Digital Communications
SSY125, Lecture 14

Coding beyond reliable communication

Alexandre Graell i Amat
alexandre.graell@chalmers.se
https://sites.google.com/site/agraellamat

CHALMERS

December 4, 2019

Coding beyond coding

We can now provably (and in practice) achieve capacity for most channels.

Coding beyond coding

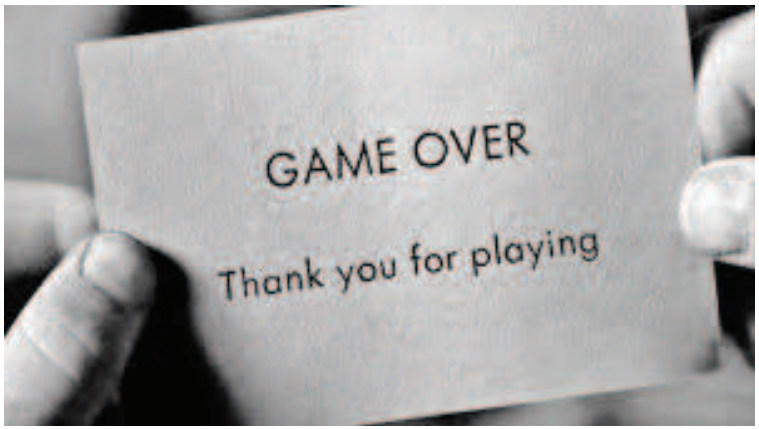
We can now provably (and in practice) achieve capacity for most channels.

- Channel coding for classical applications has reached the point of diminishing returns

Coding beyond coding

We can now provably (and in practice) achieve capacity for most channels.

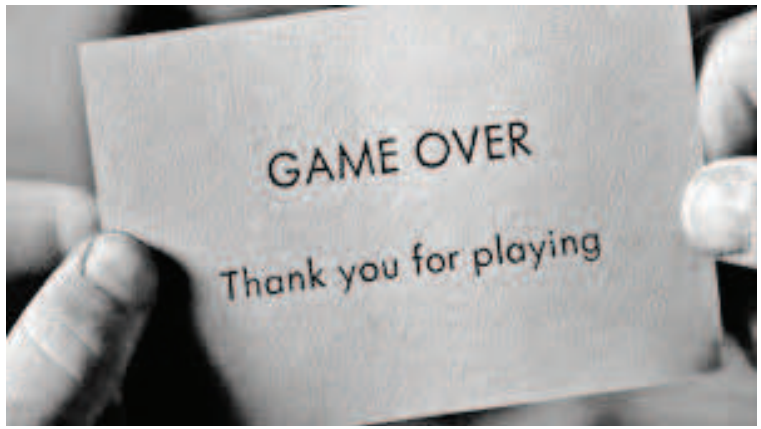
- Channel coding for classical applications has reached the point of diminishing returns



Coding beyond coding

We can now provably (and in practice) achieve capacity for most channels.

- Channel coding for classical applications has reached the point of diminishing returns

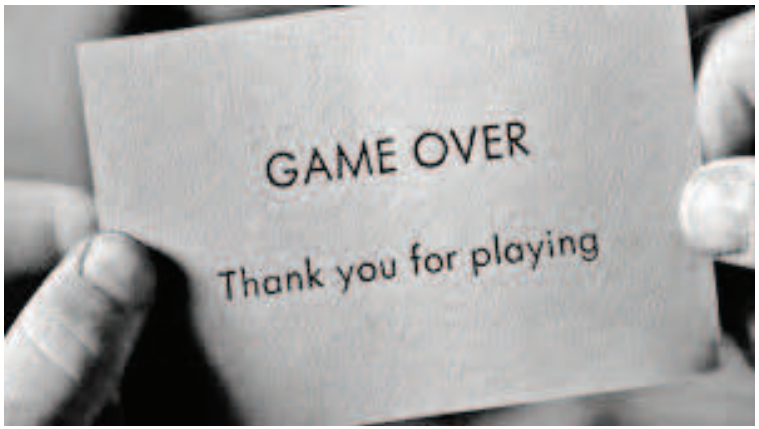


- A very powerful tool for many applications and research fields:

Coding beyond coding

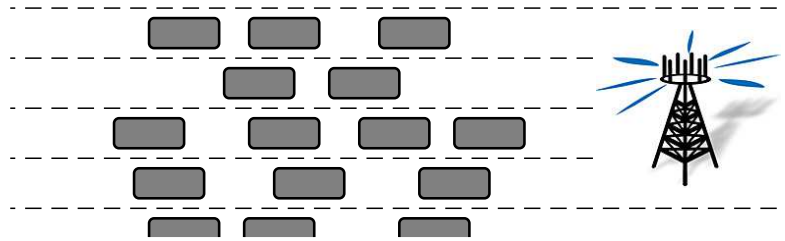
We can now provably (and in practice) achieve capacity for most channels.

- Channel coding for classical applications has reached the point of diminishing returns

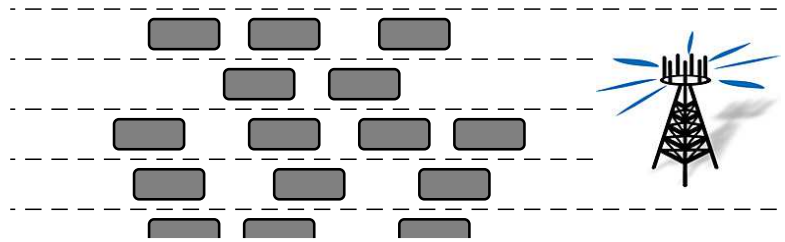


- A very powerful tool for many applications and research fields:
  - Uncoordinated multiple access
  - Post-quantum cryptography
  - DNA storage
  - Caching
  - Distributed computing

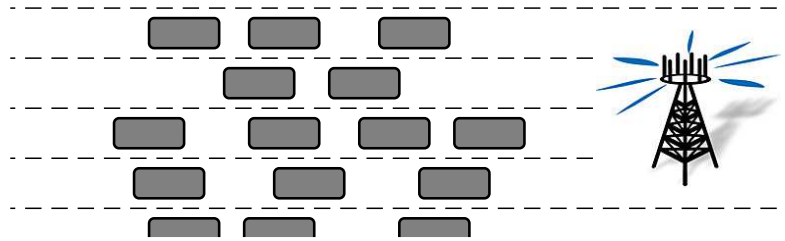
Uncoordinated multiple access
Slotted ALOHA

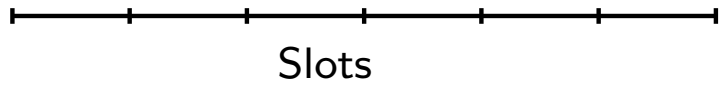


Uncoordinated multiple access
Slotted ALOHA

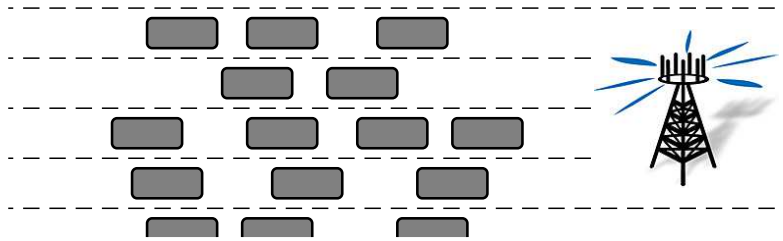


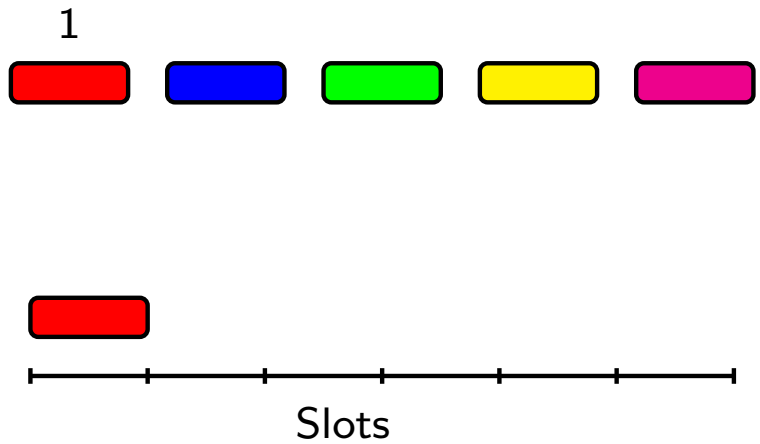
Uncoordinated multiple access
Slotted ALOHA



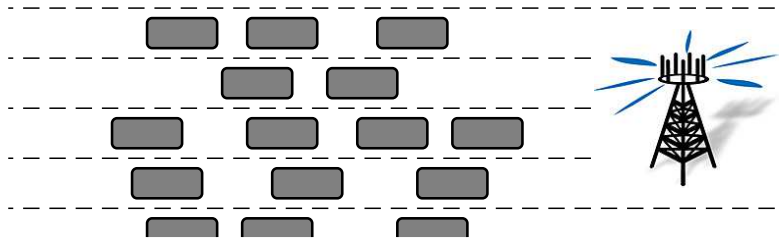


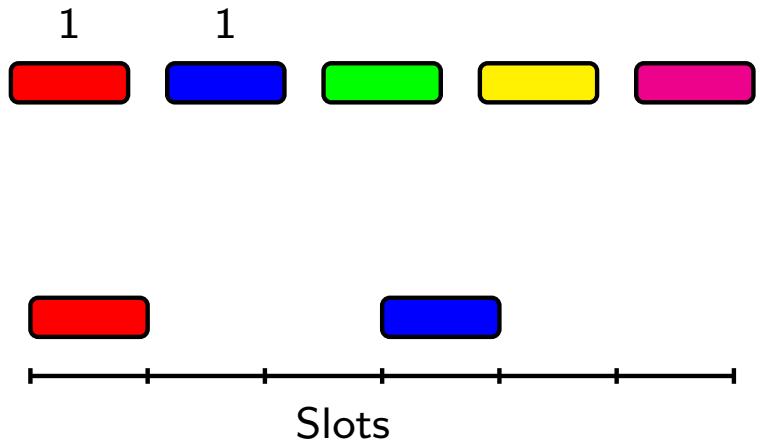
Uncoordinated multiple access
Slotted ALOHA



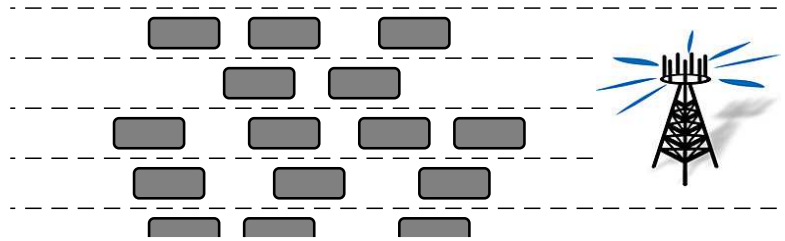


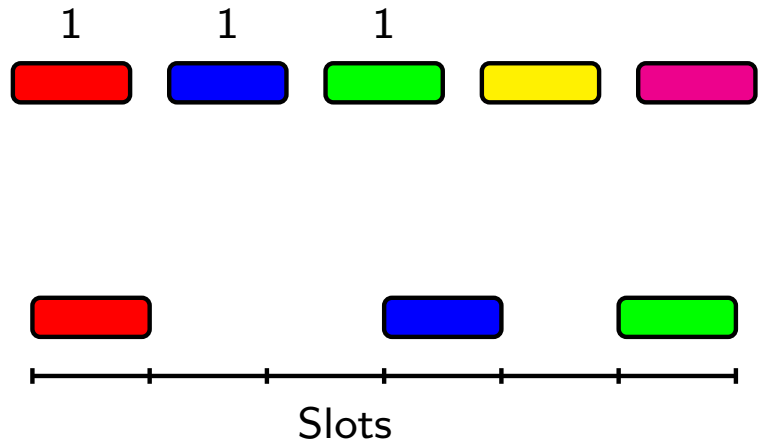
Uncoordinated multiple access
Slotted ALOHA



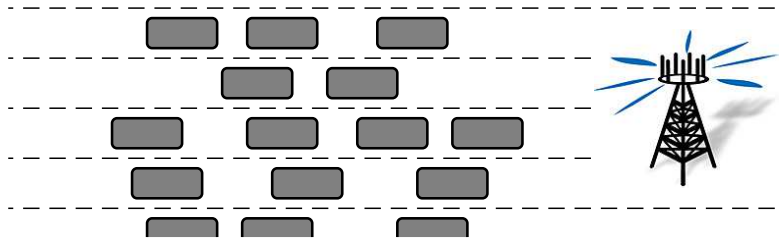


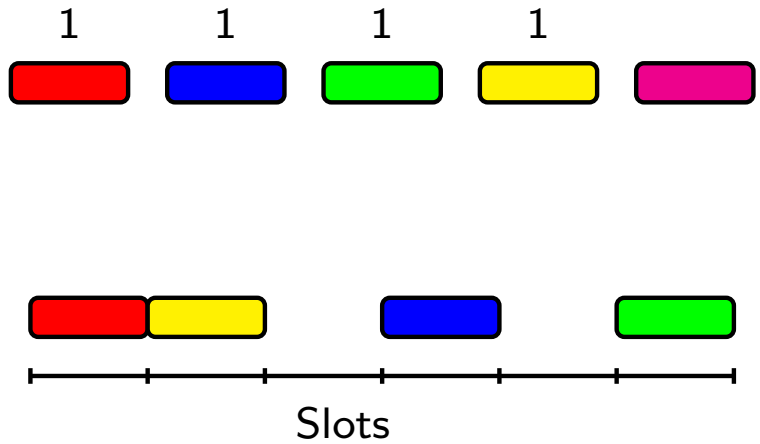
Uncoordinated multiple access
Slotted ALOHA



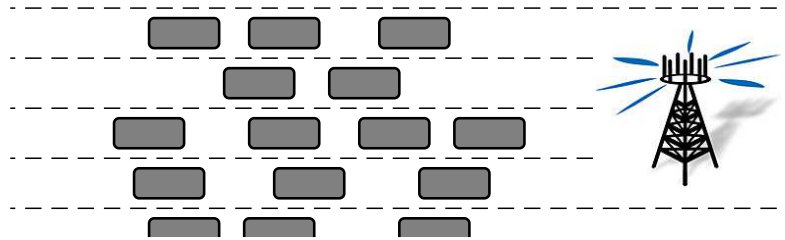


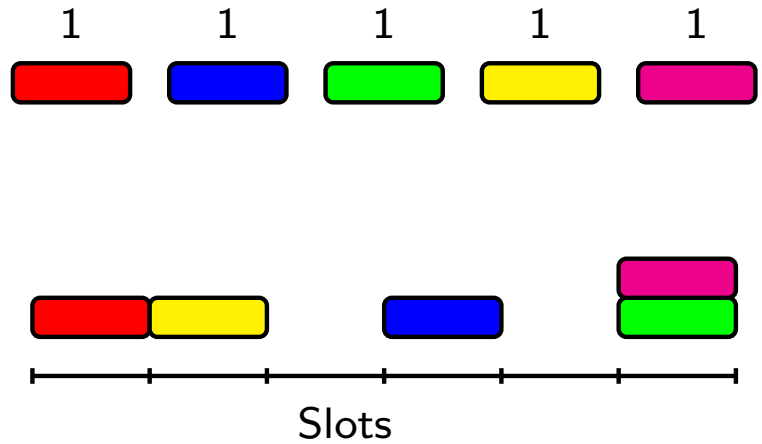
Uncoordinated multiple access
Slotted ALOHA



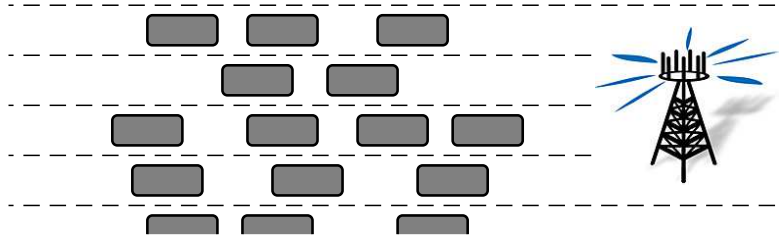


Uncoordinated multiple access
Slotted ALOHA

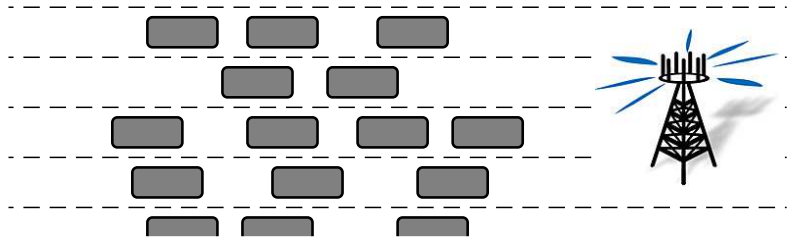




Uncoordinated multiple access
Coded slotted ALOHA

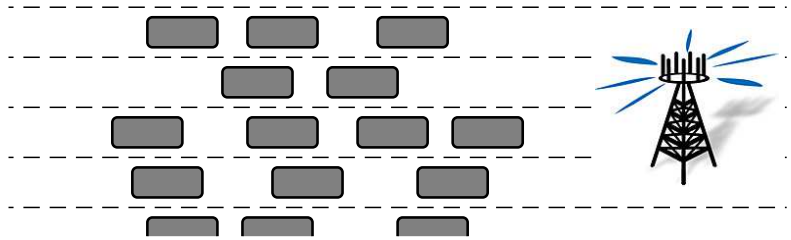


Uncoordinated multiple access
Coded slotted ALOHA



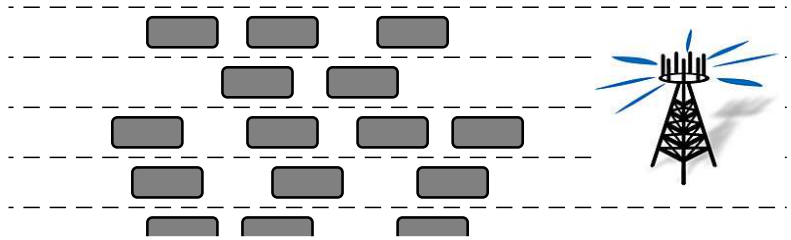
- Users repeat their packets (introduce redundancy!)

