a more generic routing approach for many practical DTNs.

A first Markov Decision Process coined RUCoP was introduced for arbitrary number of copies in uncertain DTNs. RUCoP provides a theoretical upper bound for the data delivery ratio when a global vision of the system is possible. RUCoP enabled the derivation of L-RUCoP when knowledge is restricted to a local view, and single-copy CGR-UCoP where the outcomes of the MDP model can drive routing decisions of the popular CGR routing algorithm.

To evaluate RUCoP, L-RUCoP and CGR-UCoP, we have proposed an appealing benchmark comprising random and realistic case studies as well as candidate routing solutions. Results showed that RUCoP and L-RUCoP models approach the ideal case as the number of copies increases. On the other hand, single-copy CGR-UCoP has also provided outstanding results under uncertain contact plans, outperforming both CGR (scheduled routing) by up to 25% in realistic satellite DTNs with uncertain links.

Future work involves the comparison with the simulation results reported in [40] as well as further research on multi-objective optimizations comprising delivery delay and route reliability for CGR-UCoP, which will be implemented and proposed for NASA's ION protocol stack. Succeeding in such endeavor would settle CGR-UCoP as the de-facto routing scheme for DTNs with uncertain contact plans.

Acknowledgement

This research has received support from the ERC Advanced Grant 695614 (POWVER), the DFG grant 389792660, as part of TRR 248 (https://perspicuous-computing.science), the Agencia I+D+i grant PICT-2017-3894 (RAFTSys), PICT-2017-1335, and the SeCyT-UNC grant 33620180100354CB (ARES). Part of this work has been developed while Dr. Juan Fraire was visiting Politecnico di Torino.

References

[1] K. Fall, A delay-tolerant network architecture for challenged Internets, in: Proceedings of the 2003 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications, SIGCOMM '03, ACM, New York, NY, USA, 2003, pp. 27–34. doi:10.1145/863955.863960.
URL http://doi.acm.org/10.1145/863955.863960

[2] V. Cerf, S. Burleigh, A. Hooke, L. Torgerson, R. Durst, K. Scott, K. Fall, H. Weiss, Delay-tolerant networking architecture, RFC 4838, RFC Editor (April 2007).
URL http://www.rfc-editor.org/rfc/rfc4838.txt

[3] S. Burleigh, A. Hooke, L. Torgerson, K. Fall, V. Cerf, B. Durst, K. Scott, H. Weiss, Delay-tolerant networking: An approach to interplanetary internet, Comm. Mag. 41 (6) (2003) 128–136. doi:10.1109/MCOM.2003.1204759.
URL http://dx.doi.org/10.1109/MCOM.2003.1204759

[4] C. Caini, H. Cruickshank, S. Farrell, M. Marchese, Delay- and disruption-tolerant networking (DTN): An alternative solution for future satellite networking applications, Proceedings of the IEEE 99 (11) (2011) 1980–1997. doi:10.1109/JPROC.2011.2158378.

[5] L. Gupta, R. Jain, G. Vaszkun, Survey of important issues in UAV communication networks, IEEE Communications Surveys & Tutorials 18 (2) (2015) 1123–1152.

[6] N. Benamar, K. D. Singh, M. Benamar, D. E. Ouadghiri, J.-M. Bonnin, Routing protocols in vehicular delay tolerant networks: A comprehensive survey, Computer Communications 48 (2014) 141 – 158, opportunistic networks. doi:https://doi.org/10.1016/j.comcom.2014.03.024.
URL http://www.sciencedirect.com/science/article/pii/S0140366414001212

[7] J. Hom, L. Good, Shuhui Yang, A survey of social-based routing protocols in delay tolerant networks, in: 2017 International Conference on Computing, Networking and Communications (ICNC), 2017, pp. 788–792. doi:10.1109/ICCNC.2017.7876231.

[8] F. Z. Benhamida, A. Bouabdellah, Y. Challal, Using delay tolerant network for the Internet of Things: Opportunities and challenges, in: 2017 8th International Conference on Information and Communication Systems (ICICS), 2017, pp. 252–257. doi:10.1109/IACS.2017.7921980.

