NWCRG
Internet-Draft
Intended status: Experimental
Expires: March 12, 2020

J. Heide
Steinwurf Aps
S. Shi
K. Fouli
M. Medard
Code On Network Coding LLC
V. Chook
Inmarsat PLC

September 09, 2019

Random Linear Network Coding (RLNC)-Based Symbol Representation
draft-heide-nwcrg-rlnc-03

Abstract

This document describes a symbol representation for Random Linear Network Coding (RLNC) schemes used for reliable data transfer. Specifically, the following features are discussed and incorporated: both block RLNC and a sliding window RLNC, varying data frame sizes, and one or multiple symbols associated with a single symbol representation header.

Requirements Language

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at https://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on March 12, 2020.

Copyright Notice

Copyright (c) 2019 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (https://trustee.ietf.org/license-info) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

Table of Contents

1. Introduction 2
2. Symbol Representation 3
2.1. Representation Setup 4
2.2. Field Types and Formats 5
2.3. Externally Specified Parameters Required 7
2.4. Small Encoding Window 7
2.4.1. Examples 9
2.5. Large Encoding Window 10
3. Security Considerations 11
4. IANA Considerations 11
5. References 11
5.1. Normative References 11
5.2. Informative References 12
Authors' Addresses 12

1. Introduction

Symbol representation specifies the format of the symbol-carrying data unit that is to be used in network coding operations, including header format and symbol concatenation. This document describes a symbol representation format intended to be used for Network Coding in general, and for Random Linear Network Coding (RLNC) in particular [HK03].

Owing to its dynamic structure, network coding has requirements that are distinct from conventional point-to-point codes, leading to a highly reconfigurable symbol set. Consequently, the design choices related to symbol representation are particularly important in network coding as they have a direct impact on the viability of network protocols, topologies, and architecture [RLNC-Background]. For example, recoding [RLNC-Background] requires the coefficients to

be accessible at the recoding nodes. Hence, architectures and protocols requiring recoding must specify coefficient location in their symbol representation.

In addition to providing background on RLNC, [RLNC-Background] argues that careful design and specification of a symbol representation is a requirement for any viable network coding protocol, architecture, or topology.

2. Symbol Representation

This section provides a symbol representation design for implementing RLNC-based erasure correction schemes. In the decribed symbol representation design, multiple symbols are concatenated and associated with a single symbol representation header.

The symbol representation design is provided for constructing a data payload portion of a data packet for a protocol that utilizes a generation-based or sliding-window RLNC, where recoding can be used at intermediate nodes. A data packet data payload comprises one or more symbol representations. Each symbol representation in turn comprises one or more symbols that can be systematic, coded or recoded. The use of this symbol representation design is not limited by transmission schemes. It can be applied to unicast, multiple-unicast, multicast, multi-path, and multi-source settings and the like.

Coding coefficient vectors must be implicitly or explicitly transmitted from the sender to the receiver, generally along with the coded data for successful decoding of the original data. One option is to attach each coding coefficient vector to the corresponding coded symbol as a header, thus also enabling recoding at intermediate nodes. Another option is to attach the current state of a pseudo-random generator for generating the coding coefficient vector, to reduce the size of the header. Adding a header to each symbol may result in a high overhead when the symbol size is small or when generation or sliding window size is large. Adding a joint header to the beginning of each generation may also cause synchronization to be re-initiated only at the beginning of each generation instead of every symbol. In what follows, a symbol representation is provided that allow for both of these options such that both a general representation with coding coefficients and a compact representation with a seed for generating the coding coefficients can be used, in order to reduce the header overhead.

2.1. Representation Setup

This section specifies a symbol representation that enables both a general form with coding coefficient vectors attached, and a compact form where a seed is attached which is used to generate one or multiple coding coefficient vectors. Different maximum GENERATION and WINDOW SIZE are supported for RLNC encoding, recoding, and decoding.