Uncoordinated multiple access
Coded slotted ALOHA



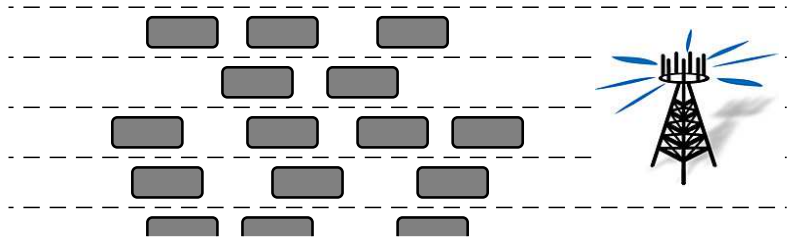
- Users repeat their packets (introduce redundancy!)
- Decoding using successive interference cancelation, exploiting the introduced redundancy

Uncoordinated multiple access
Coded slotted ALOHA

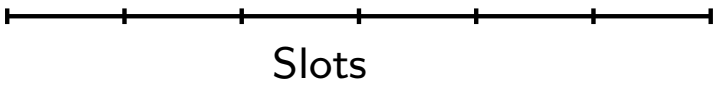


- Users repeat their packets (introduce redundancy!)
- Decoding using successive interference cancelation, exploiting the introduced redundancy

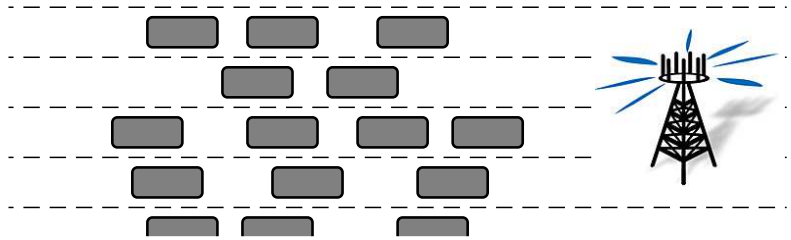
Uncoordinated multiple access
Coded slotted ALOHA



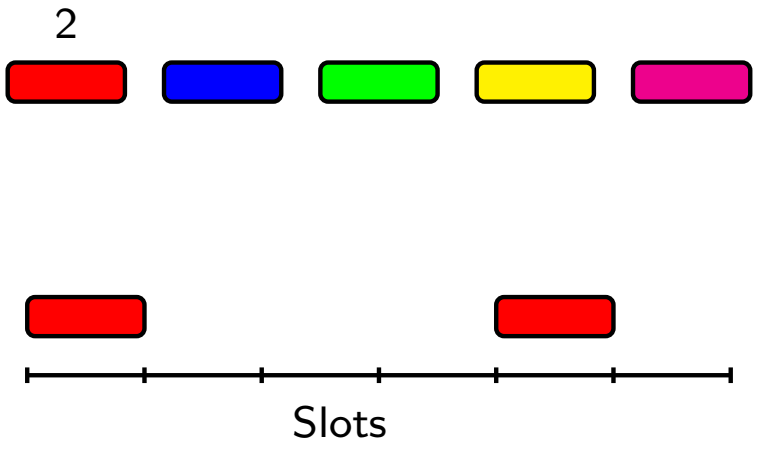
- Users repeat their packets (introduce redundancy!)
- Decoding using successive interference cancelation, exploiting the introduced redundancy



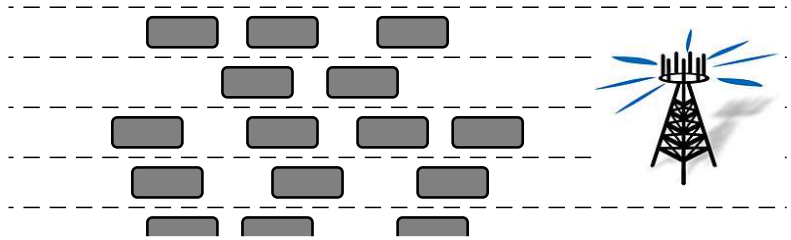
Uncoordinated multiple access
Coded slotted ALOHA



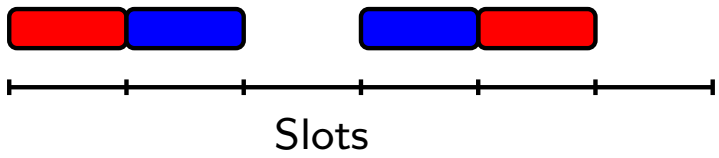
- Users repeat their packets (introduce redundancy!)
- Decoding using successive interference cancelation, exploiting the introduced redundancy



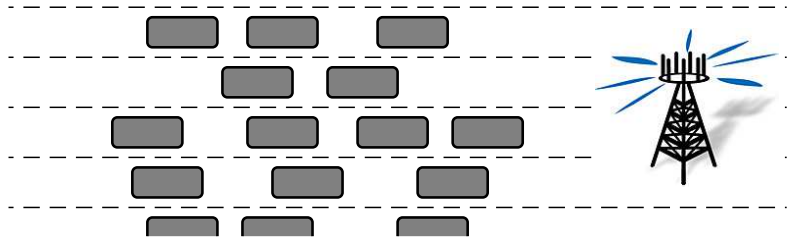
Uncoordinated multiple access
Coded slotted ALOHA



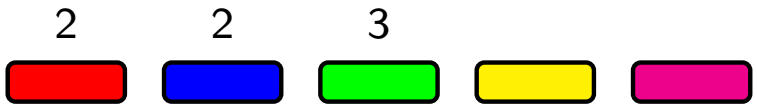
- Users repeat their packets (introduce redundancy!)
- Decoding using successive interference cancelation, exploiting the introduced redundancy

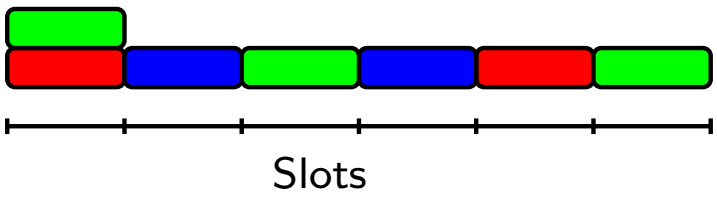


Uncoordinated multiple access
Coded slotted ALOHA

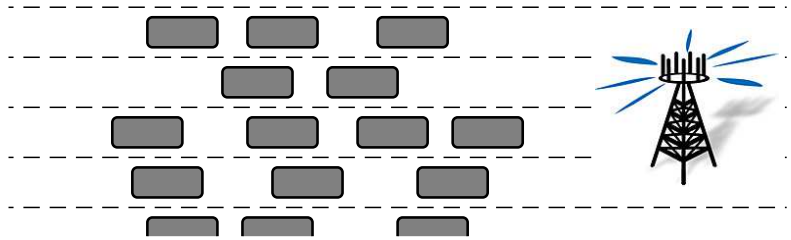


- Users repeat their packets (introduce redundancy!)
- Decoding using successive interference cancelation, exploiting the introduced redundancy

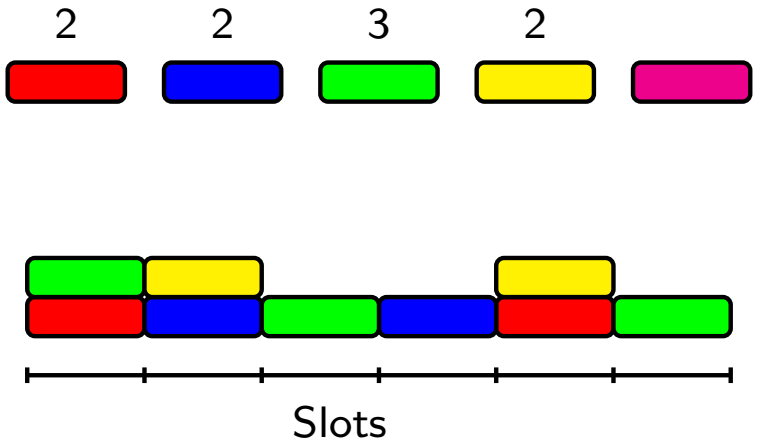




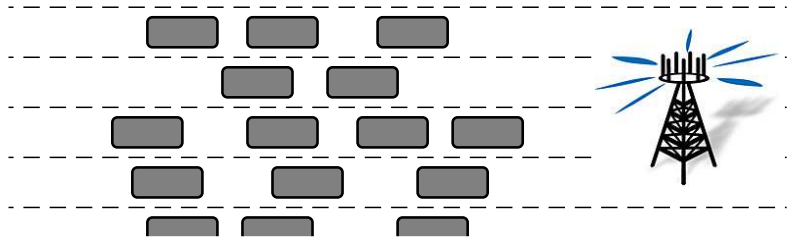
Uncoordinated multiple access
Coded slotted ALOHA



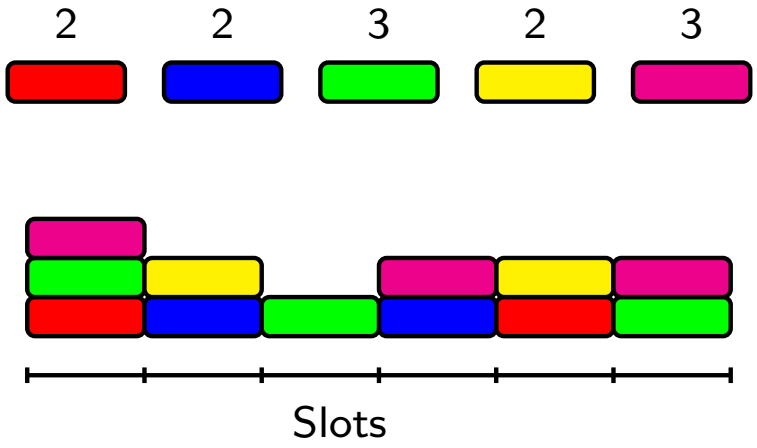
- Users repeat their packets (introduce redundancy!)
- Decoding using successive interference cancelation, exploiting the introduced redundancy



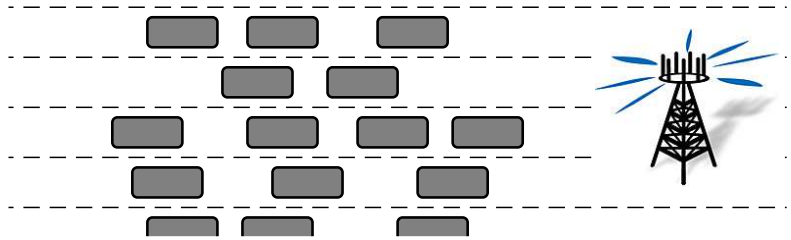
Uncoordinated multiple access
Coded slotted ALOHA



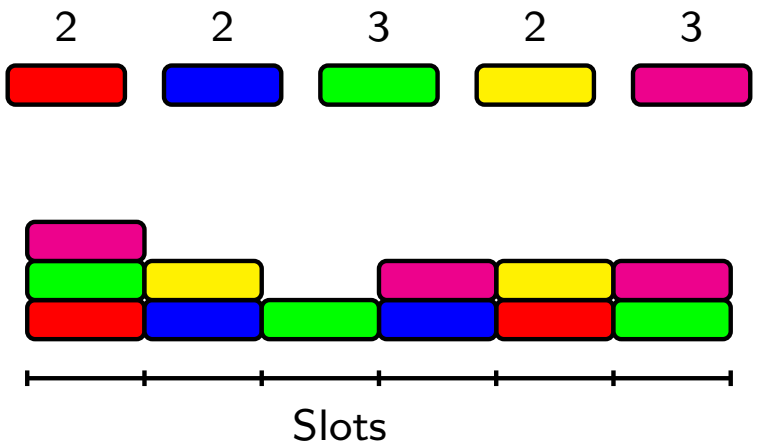
- Users repeat their packets (introduce redundancy!)
- Decoding using successive interference cancelation, exploiting the introduced redundancy



Uncoordinated multiple access
Coded slotted ALOHA

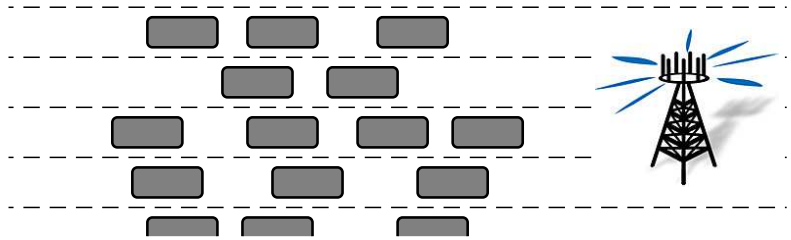


- Users repeat their packets (introduce redundancy!)
- Decoding using successive interference cancelation, exploiting the introduced redundancy

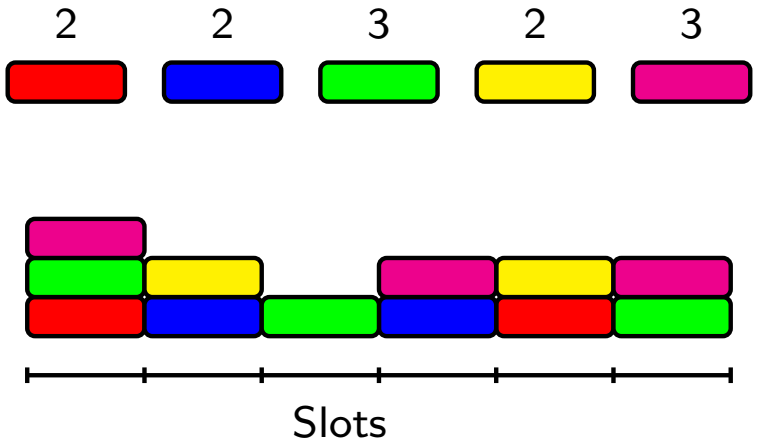


Alternative representation

Uncoordinated multiple access
Coded slotted ALOHA

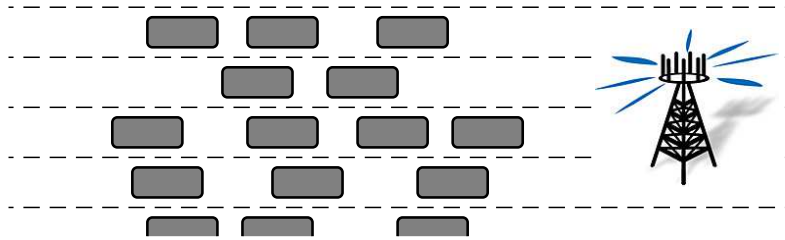


- Users repeat their packets (introduce redundancy!)
- Decoding using successive interference cancelation, exploiting the introduced redundancy

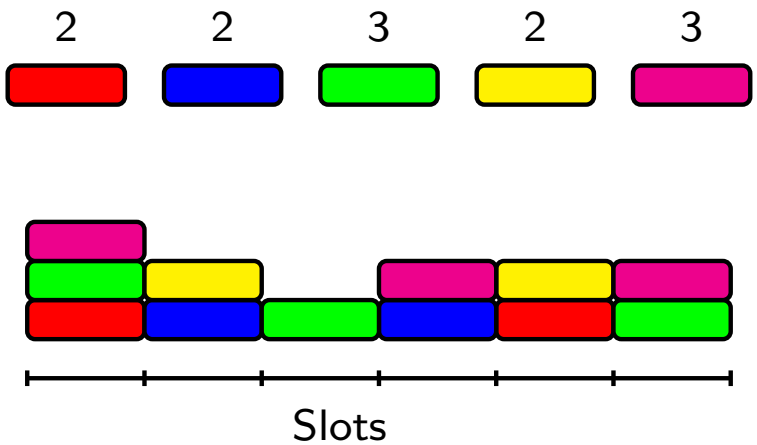


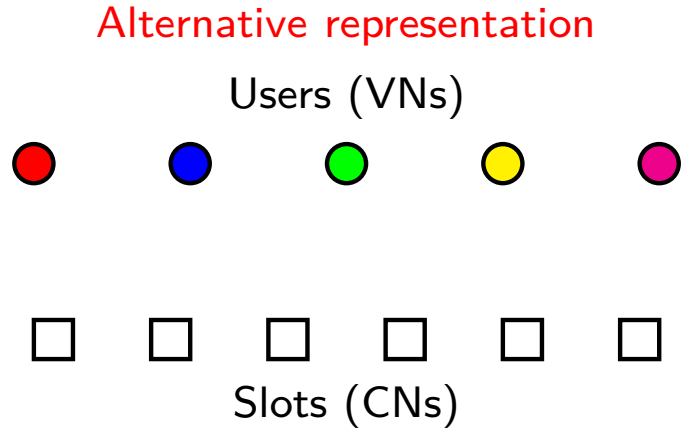
Alternative representation

Uncoordinated multiple access
Coded slotted ALOHA

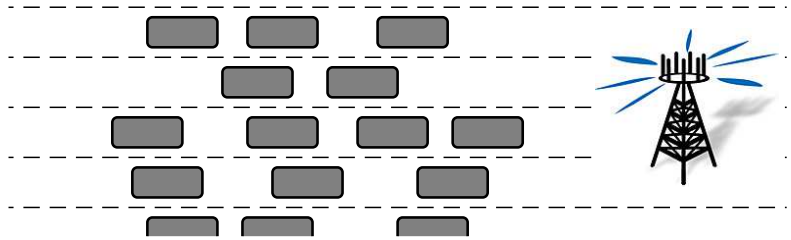


- Users repeat their packets (introduce redundancy!)
- Decoding using successive interference cancelation, exploiting the introduced redundancy