[9] J. Partan, J. Kurose, B. N. Levine, A survey of practical issues in underwater networks, SIGMOBILE Mob. Comput. Commun. Rev. 11 (4) (2007) 23–33. doi:10.1145/1347364.1347372.
URL http://doi.acm.org/10.1145/1347364.1347372

[10] K. Scott, S. Burleigh, Bundle protocol specification, RFC 5050, RFC Editor (November 2007).
URL http://www.rfc-editor.org/rfc/rfc5050.txt

[11] W.-B. Pöttner, J. Morgenroth, S. Schildt, L. Wolf, Performance comparison of DTN bundle protocol implementations, in: Proceedings of the 6th ACM workshop on Challenged networks, ACM, 2011, pp. 61–64.

[12] J. A. Fraire, J. M. Finochietto, Design challenges in contact plans for disruption-tolerant satellite networks, Communications Magazine, IEEE 53 (5) (2015) 163–169. doi:10.1109/MCOM.2015.7105656.

[13] J. A. Fraire, P. G. Madoery, J. M. Finochietto, Traffic-aware contact plan design for disruption-tolerant space sensor networks, Ad Hoc Networks 47 (2016) 41 – 52. doi:http://dx.doi.org/10.1016/j.adhoc.2016.04.007.

[14] J. A. Fraire, J. Finochietto, Routing-aware fair contact plan design for predictable delay tolerant networks, Ad Hoc Networks 25 (2015) 303 – 313. doi:https://doi.org/10.1016/j.adhoc.2014.07.006.

[15] J. A. Fraire, P. G. Madoery, J. M. Finochietto, On the design and analysis of fair contact plans in predictable delay-tolerant networks, Sensors Journal, IEEE 14 (11) (2014) 3874–3882. doi:10.1109/JSEN.2014.2348917.

[16] M. Carosino, J. A. Fraire, J. A. Ritcey, Integrating scheduled DTNs and TDMA-based MAC sublayers: Preliminary results, in: 2018 6th IEEE International Conference on Wireless for Space and Extreme Environments (WiSEE), IEEE, 2018, pp. 141–146.

[17] J. A. Fraire, O. De Jonckère, S. C. Burleigh, Routing in the space internet: A contact graph routing tutorial, Journal of Network and Computer Applications 174 (2021) 102884. doi:https://doi.org/10.1016/j.jnca.2020.102884.
URL http://www.sciencedirect.com/science/article/pii/S1084804520303489

[18] G. Araniti, N. Bezirgiannidis, E. Birrane, I. Bisio, S. Burleigh, C. Caini, M. Feldmann, M. Marchese, J. Segui, K. Suzuki, Contact graph routing in DTN space networks: overview, enhancements and performance, IEEE Comms. Magazine 53 (3) (2015) 38–46. doi:10.1109/MCOM.2015.7060480.

[19] S. Grasic, E. Davies, A. Lindgren, A. Doria, The evolution of a DTN routing protocol-PRoPHETv2, in: Proceedings of the 6th ACM workshop on Challenged networks, 2011, pp. 27–30.

[20] J. Burgess, B. Gallagher, D. D. Jensen, B. N. Levine, et al., MaxProp: Routing for vehicle-based disruption-tolerant networks., in: Infocom, Vol. 6, Barcelona, Spain, 2006.

[21] S. Jain, K. Fall, R. Patra, Routing in a delay tolerant network, Vol. 34, ACM, 2004.

[22] M. Feldmann, F. Walter, Routing in ring road networks with limited topological knowledge, in: 2017 Int. Conf. on Wireless for Space and Extreme Environments (WiSEE), 2017, pp. 63–68.

[23] A. Vahdat, D. Becker, Epidemic routing for partially-connected ad hoc networks, Tech. rep., Duke University, Department of Computer Science (2000).