To encode over a set of data symbols, a coding coefficient vector is first generated, comprising a number of finite field elements as specified by a GENERATION SIZE or WINDOW SIZE variable. For a generation based code the GENERATION SIZE defines the number of original symbols in each generation. For a window based code the WINDOW SIZE specifies the maximal number of symbols in the window over which coding can be performed. In the case of systematic codes, systematic symbols correspond to unit coding coefficient vectors.

Figure 1 illustrates the general symbol representation design. Four header fields precede the symbol data: TYPE flag (T), SYMBOLS, ENCODER RANK, and SEED or CODING COEFFICIENTS. The TYPE Flag (T) indicates if the symbol is systematic, coded, or recoded. SYMBOLS indicates the number of symbols in the SYMBOLS(S) DATA field. ENCODER RANK represents the current rank of the encoder, which is the number of symbols being linearly combined. SEED is used to generate the coding coefficient vector(s) using a pseudo-random number generator, for a compact form of the symbol representation. The CODING COEFFICIENTS field is a list of SYMBOLS number of coding coefficient vectors used to generate the SYMBOL(S) DATA, and used in the case where no random number generator is available or practical

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| T |SYMBOLS|   ENCODER RANK    |  SEED or CODING COEFFICIENTS  |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
|                        SYMBOL(S) DATA                         |
|                              ...                              |
|                                                               |
+---------------------------------------------------------------+
```

Figure 1: A general symbol representation design.

2.2. Field Types and Formats

The TYPE Flag (T) indicates if the symbol is systematic, coded, or recoded, and has the following properties:

- o 2 bits long.

- o If the TYPE flag is '1', all symbols included in this symbol representation are systematic or uncoded, with symbol index starting from ENCODER RANK. This option allows for efficient representation of systematic symbols.

- o If the TYPE is '2', all symbols included in this symbol representation are coded, with coding coefficient vectors generated using the included SEED and the ENCODER RANK. Consequently, only the first ENCODER RANK elements in the coding coefficient vector can be non-zero, whereas the remaing elements (e.g. GENERATION SIZE - ENCODER RANK) in the coding coefficient vector are zeros. This option allows for compact and efficient representation of coded symbols, which may also subsequently be recoded.

- o If the TYPE is '3', all symbols included in this symbol representation are either uncoded, coded or recoded. Each coding vector included is composed of GENERATION SIZE or WINDOW SIZE coefficients.

SYMBOLS indicates the number of symbols in the 'Symbol(s) Data' field, and has the following properties:

- o 4 bits long. A maximum number of 15 symbols are concatenated within each symbol representation.

- o The special case of SYMBOLS = 0 indicates that zero symbols are included, and consequently the size of SYMBOLS(S) DATA is 0 bytes. This can, for example, be used to implement a flush functionality or ensure that protocol operations do not stop in certain case for purely event-driven protocols.

ENCODER RANK represents the current rank of the encoder, that is, the number of original symbols used to compute the coded symbols(s). It has the following properties:

- o MUST be no larger than GENERATION/WINDOW SIZE.

- o If TYPE flag is '1', ENCODER RANK is the symbol index of the first data symbol in this symbol representation.

- o If TYPE flag is '2' or '3', ENCODER RANK is the number of data symbols over which coding was performed for all coded symbols in this symbol representation.

SEED is used to generate the coding coefficient vector(s) using a pseudo-random number generator, for a compact form of the symbol representation, and has the following properties:

- o The SEED field is only present when TYPE flag is '2'. If TYPE is '1' or '3', this field is absent.

- o The pseudo-random generator MUST be seeded with this value and all coding coefficient vectors are produced by the same generator. For example, if ENCODER RANK is 12, then the coding coefficient vector for the first symbol in this symbol representation is coefficients 0 through 11 generated by the pseudo-random generator seeded by SEED, and coding coefficient vector for the second symbol in this symbol representation is coefficients 12 through 23 generated by the pseudo-random generator seeded by SEED. If GENERATION/WINDOW SIZE is larger than ENCODER RANK, the remaining coefficients in the coding coefficient vector are zero.

- o To ensure that SEED can be interpreted correctly at the receiver, the same pseudo-random number generator MUST be used by the sender and a recoding or receiving node. Otherwise, more than one SEED field would need to be used.

- o 8 bits long. Thus, 256 different seed values can be served. One SEED is used per symbol representation, each of which can contain up to 15 symbols, all derived using the same SEED. For distinct ENCODER RANKs, different coding coefficient vectors would be generated from the same SEED, since only an ENCODER RANK number of coefficients from the random generator is grouped as a coding coefficient vector, before progressing to the next coding coefficient vector for the next symbol in the symbol representation. Consequently, the maximal number of coded symbols that can be generated for a generation is |SEED| * |SYMBOLS| * |ENCODER RANK| which in the best case is (2^8)*(2^4-1)*(2^10) ~ 2^22, which for all practical considerations can be considered as an infinite number of coded symbols. If all coded symbols that can be represented using a SEED is exhausted, symbols where the coding coefficient vectors is included can be sent instead.

CODING COEFFICIENTS field is a list of SYMBOLS number of coding vectors used to generate the ensuing SYMBOL(S) DATA, and has the following properties:

- o The CODING COEFFICIENT field is only present when TYPE flag is '3'. If TYPE is '1' or '2', this field is absent.

- o Each coding coefficient vector includes ENCODER RANK number of coding coefficients.

2.3. Externally Specified Parameters Required

This section specifies parameters that are REQUIRED for the use of this symbol representation but which are not included in the symbol representation and therefore MUST be communicated by means of some outer mecanism. Typically these parameters will be static throughout a protocol session. Consequently, there is little to gain by incorporating these parameters into the representation but conversely it would add additional overhead.

- o Field polynomial, the underlying field over which coding is performed.

- o Pseudo Random Generator, used to generate coding coefficient vectors.

- o Symbol Size, used to divide the original data into symbols.

- o GENERATION SIZE or WINDOW SIZE, for block and sliding window codes, respectively.

- o Small or large encoding window, this symbol representation supports both a small and a large coding window, but the variant used is not communicated.

2.4. Small Encoding Window

In a first small encoding window symbol representation, ENCODER RANK is 10 bits long, and the maximum GENERATION/WINDOW SIZE is 2^10.

Figures 2 to 4 below illustrate systematic, coded, and recoded symbol representations within an encoding window of size 2^10. Systematic symbols are uncoded. Coded symbols are compact in form and comprise a seed for coding coefficient generation. Recoded symbols are general in form and comprise the coding coefficient vectors explicitly.

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| 1 |SYMBOLS|   ENCODER RANK    |        SYMBOL(S) DATA         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
|                   SYMBOLS(S) DATA continued                   |
|                              ...                              |
|                                                               |
+---------------------------------------------------------------+
```