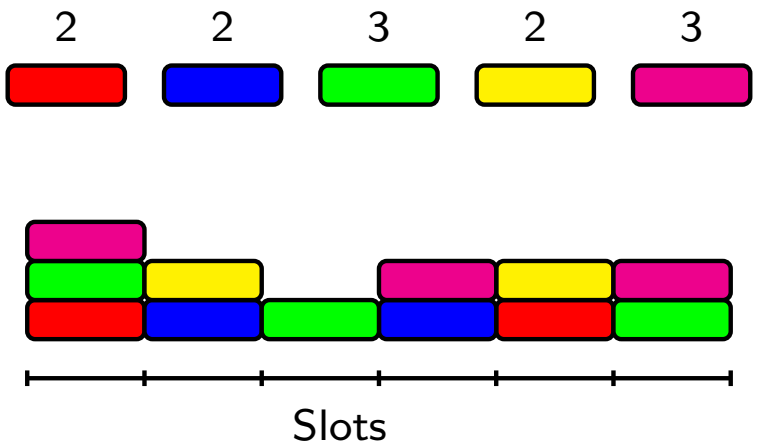




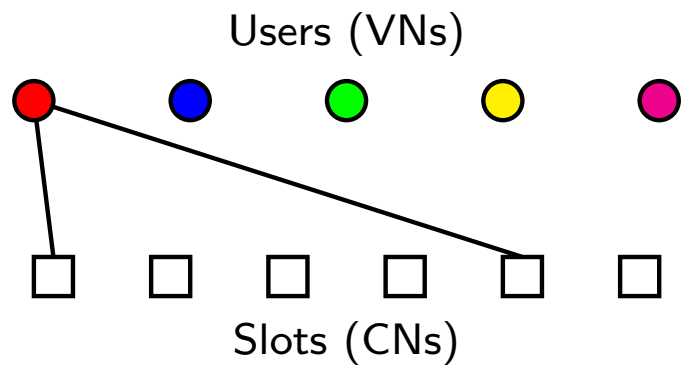
Uncoordinated multiple access
Coded slotted ALOHA



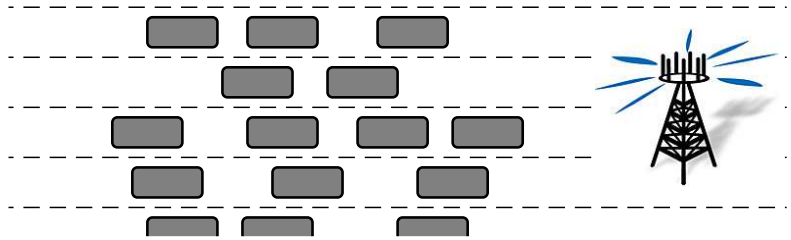
- Users repeat their packets (introduce redundancy!)
- Decoding using successive interference cancelation, exploiting the introduced redundancy



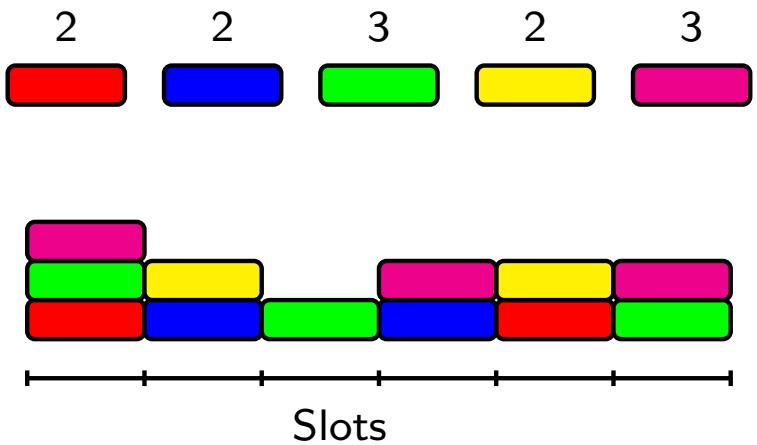
Alternative representation



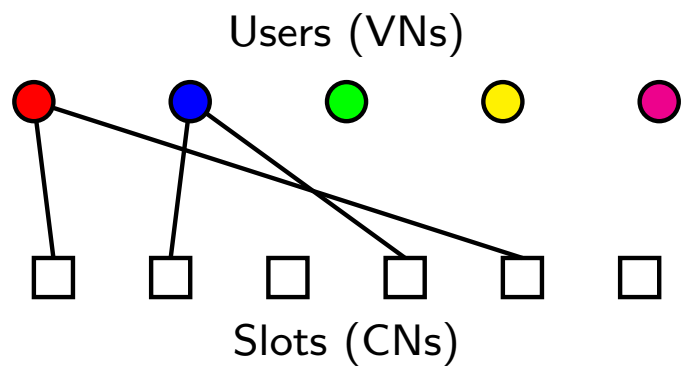
Uncoordinated multiple access
Coded slotted ALOHA



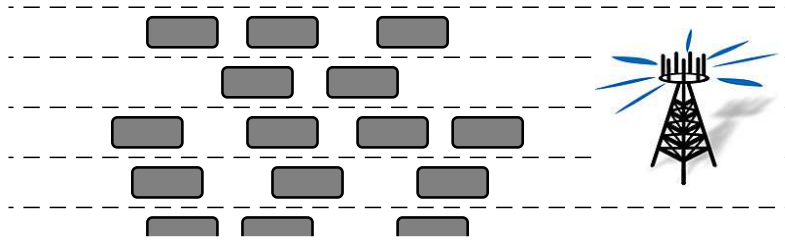
- Users repeat their packets (introduce redundancy!)
- Decoding using successive interference cancelation, exploiting the introduced redundancy



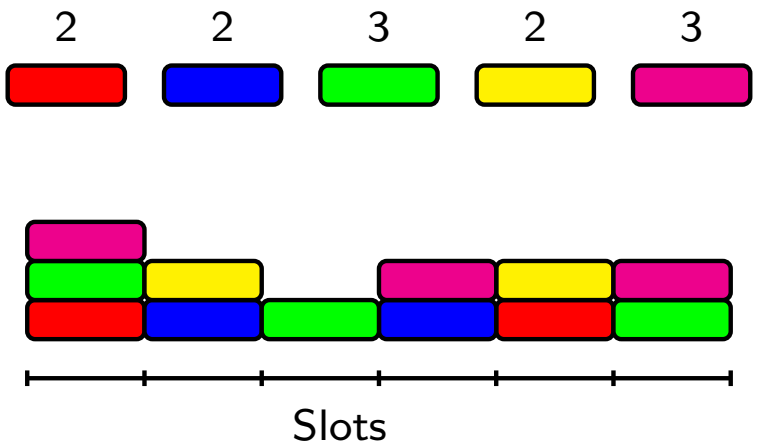
Alternative representation



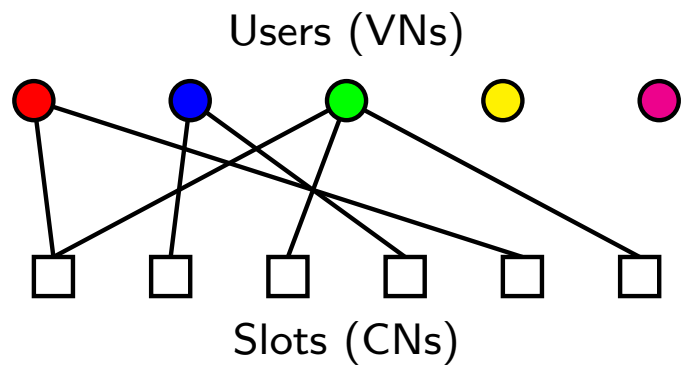
Uncoordinated multiple access
Coded slotted ALOHA



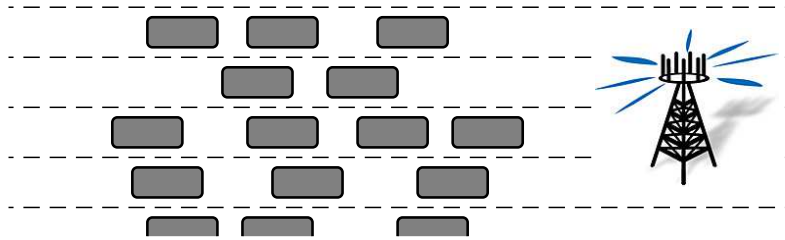
- Users repeat their packets (introduce redundancy!)
- Decoding using successive interference cancelation, exploiting the introduced redundancy



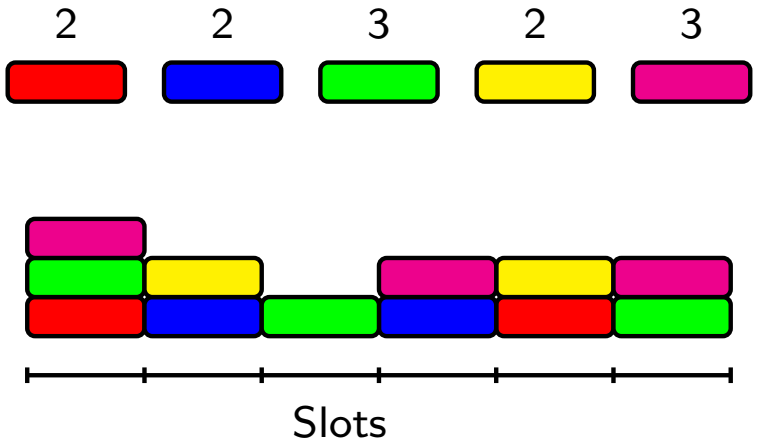
Alternative representation



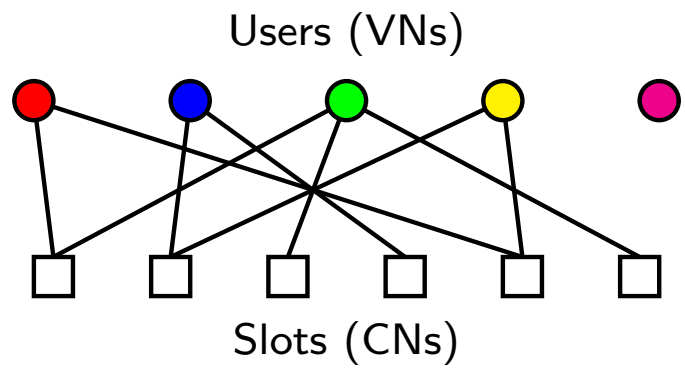
Uncoordinated multiple access
Coded slotted ALOHA



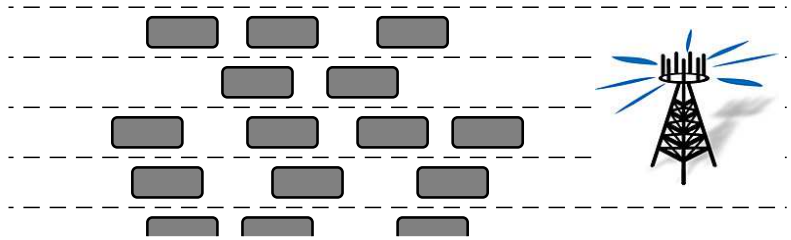
- Users repeat their packets (introduce redundancy!)
- Decoding using successive interference cancelation, exploiting the introduced redundancy



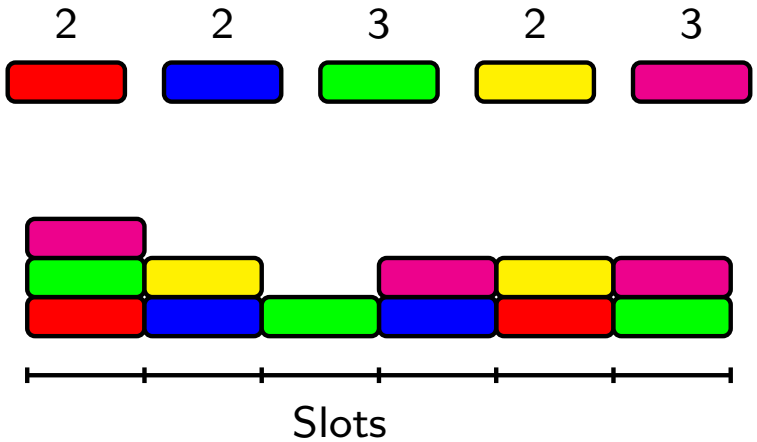
Alternative representation



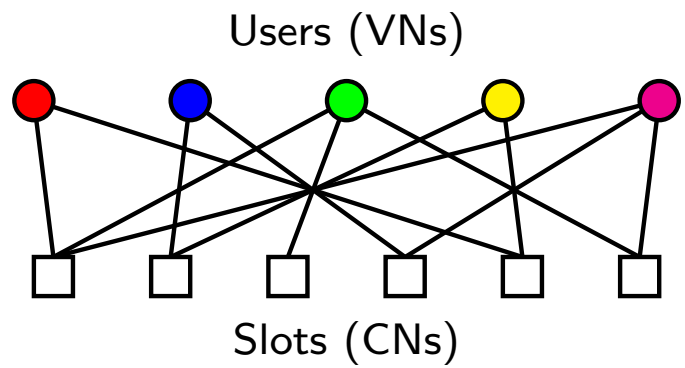
Uncoordinated multiple access
Coded slotted ALOHA



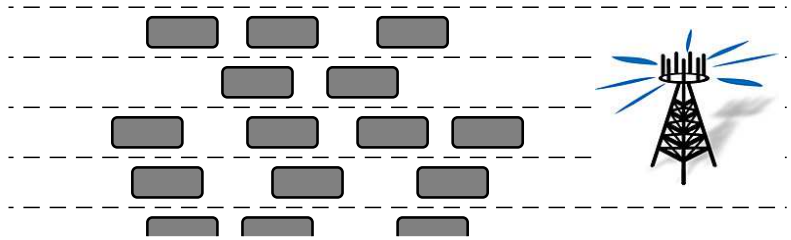
- Users repeat their packets (introduce redundancy!)
- Decoding using successive interference cancelation, exploiting the introduced redundancy



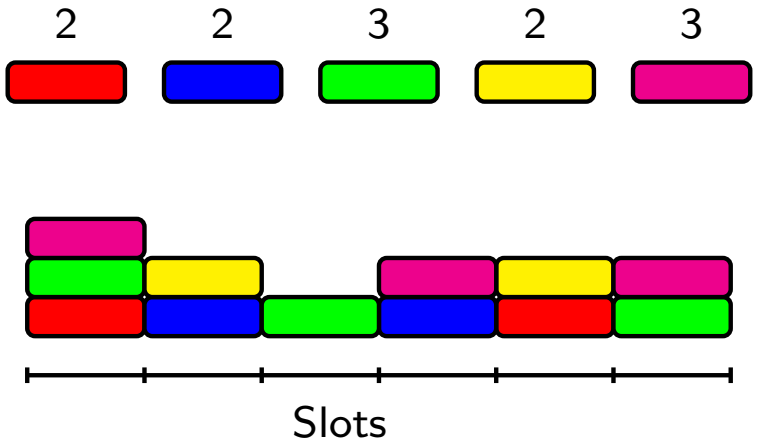
Alternative representation



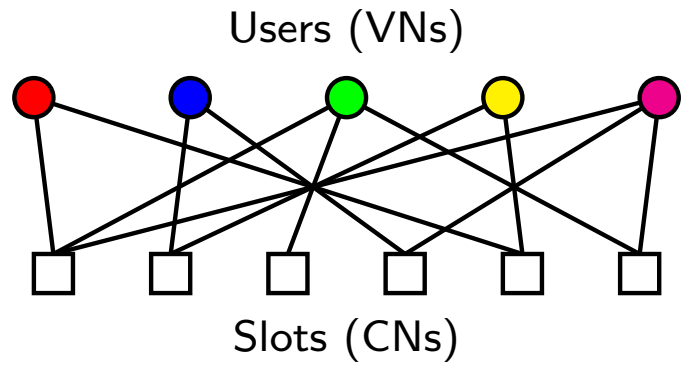
Uncoordinated multiple access
Coded slotted ALOHA



- Users repeat their packets (introduce redundancy!)
- Decoding using successive interference cancelation, exploiting the introduced redundancy

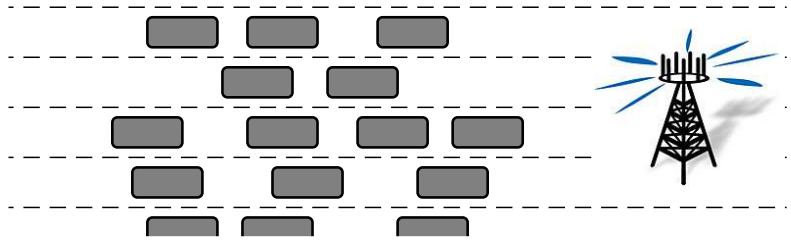


Alternative representation

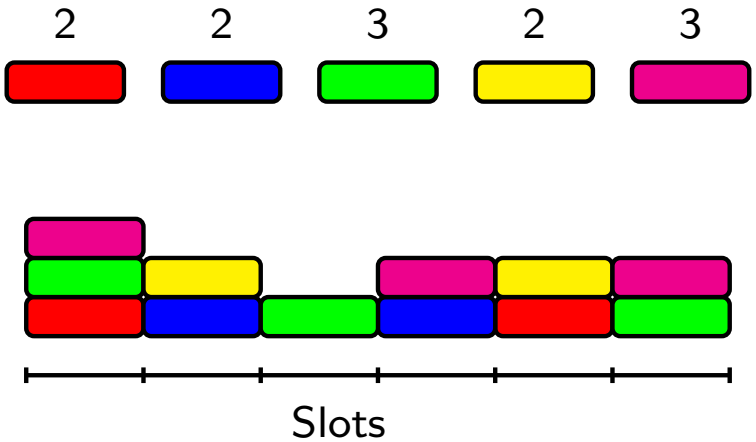


- A coding problem!