[24] T. Spyropoulos, K. Psounis, C. S. Raghavendra, Spray and wait: An efficient routing scheme for intermittently connected mobile networks, in: 2005 ACM SIGCOMM Workshop on DTN, 2005, pp. 252–259.

[25] T. Spyropoulos, K. Psounis, C. S. Raghavendra, Spray and focus: Efficient mobility-assisted routing for heterogeneous and correlated mobility, in: Fifth Annual IEEE International Conference on Pervasive Computing and Communications Workshops (PerComW'07), IEEE, 2007, pp. 79–85.

[26] K. Sakai, M.-T. Sun, W.-S. Ku, Data-intensive routing in delay-tolerant networks, in: IEEE INFOCOM 2019 - IEEE Conference on Computer Communications, 2019, pp. 2440–2448. doi:10.1109/INFOCOM.2019.8737620.

[27] S. Burleigh, C. Caini, J. Messina, M. Rodolfi, Toward a unified routing framework for DTN, in: 2016 IEEE Int. Conf. on Wireless for Space and Extreme Environments (WiSEE), 2016, pp. 82–86.

[28] J. A. Fraire, P. Madoery, S. Burleigh, M. Feldmann, J. Finochietto, A. Charif, N. Zergainoh, R. Velazco, Assessing contact graph routing performance and reliability in distributed satellite constellations, Hindawi Journal of Computer Networks and CommunicationseVol. 2017, Article ID 2830542, 18 pages (2017). doi:10.1155/2017/2830542.

[29] R. Kalaputapu, M. J. Demetsky, Modeling schedule deviations of buses using automatic vehicle-location data and artificial neural networks, Transportation Research Record (1995) 44–52.

[30] A. Sahai, R. Tandra, S. M. Mishra, N. Hoven, Fundamental design tradeoffs in cognitive radio systems, in: Proceedings of the first international workshop on Technology and policy for accessing spectrum, ACM, 2006, p. 2.

[31] C. Hwang, F. A. Tillman, M. Lee, System-reliability evaluation techniques for complex/large systems: A review, IEEE Transactions on Reliability 30 (5) (1981) 416–423.

[32] Q. Liang, E. Modiano, Survivability in time-varying networks, IEEE Transactions on Mobile Computing 16 (9) (2017) 2668–2681. doi:10.1109/TMC.2016.2636152.

[33] F. Li, S. Chen, M. Huang, Z. Yin, C. Zhang, Y. Wang, Reliable topology design in time-evolving delay-tolerant networks with unreliable links, IEEE Transactions on Mobile Computing 14 (6) (2015) 1301–1314. doi:10.1109/TMC.2014.2345392.

[34] P. Madoery, F. Raverta, J. Fraire, J. Finochietto, On the performance analysis of disruption tolerant satellite networks under uncertainties, in: Proceedings of the 2017 XVII RPIC Workshop, 2017.

[35] P. G. Madoery, F. D. Raverta, J. A. Fraire, J. M. Finochietto, Routing in space delay tolerant networks under uncertain contact plans, in: 2018 IEEE International Conference on Communications (ICC), 2018, pp. 1–6. doi:10.1109/ICC.2018.8422917.

[36] F. D. Raverta, R. Demasi, P. G. Madoery, J. A. Fraire, J. M. Finochietto, P. R. D'Argenio, A Markov decision process for routing in space DTNs with uncertain contact plans, in: 2018 6th IEEE International Conference on Wireless for Space and Extreme Environments (WiSEE), 2018, pp. 189–194. doi:10.1109/WiSEE.2018.8637330.

[37] A. Bianco, L. de Alfaro, Model checking of probabalistic and nondeterministic systems, in: P. S. Thiagarajan (Ed.), Foundations of Software Technology and Theoretical Computer Science, 15th Conference, Bangalore, India, December 18-20, 1995, Proceedings, Vol. 1026 of LNCS, Springer, 1995, pp. 499–513. doi:10.1007/3-540-60692-0_70.
URL https://doi.org/10.1007/3-540-60692-0_70

[38] C. Baier, J. Katoen, Principles of model checking, MIT Press, 2008.