Figure 2: A systematic symbol representation.

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| 2 |SYMBOLS|   ENCODER RANK    |     SEED      |SYMBOL(S) DATA |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
|                    SYMBOL(S) DATA continued                   |
|                              ...                              |
|                                                               |
+---------------------------------------------------------------+
```

Figure 3: A compact, coded symbol representation.

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| 3 |SYMBOLS|   ENCODER RANK    |      CODING COEFFICIENTS      |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
|                 CODING COEFFICIENTS continued                 |
|                              ...                              |
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
|                        SYMBOL(S) DATA                         |
|                              ...                              |
|                                                               |
+---------------------------------------------------------------+
```

Figure 4: A recoded symbol representation.

2.4.1. Examples

The following examples show different symbol representations for an illustrative case where the symbol size is 2 bytes, GENERATION/WINDOW SIZE is 8, and field size is 2^8.

Example 1: Three systematic symbols with ID 0, 1 and 2. As the TYPE flag is '1' , SEED/CODING COEFFICIENTS is absent, and ENCODER RANK is the symbol index of the first data symbol with ID 0 in this compact symbol representation.

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| 1 |   3   |         0         |   Systematic Symbol 0 Data    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|   Systematic Symbol 1 Data    |   Systematic Symbol 2 Data    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 5: A symbol representation with 3 systematic, uncoded symbols.