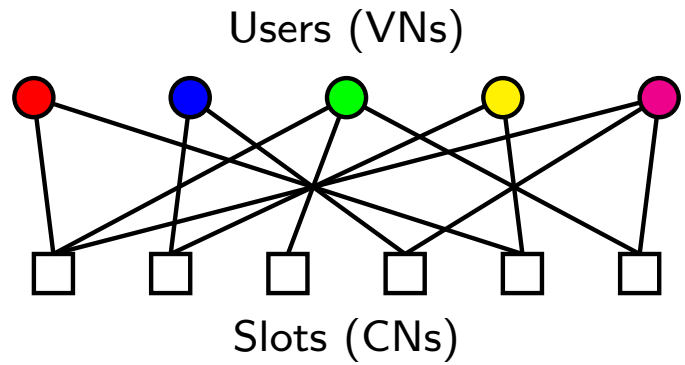
Uncoordinated multiple access
Coded slotted ALOHA



- Users repeat their packets (introduce redundancy!)
- Decoding using successive interference cancelation, exploiting the introduced redundancy

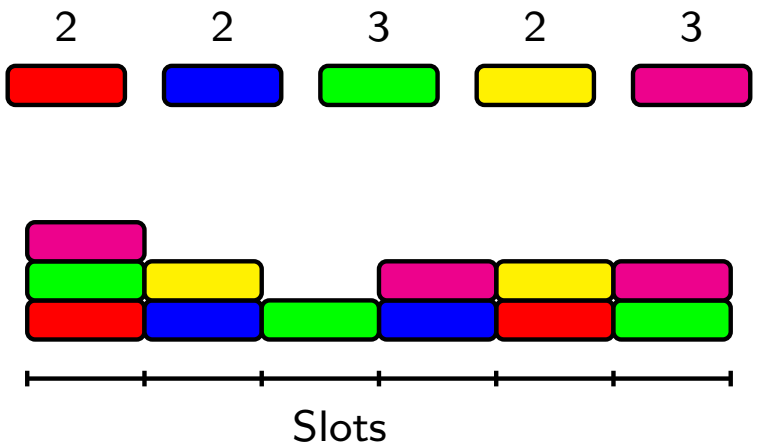


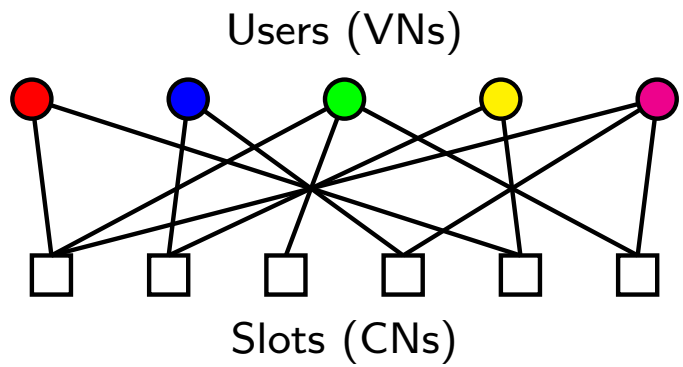
Alternative representation



- A coding problem! $\rightarrow$ Can be analyzed using the theory of codes on graphs

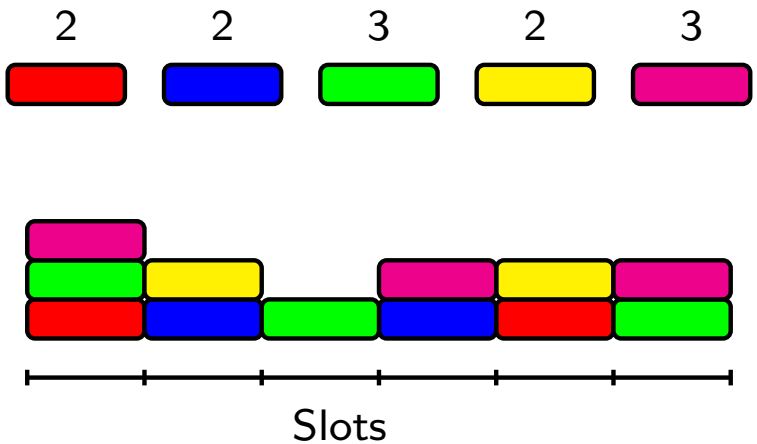
Coded slotted ALOHA: Decoding

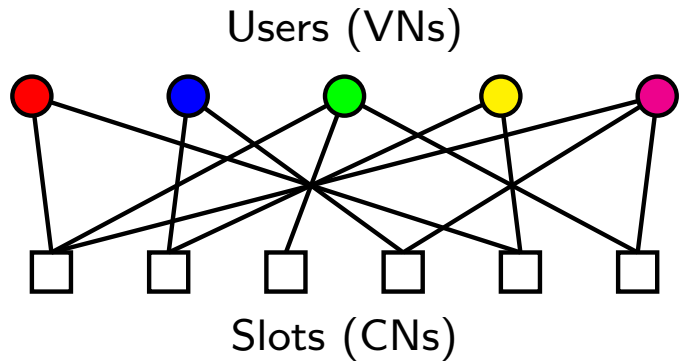




Coded slotted ALOHA: Decoding

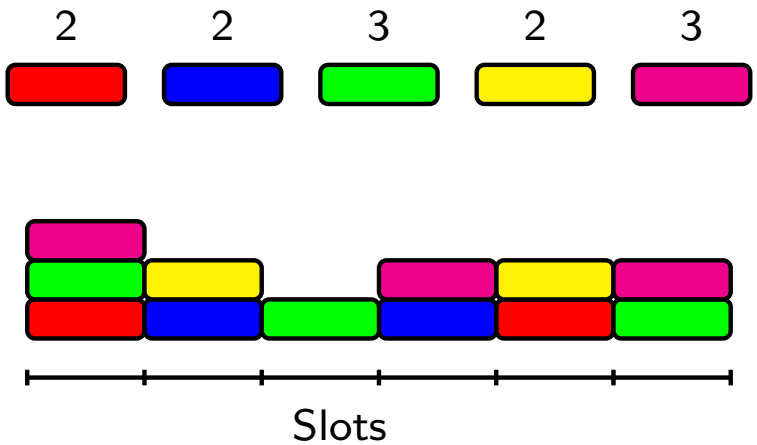
- A packet can be reliably decoded if no collision occurs

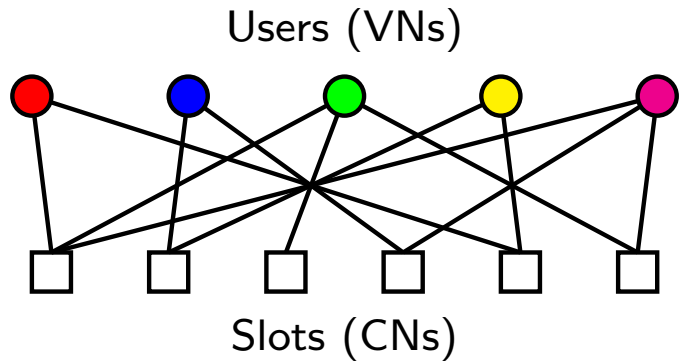




Coded slotted ALOHA: Decoding

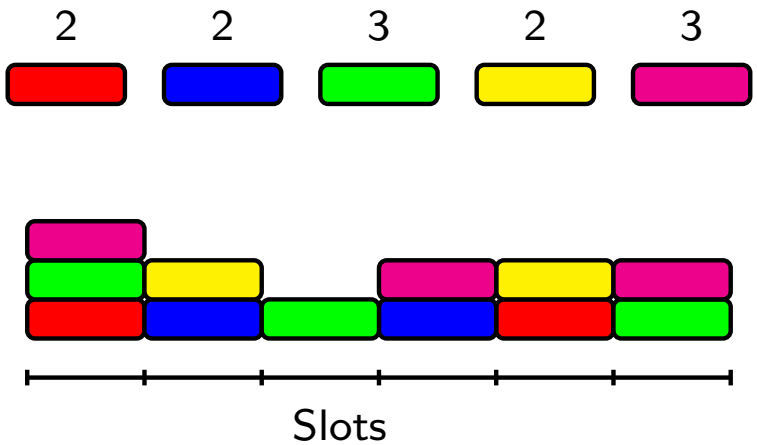
- A packet can be reliably decoded if no collision occurs
- Each packet contains pointers to its replicas

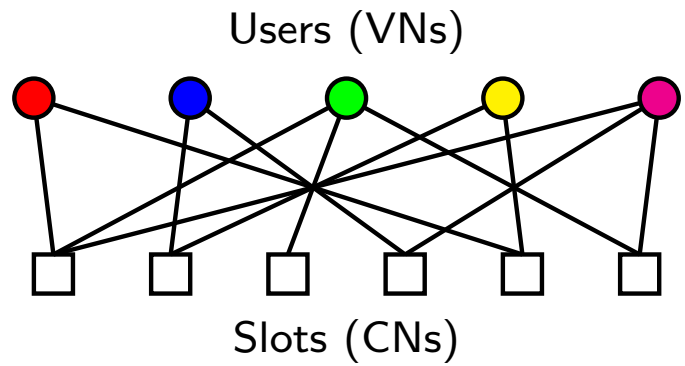




Coded slotted ALOHA: Decoding

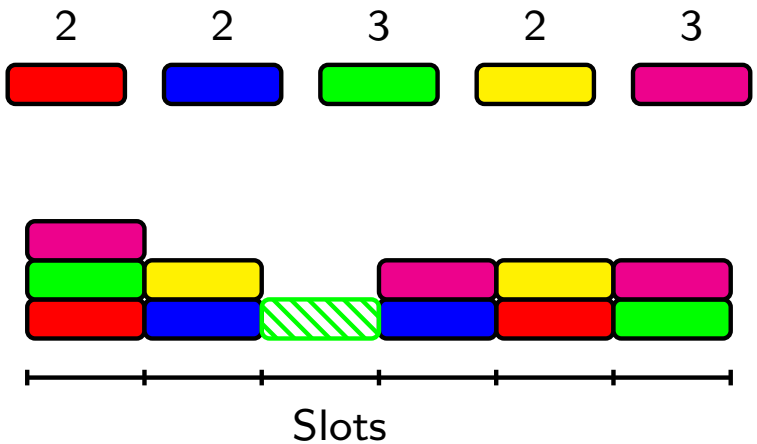
- A packet can be reliably decoded if no collision occurs
- Each packet contains pointers to its replicas
- The replicas of a decoded packet can be perfectly removed from the signal

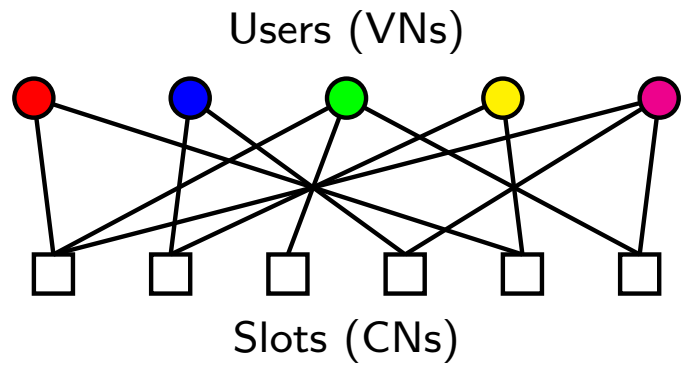




Coded slotted ALOHA: Decoding

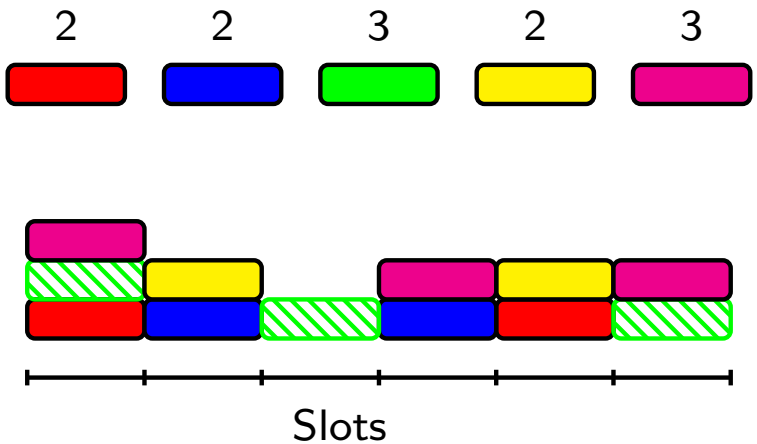
- A packet can be reliably decoded if no collision occurs
- Each packet contains pointers to its replicas
- The replicas of a decoded packet can be perfectly removed from the signal

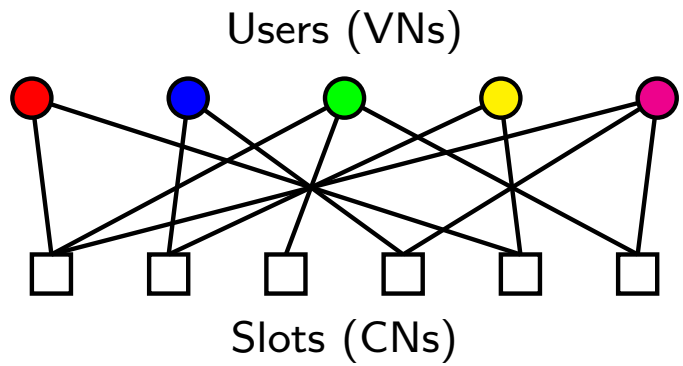




Coded slotted ALOHA: Decoding

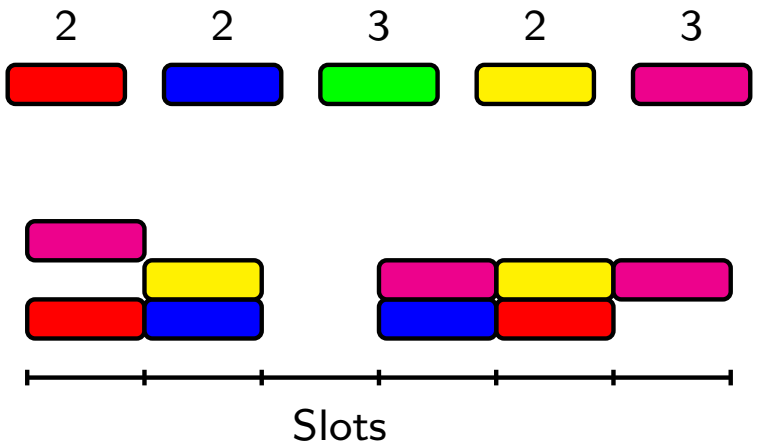
- A packet can be reliably decoded if no collision occurs
- Each packet contains pointers to its replicas
- The replicas of a decoded packet can be perfectly removed from the signal

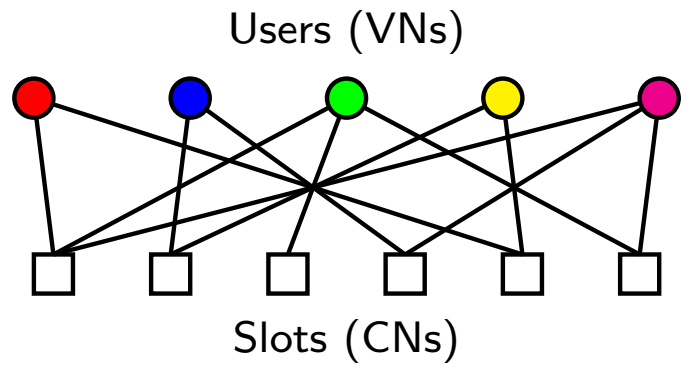




Coded slotted ALOHA: Decoding

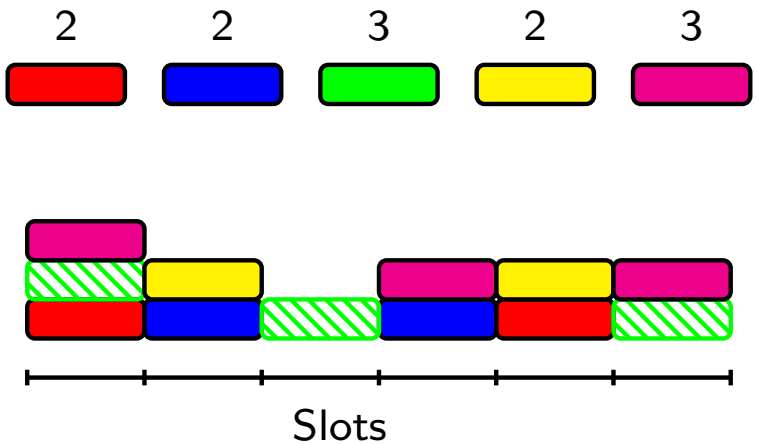
- A packet can be reliably decoded if no collision occurs
- Each packet contains pointers to its replicas
- The replicas of a decoded packet can be perfectly removed from the signal