[39] C. Baier, L. de Alfaro, V. Forejt, M. Kwiatkowska, Model checking probabilistic systems, in: E. M. Clarke, T. A. Henzinger, H. Veith, R. Bloem (Eds.), Handbook of Model Checking, Springer, 2018, pp. 963–999. doi:10.1007/978-3-319-10575-8_28.
URL https://doi.org/10.1007/978-3-319-10575-8_28

[40] P. R. D'Argenio, J. A. Fraire, A. Hartmanns, Sampling distributed schedulers for resilient space communication, in: NASA Formal Methods Symposium, Springer, 2020, pp. 291–310.

[41] M. L. Puterman, Markov Decision Processes: Discrete Stochastic Dynamic Programming, 1st Edition, John Wiley & Sons, Inc., New York, NY, USA, 1994.

[42] J. Filar, K. Vrieze, Competitive Markov Decision Processes, Springer-Verlag, Berlin, Heidelberg, 1996.

[43] M. Kwiatkowska, G. Norman, D. Parker, PRISM 4.0: Verification of probabilistic real-time systems, in: G. Gopalakrishnan, S. Qadeer (Eds.), Proc. 23rd International Conference on Computer Aided Verification (CAV'11), Vol. 6806 of LNCS, Springer, 2011, pp. 585–591.

[44] S. R. Eddy, Hidden Markov models, Current opinion in structural biology 6 (3) (1996) 361–365.

[45] L. Cheung, N. A. Lynch, R. Segala, F. W. Vaandrager, Switched PIOA: parallel composition via distributed scheduling, Theor. Comput. Sci. 365 (1-2) (2006) 83–108. doi:10.1016/j.tcs.2006.07.033.
URL https://doi.org/10.1016/j.tcs.2006.07.033

[46] S. Giro, P. R. D'Argenio, L. M. F. Fioriti, Distributed probabilistic input/output automata: Expressiveness, (un)decidability and algorithms, Theor. Comput. Sci. 538 (2014) 84–102. doi:10.1016/j.tcs.2013.07.017.
URL https://doi.org/10.1016/j.tcs.2013.07.017

[47] S. Burleigh, Interplanetary overlay network: An implementation of the DTN bundle protocol, in: 2007 4th IEEE Consumer Communications and Networking Conference, 2007, pp. 222–226. doi:10.1109/CCNC.2007.51.

[48] P. G. Madoery, J. A. Fraire, J. M. Finochietto, Congestion management techniques for disruption-tolerant satellite networks, International Journal of Satellite Communications and Networking (2018) n/a–n/aSat.1210. doi:10.1002/sat.1210.
URL http://dx.doi.org/10.1002/sat.1210

[49] AGI Systems Tool Kit (STK), http://www.agi.com/STK.

[50] J. A. Fraire, Introducing contact plan designer: A planning tool for DTN-based space-terrestrial networks, in: 2017 6th International Conference on Space Mission Challenges for Information Technology (SMC-IT), 2017, pp. 124–127. doi:10.1109/SMC-IT.2017.28.

[51] J. A. Fraire, P. G. Madoery, F. Raverta, J. M. Finochietto, R. Velazco, DtnSim: Bridging the gap between simulation and implementation of space-terrestrial DTNs, in: Space Mission Challenges for Information Technology (SMC-IT), 2017 IEEE Int. Conference on, 2017.

[52] J. A. Fraire, P. G. Madoery, A. Charif, J. M. Finochietto, On route table computation strategies in delay-tolerant satellite networks, Ad Hoc Networks 80 (2018) 31 – 40. doi:https://doi.org/10.1016/j.adhoc.2018.07.002.
URL http://www.sciencedirect.com/science/article/pii/S1570870518304359

[53] A. P. Silva, S. Burleigh, C. M. Hirata, K. Obraczka, A survey on congestion control for delay and disruption tolerant networks, Ad Hoc Networks 25 (2015) 480–494.