Example 2: Two coded symbols using a compact representation. In this example, TYPE is '2', the SEED to the pseudo-random number generator shared by the sender and receiver is 4. The coding coefficient vector for Symbol A is coefficients 0 to 7 generated by the pseudo-random number generator, the coding coefficient vector for symbol B is coefficients 8 to 15 generated by the pseudo-random number generator.

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| 2 |   2   |         8         |       4       | Coded Symbol A
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
 Data           |      Coded Symbol B Data      |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 6: A symbol representation with 2 coded symbols.

Example 3: Two recoded symbols. Coefficients A0 to A7 constitute the coding coefficient vector for Symbol A, coefficients B0 to B7 constitute the coding coefficient vector for symbol B. In practical implementations, symbol sizes are much larger than 2, leading to amortization of the coding coefficient overheads.

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| 3 |   2   |         8         |      A0       |      A1       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|      A2       |      A3       |      A4       |      A5       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|      A6       |      A7       |      B0       |      B1       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|      B2       |      B3       |      B4       |      B5       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|      B6       |      B7       |      Coded Symbol A Data      |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|      Coded Symbol B Data      |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 7: A symbol representation with 2 recoded symbols having coding coefficients attached.

2.5. Large Encoding Window

In a second large encoding window symbol representation, ENCODER RANK is 18-bit long, and the maximum GENERATION/WINDOW SIZE is 2^18.

Figures 8 to 10 below illustrate systematic, coded, and recoded symbol representations within an encoding window of size 2^18. Systematic symbols are uncoded. Coded symbols are compact in form and comprise a seed for coding coefficient generation. Recoded symbols are general in form and comprise the coding coefficient vectors explicitly (CODING COEFFICIENTS or CODING COEFFS).

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| 1 |SYMBOLS|           ENCODER RANK            |SYMBOL(S) DATA |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
|                     SYMBOL(S) DATA continued                  |
|                               ...                             |
|                                                               |
+---------------------------------------------------------------+
```

Figure 8: A systematic symbol representation.

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| 2 |SYMBOLS|           ENCODER RANK            |     SEED      |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
|                         SYMBOL(S) DATA                        |
|                               ...                             |
|                                                               |
+---------------------------------------------------------------+
```

Figure 9: A coded symbol representation.

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
| 3 |SYMBOLS|           ENCODER RANK            | CODING COEFFS |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
|                 CODING COEFFICIENTS continued                 |
|                               ...                             |
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                                                               |
|                         SYMBOL(S) DATA                        |
|                               ...                             |
|                                                               |
+---------------------------------------------------------------+
```

Figure 10: A recoded symbol representation.

3. Security Considerations

This document does not present new security considerations.

4. IANA Considerations

This document has no actions for IANA.

5. References

5.1. Normative References

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

[RLNC-Background]
Heide, J., Shi, S., Fouli, K., Medard, M., and V. Chook, "Random Linear Network Coding (RLNC): Background and Practical Considerations", February 2018, <https://datatracker.ietf.org/doc/draft-heide-nwcrg-rlnc-background/>.

5.2. Informative References

[HK03] Ho, T., Koetter, R., Medard, M., Karger, D., and M. Effros, "The Benefits of Coding over Routing in a Randomized Setting", July 2003, <http://ieeexplore.ieee.org/document/1228459/>.

Authors' Addresses

Janus Heide
Steinwurf Aps
Aalborg
Denmark

Email: janus@steinwurf.com

Shirley Shi
Code On Network Coding LLC
Cambridge
USA

Email: xshi@alum.mit.edu

Kerim Fouli
Code On Network Coding LLC
Cambridge
USA

Email: fouli@codeontechnologies.com

Muriel Medard
Code On Network Coding LLC
Cambridge
USA

Email: muriel.medard@codeontechnologies.com

Vince Chook
Inmarsat PLC
London
United Kingdom

Email: Vince.Chook@inmarsat.com