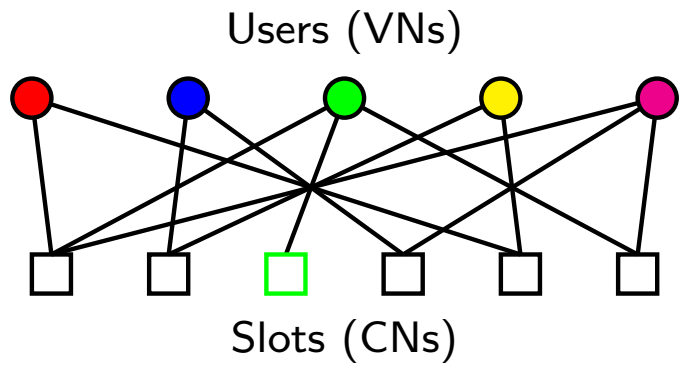




Coded slotted ALOHA: Decoding

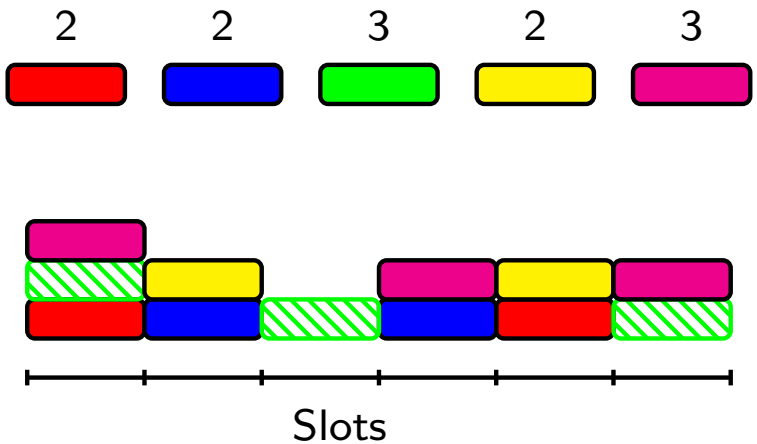
- A packet can be reliably decoded if no collision occurs
- Each packet contains pointers to its replicas
- The replicas of a decoded packet can be perfectly removed from the signal

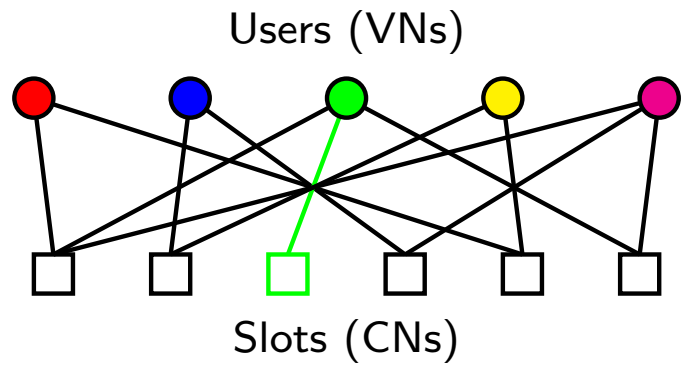




Coded slotted ALOHA: Decoding

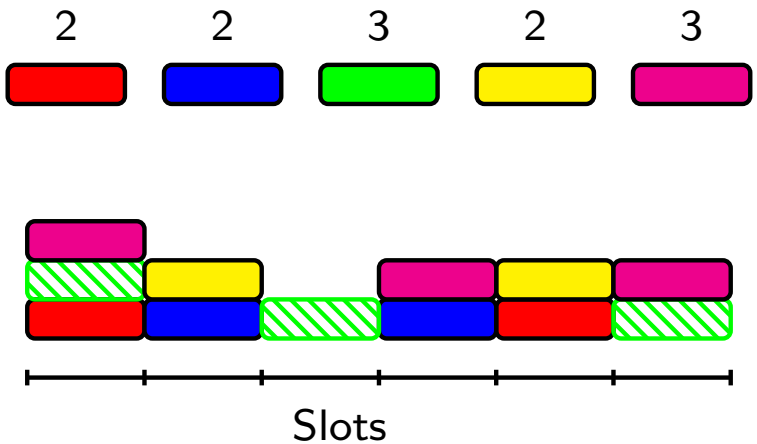
- A packet can be reliably decoded if no collision occurs
- Each packet contains pointers to its replicas
- The replicas of a decoded packet can be perfectly removed from the signal

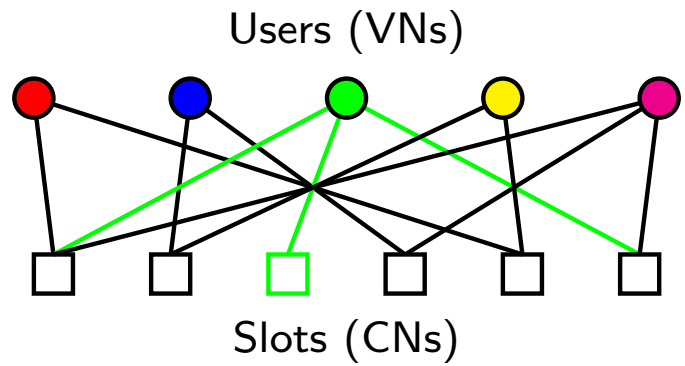




Coded slotted ALOHA: Decoding

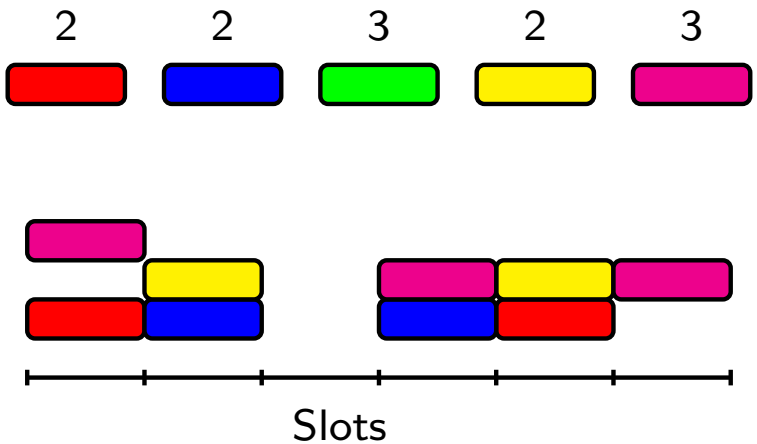
- A packet can be reliably decoded if no collision occurs
- Each packet contains pointers to its replicas
- The replicas of a decoded packet can be perfectly removed from the signal

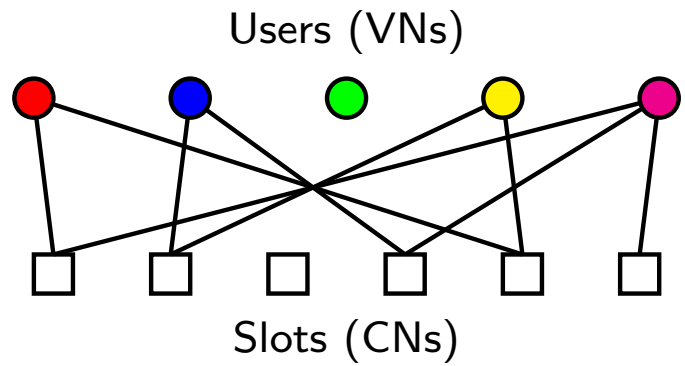




Coded slotted ALOHA: Decoding

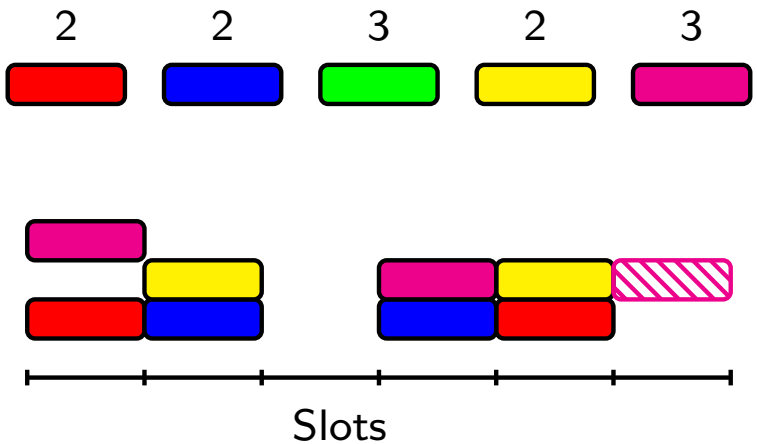
- A packet can be reliably decoded if no collision occurs
- Each packet contains pointers to its replicas
- The replicas of a decoded packet can be perfectly removed from the signal

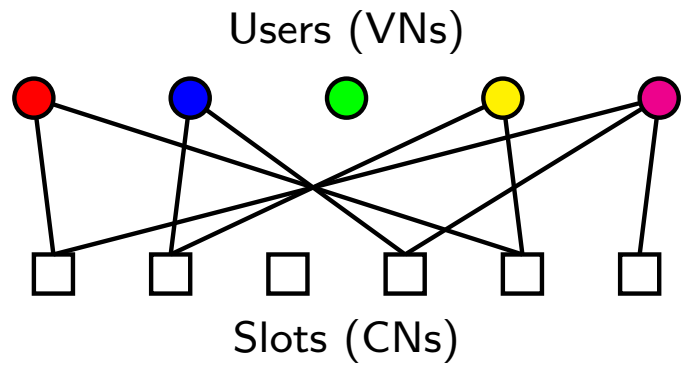




Coded slotted ALOHA: Decoding

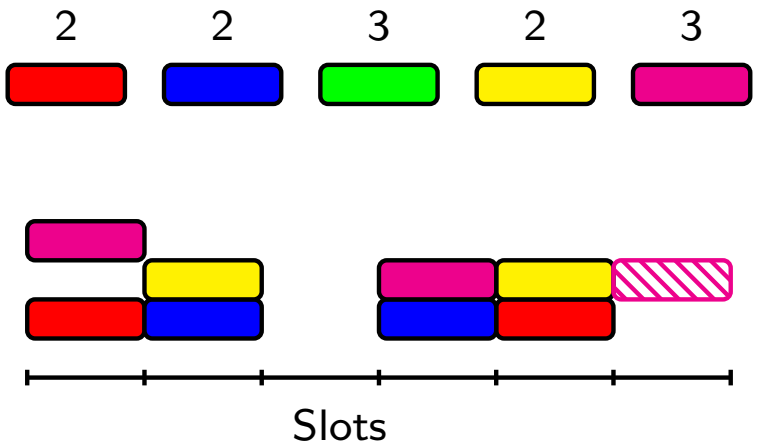
- A packet can be reliably decoded if no collision occurs
- Each packet contains pointers to its replicas
- The replicas of a decoded packet can be perfectly removed from the signal

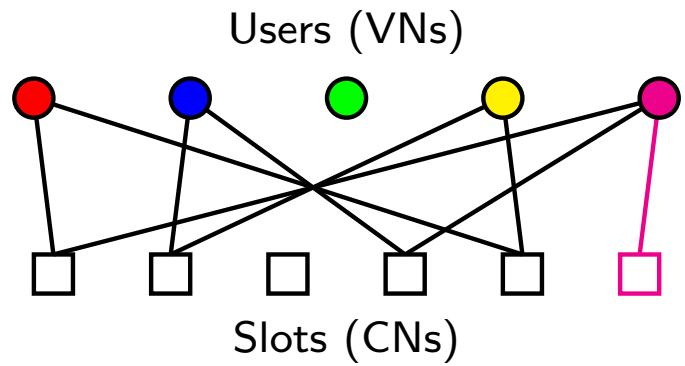




Coded slotted ALOHA: Decoding

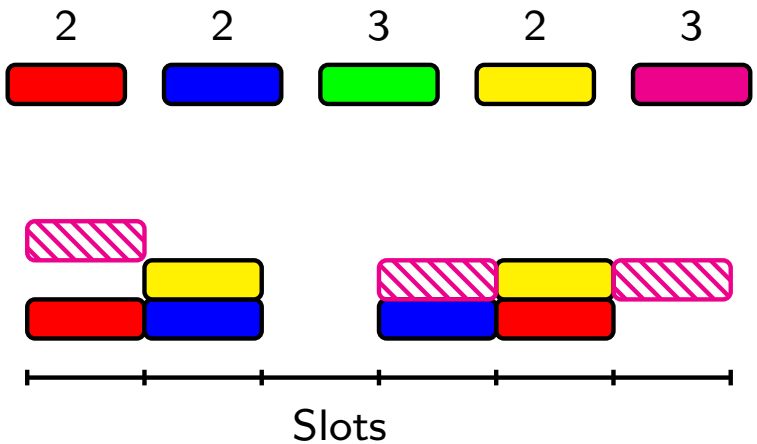
- A packet can be reliably decoded if no collision occurs
- Each packet contains pointers to its replicas
- The replicas of a decoded packet can be perfectly removed from the signal

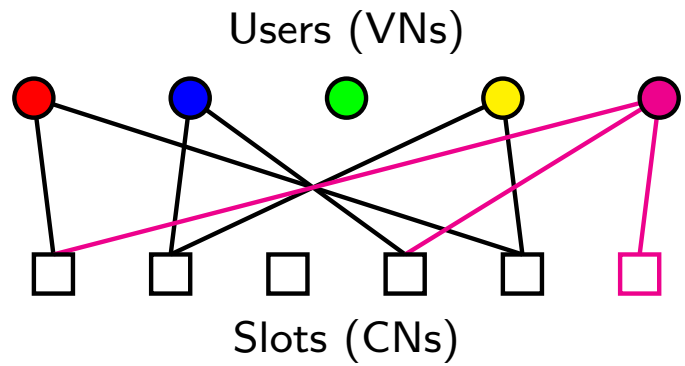




Coded slotted ALOHA: Decoding

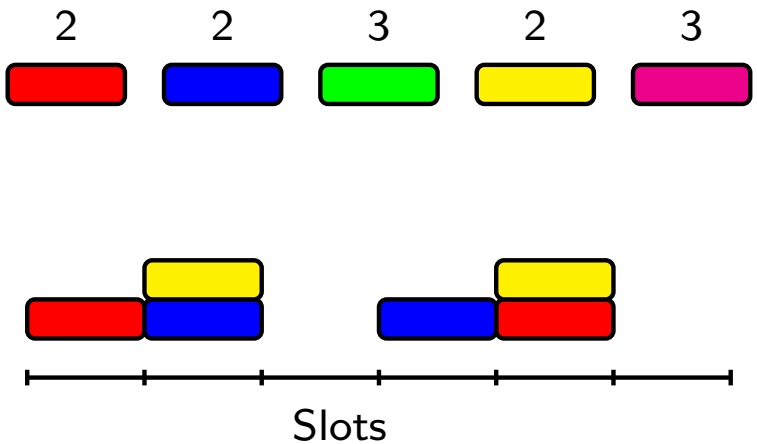
- A packet can be reliably decoded if no collision occurs
- Each packet contains pointers to its replicas
- The replicas of a decoded packet can be perfectly removed from the signal

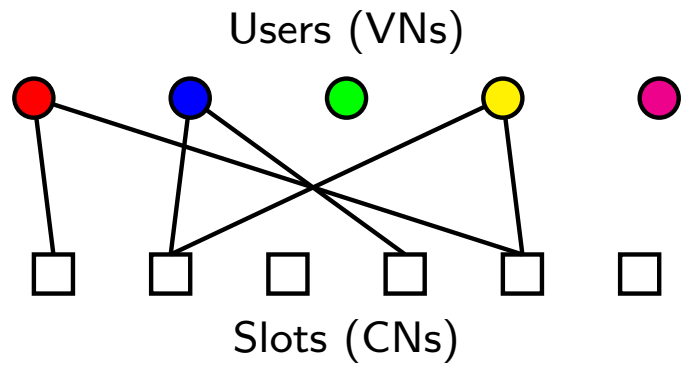




Coded slotted ALOHA: Decoding

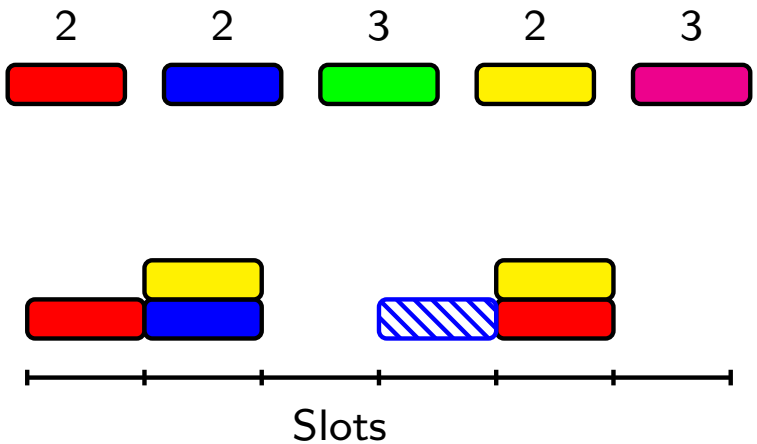
- A packet can be reliably decoded if no collision occurs
- Each packet contains pointers to its replicas
- The replicas of a decoded packet can be perfectly removed from the signal

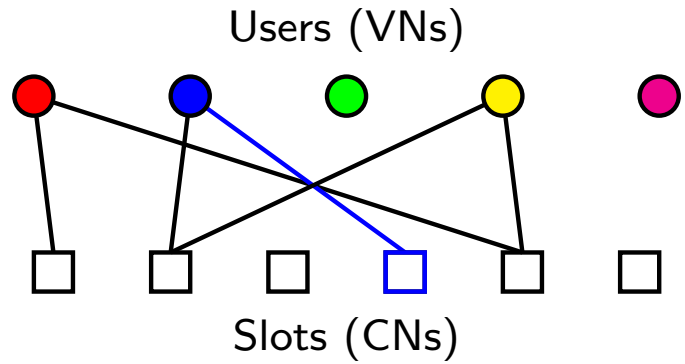




Coded slotted ALOHA: Decoding

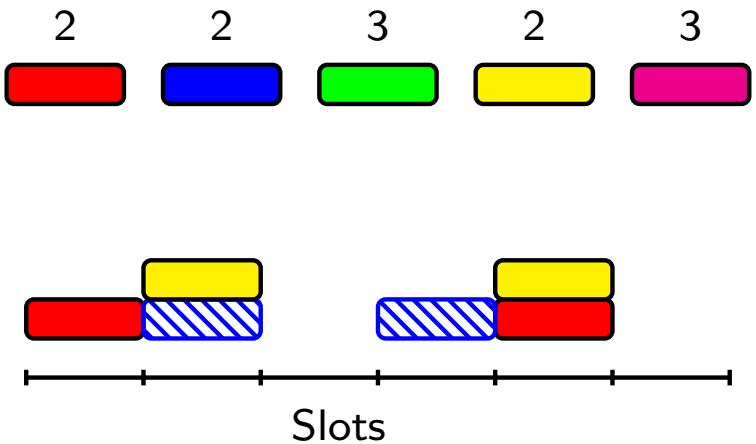
- A packet can be reliably decoded if no collision occurs
- Each packet contains pointers to its replicas
- The replicas of a decoded packet can be perfectly removed from the signal

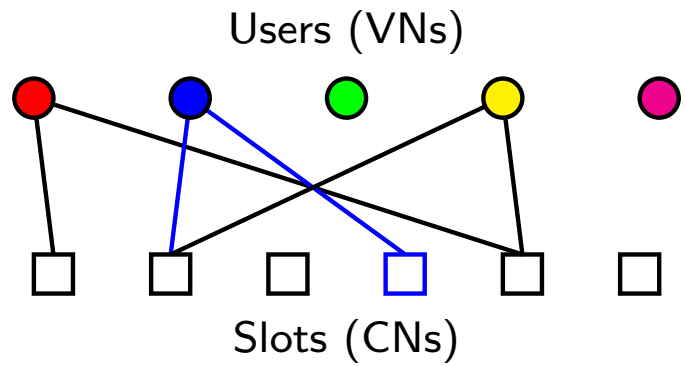




Coded slotted ALOHA: Decoding

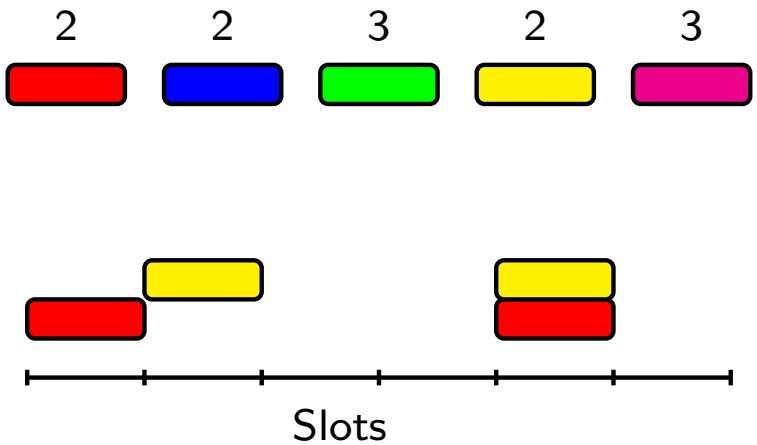
- A packet can be reliably decoded if no collision occurs
- Each packet contains pointers to its replicas
- The replicas of a decoded packet can be perfectly removed from the signal

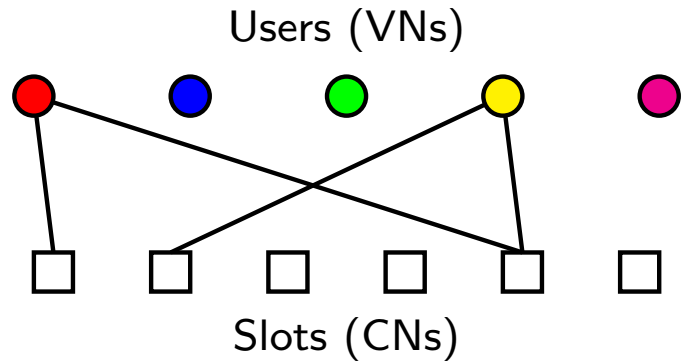




Coded slotted ALOHA: Decoding

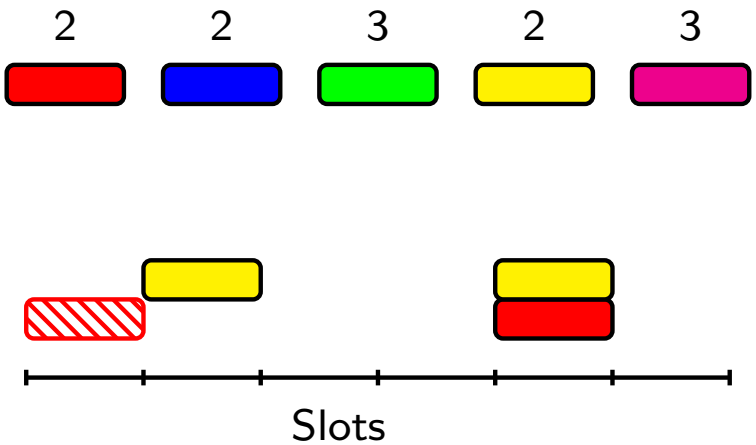
- A packet can be reliably decoded if no collision occurs
- Each packet contains pointers to its replicas
- The replicas of a decoded packet can be perfectly removed from the signal

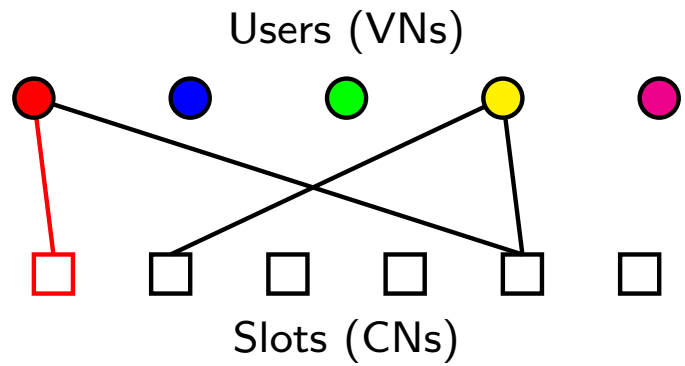




Coded slotted ALOHA: Decoding

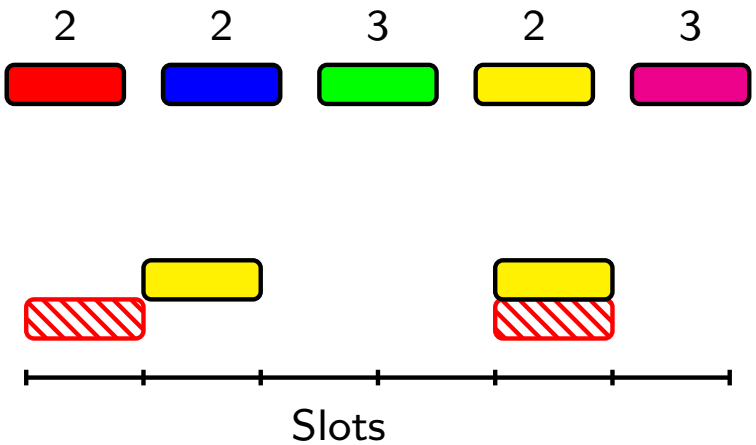
- A packet can be reliably decoded if no collision occurs
- Each packet contains pointers to its replicas
- The replicas of a decoded packet can be perfectly removed from the signal

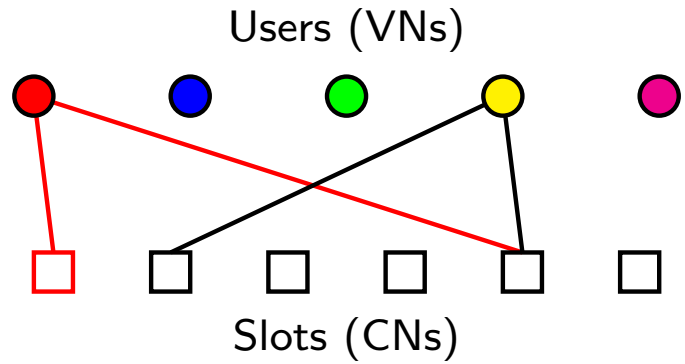




Coded slotted ALOHA: Decoding

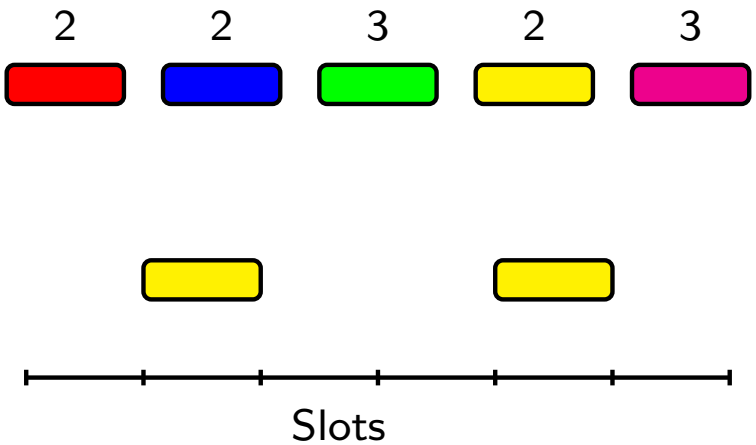
- A packet can be reliably decoded if no collision occurs
- Each packet contains pointers to its replicas
- The replicas of a decoded packet can be perfectly removed from the signal

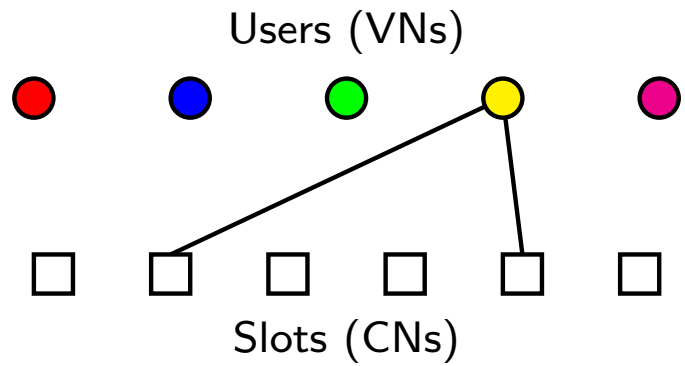




Coded slotted ALOHA: Decoding

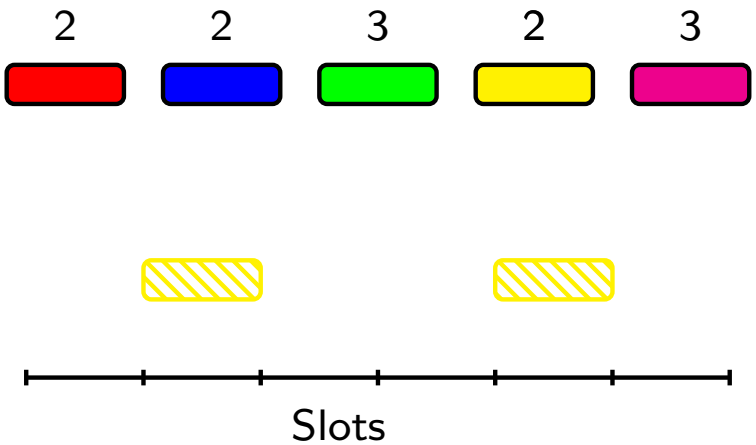
- A packet can be reliably decoded if no collision occurs
- Each packet contains pointers to its replicas
- The replicas of a decoded packet can be perfectly removed from the signal

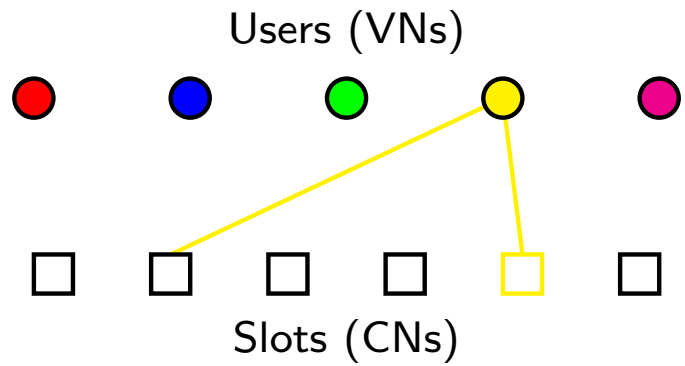




Coded slotted ALOHA: Decoding

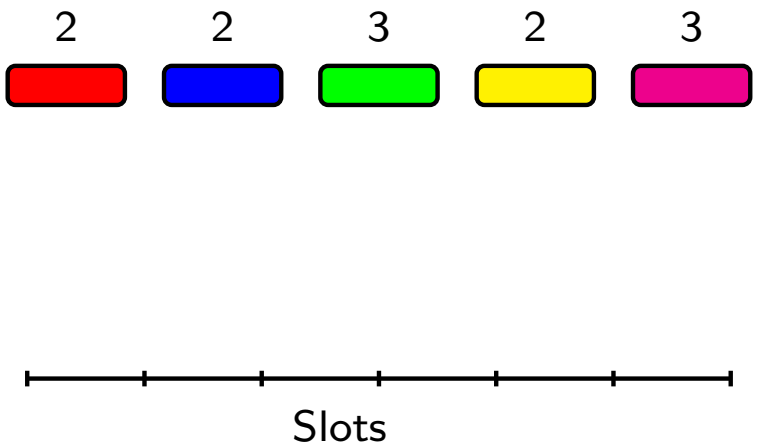
- A packet can be reliably decoded if no collision occurs
- Each packet contains pointers to its replicas
- The replicas of a decoded packet can be perfectly removed from the signal

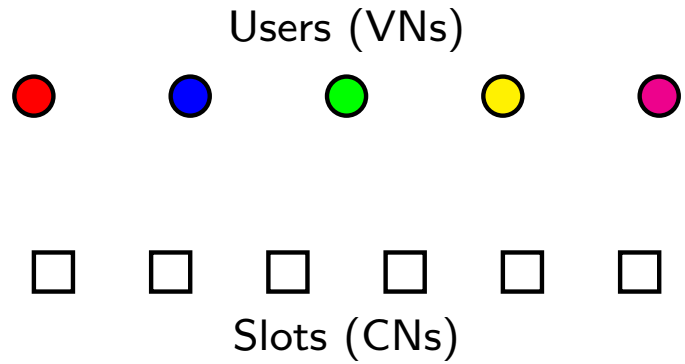




Coded slotted ALOHA: Decoding

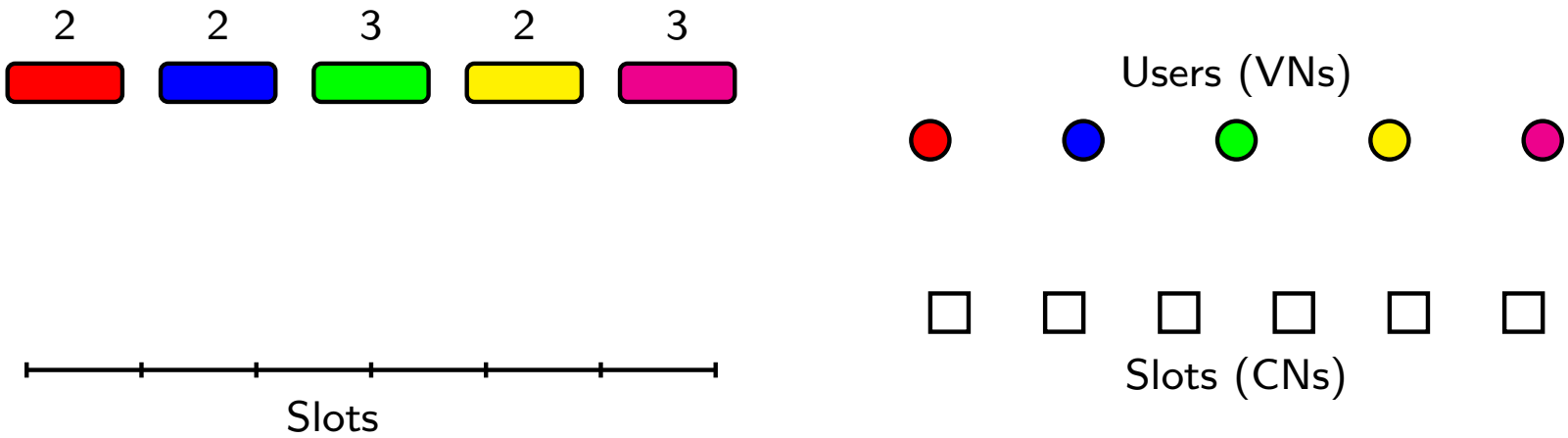
- A packet can be reliably decoded if no collision occurs
- Each packet contains pointers to its replicas
- The replicas of a decoded packet can be perfectly removed from the signal





Coded slotted ALOHA: Decoding

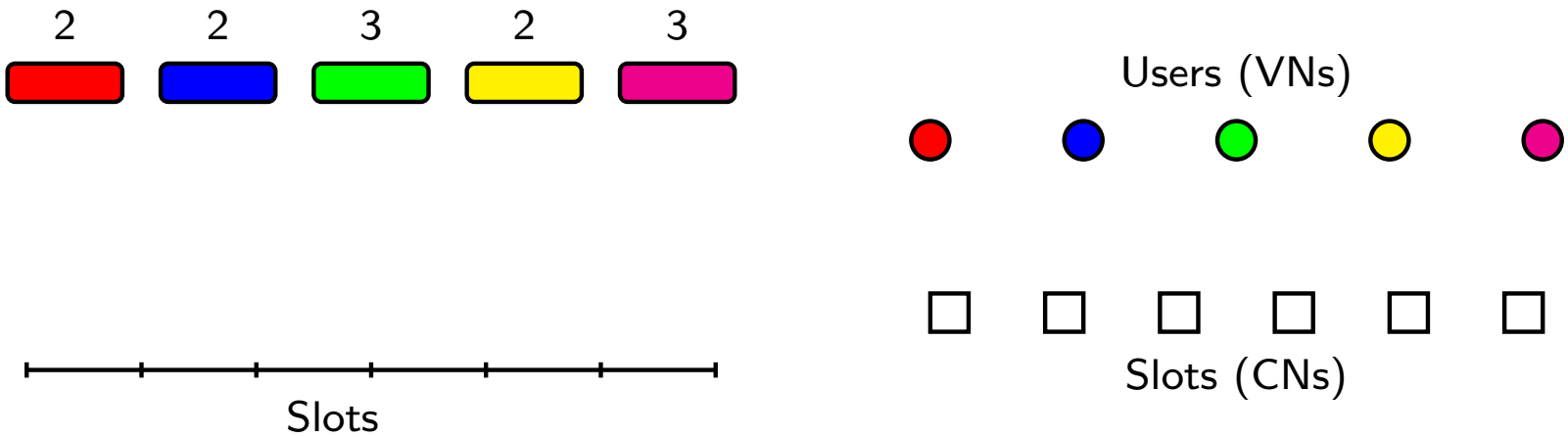
- A packet can be reliably decoded if no collision occurs
- Each packet contains pointers to its replicas
- The replicas of a decoded packet can be perfectly removed from the signal



- Decoding of CSA is equivalent to peeling (BP) decoding on a bipartite graph!

Coded slotted ALOHA: Decoding

- A packet can be reliably decoded if no collision occurs
- Each packet contains pointers to its replicas
- The replicas of a decoded packet can be perfectly removed from the signal



- Decoding of CSA is equivalent to peeling (BP) decoding on a bipartite graph!
- It can be analyzed using the theory of codes-on-graphs!

Public-key cryptosystems (RSA)

- Involves a pair of keys: a public key and a private key.

Public-key cryptosystems (RSA)

- Involves a pair of keys: a public key and a private key.
- Security based on the hardness of factoring large numbers.

Public-key cryptosystems (RSA)

- Involves a pair of keys: a public key and a private key.
- Security based on the hardness of factoring large numbers.

RSA [Rivest, Shamir, Adleman '77]:

Public-key cryptosystems (RSA)

- Involves a pair of keys: a public key and a private key.
- Security based on the hardness of factoring large numbers.

RSA [Rivest, Shamir, Adleman '77]:

- Randomly choose two different prime numbers p and q.

Public-key cryptosystems (RSA)

- Involves a pair of keys: a public key and a private key.
- Security based on the hardness of factoring large numbers.

RSA [Rivest, Shamir, Adleman '77]:

- Randomly choose two different prime numbers p and q.
- Calculate $n = pq$.

Public-key cryptosystems (RSA)

- Involves a pair of keys: a public key and a private key.
- Security based on the hardness of factoring large numbers.

RSA [Rivest, Shamir, Adleman '77]:

- Randomly choose two different prime numbers p and q.
- Calculate $n = pq$.
- Choose an integer e such that is co-prime to $\phi(n) = (p-1)(q-1)$.

Public-key cryptosystems (RSA)

- Involves a pair of keys: a public key and a private key.
- Security based on the hardness of factoring large numbers.

RSA [Rivest, Shamir, Adleman '77]:

- Randomly choose two different prime numbers p and q.
- Calculate $n = pq$.
- Choose an integer e such that is co-prime to $\phi(n) = (p-1)(q-1)$.
- Compute d that satisfies $de \equiv 1 (\text{mod} \phi(n))$.

Public-key cryptosystems (RSA)

- Involves a pair of keys: a public key and a private key.
- Security based on the hardness of factoring large numbers.

RSA [Rivest, Shamir, Adleman '77]:

- Randomly choose two different prime numbers p and q.
- Calculate $n = pq$.
- Choose an integer e such that is co-prime to $\phi(n) = (p-1)(q-1)$.
- Compute d that satisfies $de \equiv 1 (\text{mod} \phi(n))$.
- (n, e) is the public key, d is the private key.

Encrypting/Decrypting the message and attack

- Alice turns the message into a number $u < n$ and encrypts u to $c = u^e \pmod{n}$.

Encrypting/Decrypting the message and attack

- Alice turns the message into a number $u < n$ and encrypts u to $c = u^e \pmod{n}$.
- By using d, Bob can recover the message u, $c^d = (u^e)^d \equiv u \pmod{n}$.

Encrypting/Decrypting the message and attack

- Alice turns the message into a number $u < n$ and encrypts u to $c = u^e \pmod{n}$.
- By using d, Bob can recover the message u, $c^d = (u^e)^d \equiv u \pmod{n}$.
- To recover d, an attacker needs to factor $n \rightarrow$ Very time consuming!

Encrypting/Decrypting the message and attack

- Alice turns the message into a number $u < n$ and encrypts u to $c = u^e \pmod{n}$.
- By using d, Bob can recover the message u, $c^d = (u^e)^d \equiv u \pmod{n}$.
- To recover d, an attacker needs to factor $n \rightarrow$ Very time consuming!
- But...a quantum computer can factor n in polynomial time!

Encrypting/Decrypting the message and attack

- Alice turns the message into a number $u < n$ and encrypts u to $c = u^e \pmod{n}$.
- By using d, Bob can recover the message u, $c^d = (u^e)^d \equiv u \pmod{n}$.
- To recover d, an attacker needs to factor $n \rightarrow$ Very time consuming!
- But...a quantum computer can factor n in polynomial time!
- We need to find cryptosystems that are robust against the attack by a quantum computer

Encrypting/Decrypting the message and attack

- Alice turns the message into a number $u < n$ and encrypts u to $c = u^e \pmod{n}$.
- By using d, Bob can recover the message u, $c^d = (u^e)^d \equiv u \pmod{n}$.
- To recover d, an attacker needs to factor $n \rightarrow$ Very time consuming!
- But...a quantum computer can factor n in polynomial time!
- We need to find cryptosystems that are robust against the attack by a quantum computer $\rightarrow$ Post-quantum cryptography

Code-based cryptography: The McEliece cryptosystem

- A public-key cryptosystem based on coding theory [McEliece '78].

Code-based cryptography: The McEliece cryptosystem

- A public-key cryptosystem based on coding theory [McEliece '78].
- Decoding general linear codes is NP-complete

Code-based cryptography: The McEliece cryptosystem

- A public-key cryptosystem based on coding theory [McEliece '78].
- Decoding general linear codes is NP-complete $\rightarrow$ Robust against the attack of a quantum computer!

Code-based cryptography: The McEliece cryptosystem

- A public-key cryptosystem based on coding theory [McEliece '78].
- Decoding general linear codes is NP-complete $\rightarrow$ Robust against the attack of a quantum computer!

Principle:

Code-based cryptography: The McEliece cryptosystem

- A public-key cryptosystem based on coding theory [McEliece '78].
- Decoding general linear codes is NP-complete $\rightarrow$ Robust against the attack of a quantum computer!

Principle:

- Start with a code $\mathcal{C}$ for with generator matrix $\boldsymbol{G}$ and efficient decoding capable of error correcting capability t.

Code-based cryptography: The McEliece cryptosystem

- A public-key cryptosystem based on coding theory [McEliece '78].
- Decoding general linear codes is NP-complete $\rightarrow$ Robust against the attack of a quantum computer!

Principle:

- Start with a code $\mathcal{C}$ for with generator matrix $\boldsymbol{G}$ and efficient decoding capable of error correcting capability t.
- Choose a nonsingular $k \times k$ matrix $\boldsymbol{S}$ and an $n \times n$ permutation matrix $\boldsymbol{P}$, and generate

$$\boldsymbol{G}' = \boldsymbol{SGP}$$

Code-based cryptography: The McEliece cryptosystem

- A public-key cryptosystem based on coding theory [McEliece '78].
- Decoding general linear codes is NP-complete $\rightarrow$ Robust against the attack of a quantum computer!

Principle:

- Start with a code $\mathcal{C}$ for with generator matrix $\boldsymbol{G}$ and efficient decoding capable of error correcting capability t.
- Choose a nonsingular $k \times k$ matrix $\boldsymbol{S}$ and an $n \times n$ permutation matrix $\boldsymbol{P}$, and generate

$$\boldsymbol{G}' = \boldsymbol{SGP}$$

- $\boldsymbol{G}'$ is the public key, and $(\boldsymbol{G}, \boldsymbol{S}, \boldsymbol{P})$ is the secret key.

Code-based cryptography: The McEliece cryptosystem

- A public-key cryptosystem based on coding theory [McEliece '78].
- Decoding general linear codes is NP-complete $\rightarrow$ Robust against the attack of a quantum computer!

Principle:

- Start with a code $\mathcal{C}$ for with generator matrix $\boldsymbol{G}$ and efficient decoding capable of error correcting capability t.
- Choose a nonsingular $k \times k$ matrix $\boldsymbol{S}$ and an $n \times n$ permutation matrix $\boldsymbol{P}$, and generate

$$\boldsymbol{G}' = \boldsymbol{SGP}$$

- $\boldsymbol{G}'$ is the public key, and $(\boldsymbol{G}, \boldsymbol{S}, \boldsymbol{P})$ is the secret key.
- The new matrix $\boldsymbol{G}'$ is the generator matrix of another linear code, that is very difficult to decode if the secret key is not known.

The McEliece cryptosystem: Encoding and decoding

The McEliece cryptosystem: Encoding and decoding

- Alice generates the cyphertext $\boldsymbol{c}$ is obtained by encoding k-bit message $\boldsymbol{u}$ by $\boldsymbol{G}$ and adding a random error vector $\boldsymbol{e}$ of Hamming weight t,

$$\boldsymbol{c} = \boldsymbol{m}\boldsymbol{G}' + \boldsymbol{e}.$$

The McEliece cryptosystem: Encoding and decoding

- Alice generates the cyphertext $\boldsymbol{c}$ is obtained by encoding k-bit message $\boldsymbol{u}$ by $\boldsymbol{G}$ and adding a random error vector $\boldsymbol{e}$ of Hamming weight t,

$$\boldsymbol{c} = \boldsymbol{m}\boldsymbol{G}' + \boldsymbol{e}.$$

- After receiving $\boldsymbol{c}$, Bob inverts the transformation as

$$\boldsymbol{c}' = \boldsymbol{c}\boldsymbol{P}^{-1} = \boldsymbol{u}\boldsymbol{S}\boldsymbol{G} + \boldsymbol{e}\boldsymbol{P}^{-1},$$

The McEliece cryptosystem: Encoding and decoding

- Alice generates the cyphertext $\boldsymbol{c}$ is obtained by encoding k-bit message $\boldsymbol{u}$ by $\boldsymbol{G}$ and adding a random error vector $\boldsymbol{e}$ of Hamming weight t,

$$\boldsymbol{c} = \boldsymbol{m}\boldsymbol{G}' + \boldsymbol{e}.$$

- After receiving $\boldsymbol{c}$, Bob inverts the transformation as

$$\boldsymbol{c}' = \boldsymbol{c}\boldsymbol{P}^{-1} = \boldsymbol{u}\boldsymbol{S}\boldsymbol{G} + \boldsymbol{e}\boldsymbol{P}^{-1},$$

  thus obtaining a codeword of the secret code generated by $\boldsymbol{G}$ affected by the error vector $\boldsymbol{e}\boldsymbol{P}^{-1}$.

The McEliece cryptosystem: Encoding and decoding

- Alice generates the cyphertext $\boldsymbol{c}$ is obtained by encoding k-bit message $\boldsymbol{u}$ by $\boldsymbol{G}$ and adding a random error vector $\boldsymbol{e}$ of Hamming weight t,

$$\boldsymbol{c} = \boldsymbol{m}\boldsymbol{G}' + \boldsymbol{e}.$$

- After receiving $\boldsymbol{c}$, Bob inverts the transformation as

$$\boldsymbol{c}' = \boldsymbol{c}\boldsymbol{P}^{-1} = \boldsymbol{u}\boldsymbol{S}\boldsymbol{G} + \boldsymbol{e}\boldsymbol{P}^{-1},$$

  thus obtaining a codeword of the secret code generated by $\boldsymbol{G}$ affected by the error vector $\boldsymbol{e}\boldsymbol{P}^{-1}$.

- Bob decodes the codeword $\boldsymbol{c}'$ to obtain $\boldsymbol{u}\boldsymbol{S}$, and finally computes $\boldsymbol{u} = \boldsymbol{u}\boldsymbol{S}^{-1}$.

The McEliece cryptosystem: Encoding and decoding

- Alice generates the cyphertext $\boldsymbol{c}$ is obtained by encoding k-bit message $\boldsymbol{u}$ by $\boldsymbol{G}$ and adding a random error vector $\boldsymbol{e}$ of Hamming weight t,

$$\boldsymbol{c} = \boldsymbol{m}\boldsymbol{G}' + \boldsymbol{e}.$$

- After receiving $\boldsymbol{c}$, Bob inverts the transformation as

$$\boldsymbol{c}' = \boldsymbol{c}\boldsymbol{P}^{-1} = \boldsymbol{u}\boldsymbol{S}\boldsymbol{G} + \boldsymbol{e}\boldsymbol{P}^{-1},$$

  thus obtaining a codeword of the secret code generated by $\boldsymbol{G}$ affected by the error vector $\boldsymbol{e}\boldsymbol{P}^{-1}$.

- Bob decodes the codeword $\boldsymbol{c}'$ to obtain $\boldsymbol{u}\boldsymbol{S}$, and finally computes $\boldsymbol{u} = \boldsymbol{u}\boldsymbol{S}^{-1}$.

Problem

Very big key sizes (hundreds of thousands of bits, compared to few thousands for RSA)

Code-based cryptography: Current research

Code-based cryptography: Current research

- Idea: Use modern codes!

Code-based cryptography: Current research

- Idea: Use modern codes!
- Quasi-cyclic medium-density parity-check (MDPC) codes

Code-based cryptography: Current research

- Idea: Use modern codes!
- Quasi-cyclic medium-density parity-check (MDPC) codes
  - Efficient decoding.

Code-based cryptography: Current research

- Idea: Use modern codes!
- Quasi-cyclic medium-density parity-check (MDPC) codes
  - Efficient decoding.
  - Much shorter key sizes.

DNA storage: Using DNA to store data

A unique storage medium:

DNA storage: Using DNA to store data

A unique storage medium:

- Outstanding integrity: can still recover the DNA of species extinct for more than 10000 years.

DNA storage: Using DNA to store data

A unique storage medium:

- Outstanding integrity: can still recover the DNA of species extinct for more than 10000 years.
- Extremely high storage capacity: a human cell, with a mass of roughly 3 pgrams, hosts DNA encoding 6.4 GB of information.

DNA storage: Using DNA to store data

A unique storage medium:

- Outstanding integrity: can still recover the DNA of species extinct for more than 10000 years.
- Extremely high storage capacity: a human cell, with a mass of roughly 3 pgrams, hosts DNA encoding 6.4 GB of information.

- Archival DNA-based storage already feasible!

DNA storage: Main principle

- The DNA strands are composed of units called nucleotides.

DNA storage: Main principle

- The DNA strands are composed of units called nucleotides.
- Four DNA base nucleotides: Adenine (A), Cyanine (C), Guanine (G), and Thymine (T).

DNA storage: Main principle

- The DNA strands are composed of units called nucleotides.
- Four DNA base nucleotides: Adenine (A), Cyanine (C), Guanine (G), and Thymine (T).
- It's possible to synthesize (almost) arbitrary chain molecules by concatenating the DNA base nucleotides. (Modern synthesizers: a few hundreds of nucleotides)

DNA storage: Main principle

- The DNA strands are composed of units called nucleotides.
- Four DNA base nucleotides: Adenine (A), Cyanine (C), Guanine (G), and Thymine (T).
- It's possible to synthesize (almost) arbitrary chain molecules by concatenating the DNA base nucleotides. (Modern synthesizers: a few hundreds of nucleotides)
- Each nucleotide (called oligo) can be considered to represent 2 bits $\rightarrow$ Data can be stored in oligos!

DNA storage: Main principle

- The DNA strands are composed of units called nucleotides.
- Four DNA base nucleotides: Adenine (A), Cyanine (C), Guanine (G), and Thymine (T).
- It's possible to synthesize (almost) arbitrary chain molecules by concatenating the DNA base nucleotides. (Modern synthesizers: a few hundreds of nucleotides)
- Each nucleotide (called oligo) can be considered to represent 2 bits $\rightarrow$ Data can be stored in oligos!
- Data can be recovered reading the sequence of DNA by a DNA sequencer.

DNA storage: Main principle

- The DNA strands are composed of units called nucleotides.
- Four DNA base nucleotides: Adenine (A), Cyanine (C), Guanine (G), and Thymine (T).
- It's possible to synthesize (almost) arbitrary chain molecules by concatenating the DNA base nucleotides. (Modern synthesizers: a few hundreds of nucleotides)
- Each nucleotide (called oligo) can be considered to represent 2 bits $\rightarrow$ Data can be stored in oligos!
- Data can be recovered reading the sequence of DNA by a DNA sequencer.
- Once desired information is stored in DNA, it may be rewritten using DNA editing techniques.

DNA storage

Problem:

DNA storage

Problem:

- Errors occur during the writing (synthesizing) and reading (sequencing).

DNA storage

Problem:

- Errors occur during the writing (synthesizing) and reading (sequencing).
- Synthesis introduces substitution, insertion, and deletion errors.

DNA storage

Problem:

- Errors occur during the writing (synthesizing) and reading (sequencing).
- Synthesis introduces substitution, insertion, and deletion errors.
- In particular, single insertions and deletions or substitution errors in the sequence often cannot be removed.

DNA storage

Problem:

- Errors occur during the writing (synthesizing) and reading (sequencing).
- Synthesis introduces substitution, insertion, and deletion errors.
- In particular, single insertions and deletions or substitution errors in the sequence often cannot be removed.
- Sequencing introduces further errors $\rightarrow$ Some DNA patterns are more prone to errors than others.

DNA storage

Problem:

- Errors occur during the writing (synthesizing) and reading (sequencing).
- Synthesis introduces substitution, insertion, and deletion errors.
- In particular, single insertions and deletions or substitution errors in the sequence often cannot be removed.
- Sequencing introduces further errors $\rightarrow$ Some DNA patterns are more prone to errors than others.

Solution:

DNA storage

Problem:

- Errors occur during the writing (synthesizing) and reading (sequencing).
- Synthesis introduces substitution, insertion, and deletion errors.
- In particular, single insertions and deletions or substitution errors in the sequence often cannot be removed.
- Sequencing introduces further errors $\rightarrow$ Some DNA patterns are more prone to errors than others.

Solution: Use coding!

DNA storage

Problem:

- Errors occur during the writing (synthesizing) and reading (sequencing).
- Synthesis introduces substitution, insertion, and deletion errors.
- In particular, single insertions and deletions or substitution errors in the sequence often cannot be removed.
- Sequencing introduces further errors $\rightarrow$ Some DNA patterns are more prone to errors than others.

Solution: Use coding!

- Coding to correct insertions, deletions and substitutions.

DNA storage

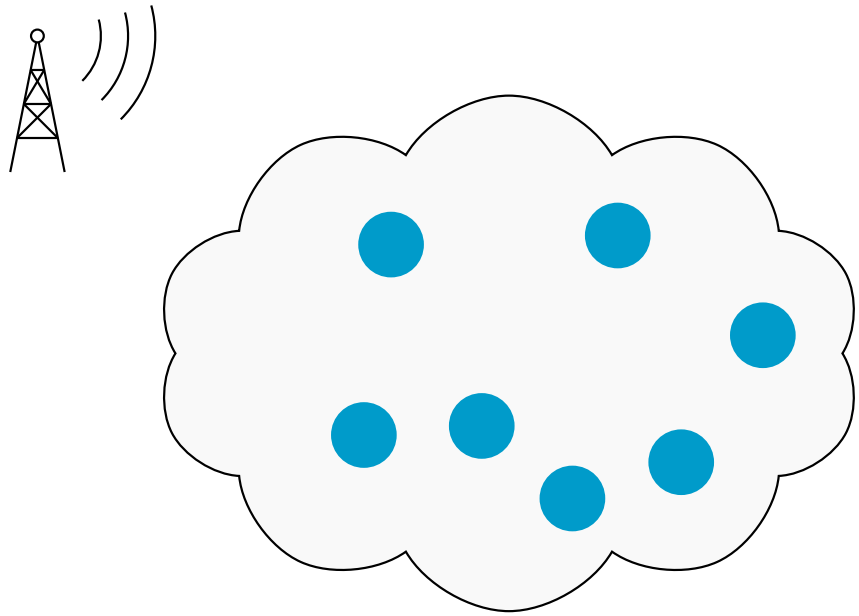
Problem:

- Errors occur during the writing (synthesizing) and reading (sequencing).
- Synthesis introduces substitution, insertion, and deletion errors.
- In particular, single insertions and deletions or substitution errors in the sequence often cannot be removed.
- Sequencing introduces further errors $\rightarrow$ Some DNA patterns are more prone to errors than others.

Solution: Use coding!

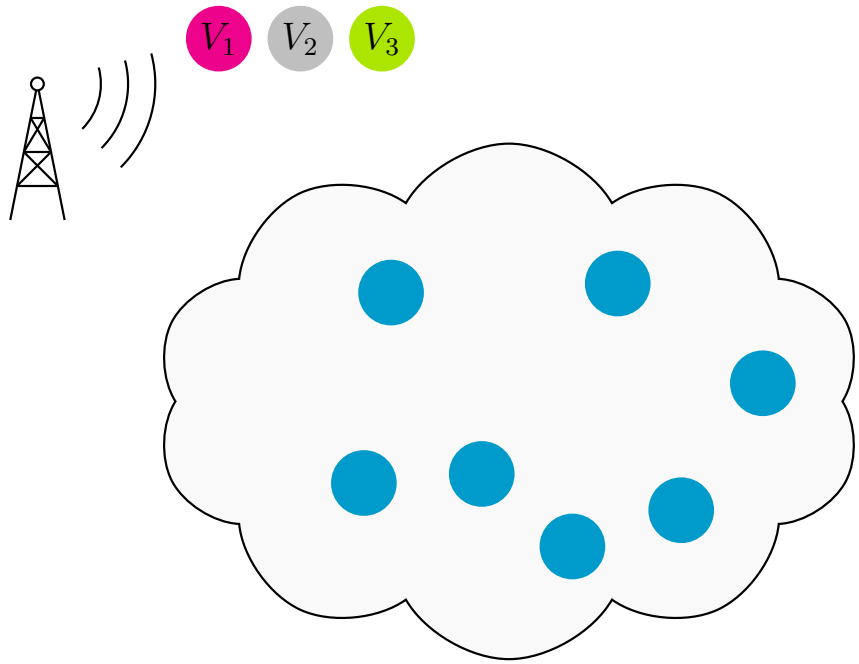
- Coding to correct insertions, deletions and substitutions.
- Constrained coding to avoid some DNA patterns.

Coding for content delivery: Distributed caching



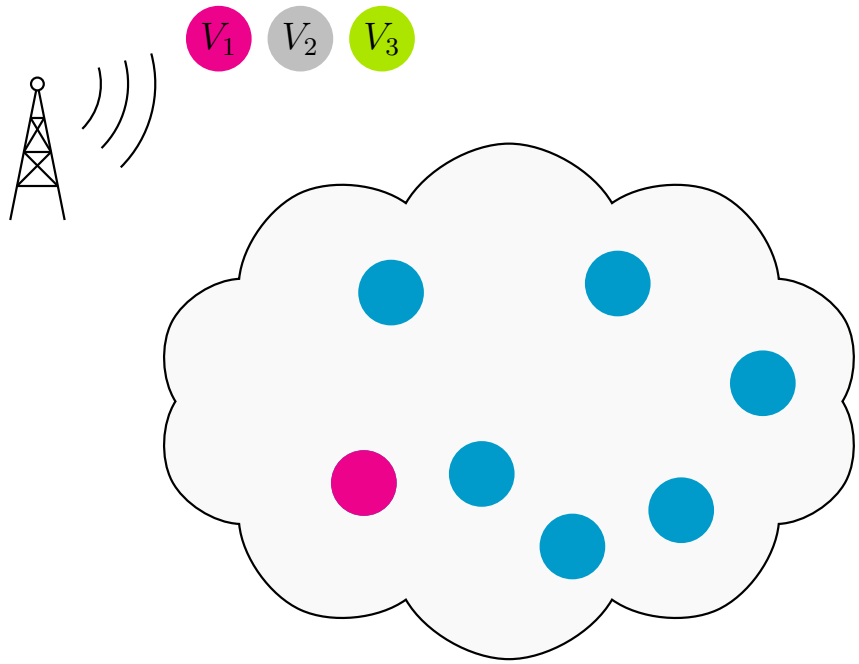
video on demand

Coding for content delivery: Distributed caching



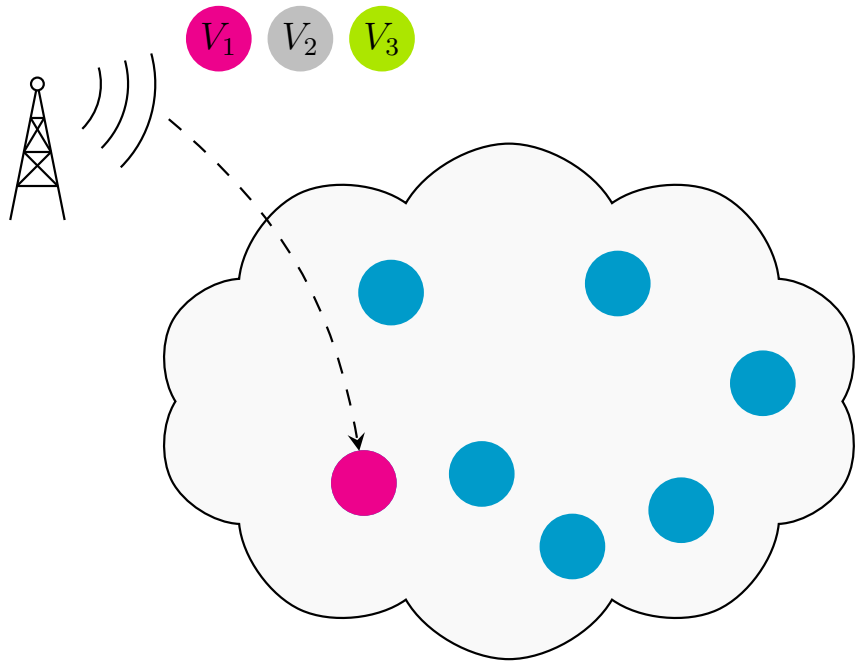
video on demand

Coding for content delivery: Distributed caching



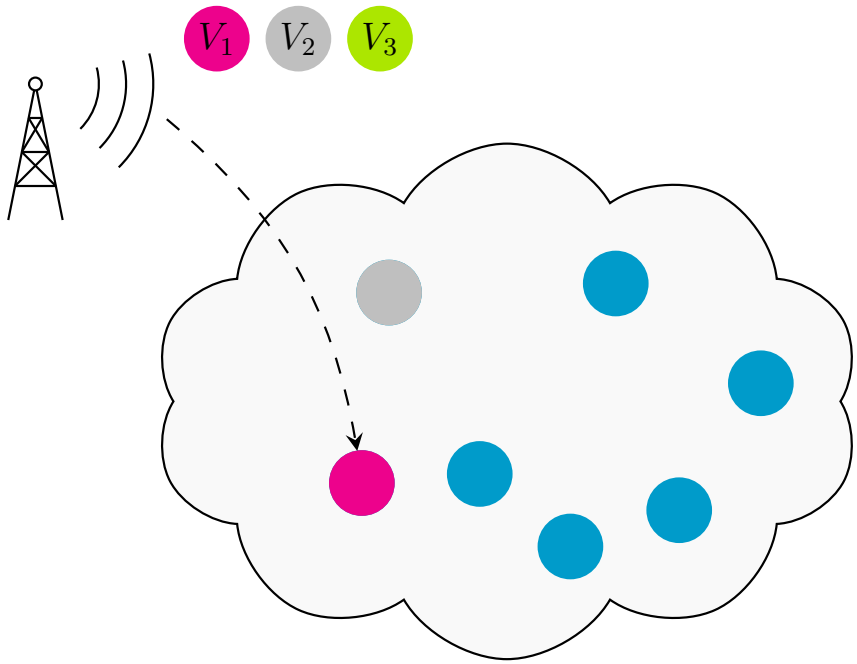
video on demand

Coding for content delivery: Distributed caching



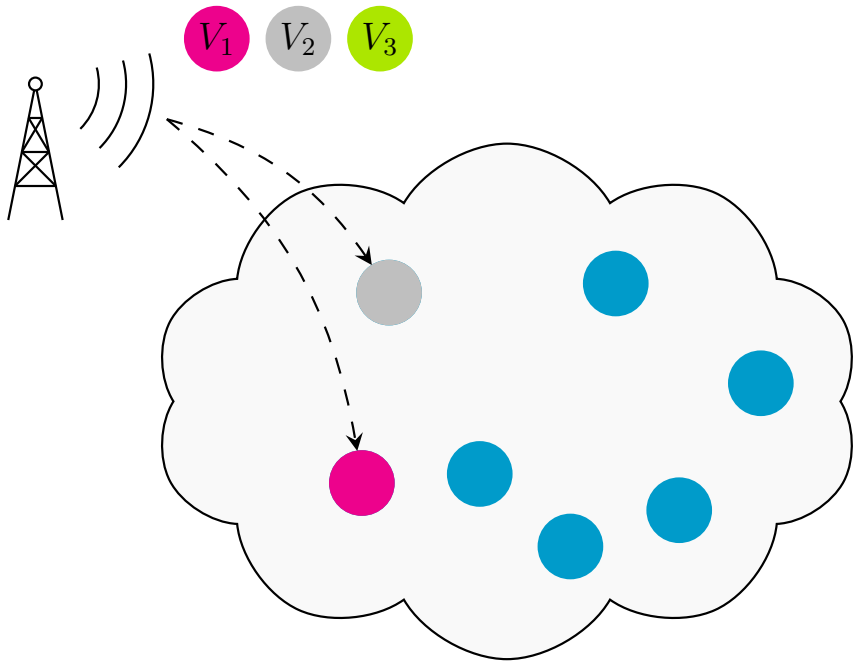
video on demand

Coding for content delivery: Distributed caching



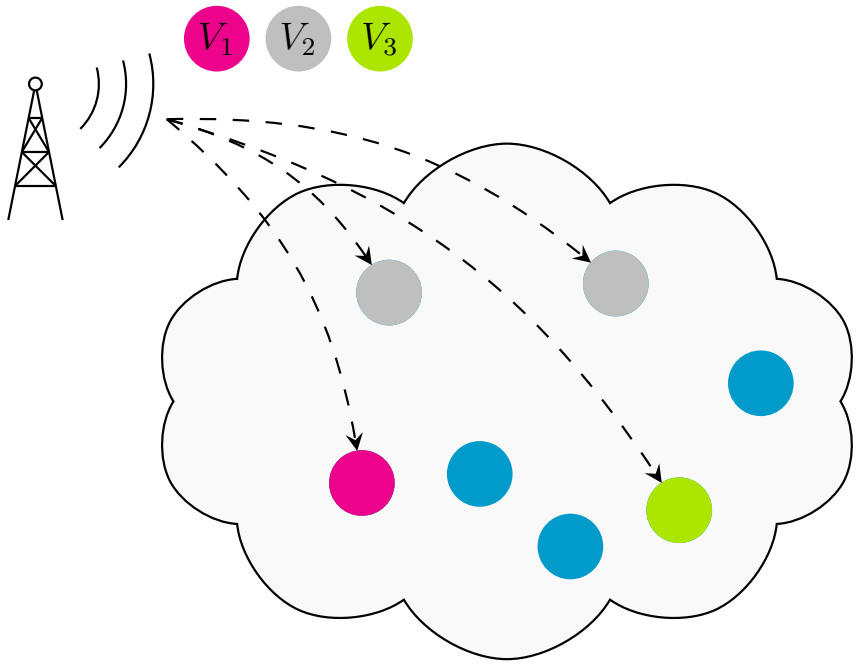
video on demand

Coding for content delivery: Distributed caching



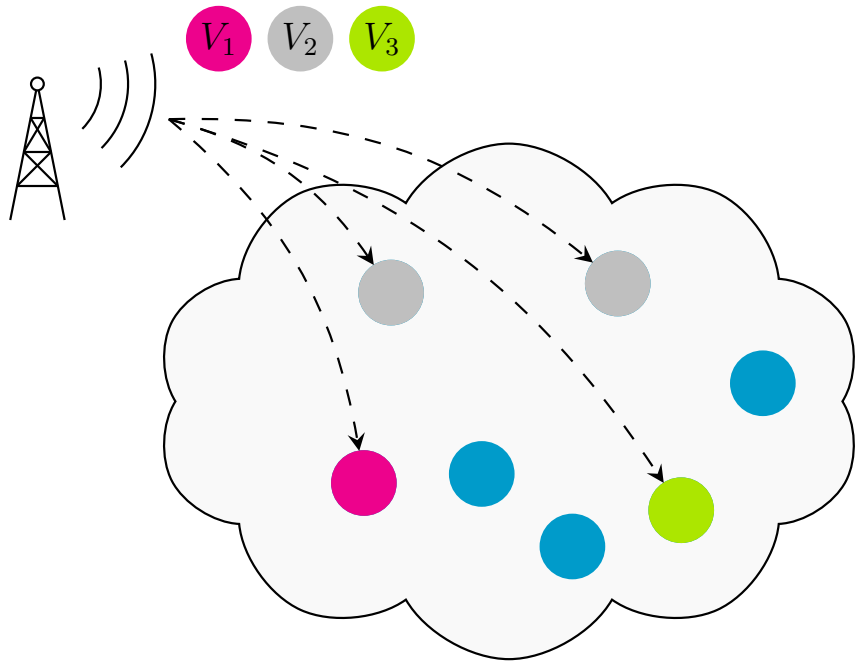
video on demand

Coding for content delivery: Distributed caching



video on demand

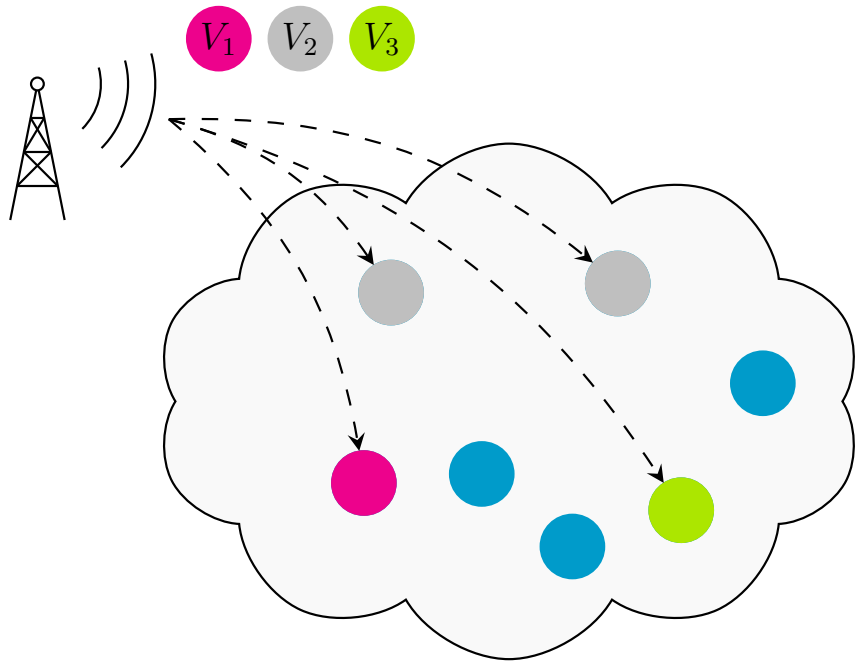
Coding for content delivery: Distributed caching



video on demand

- Mobile data traffic is expected to rise at a rate of 45% per year

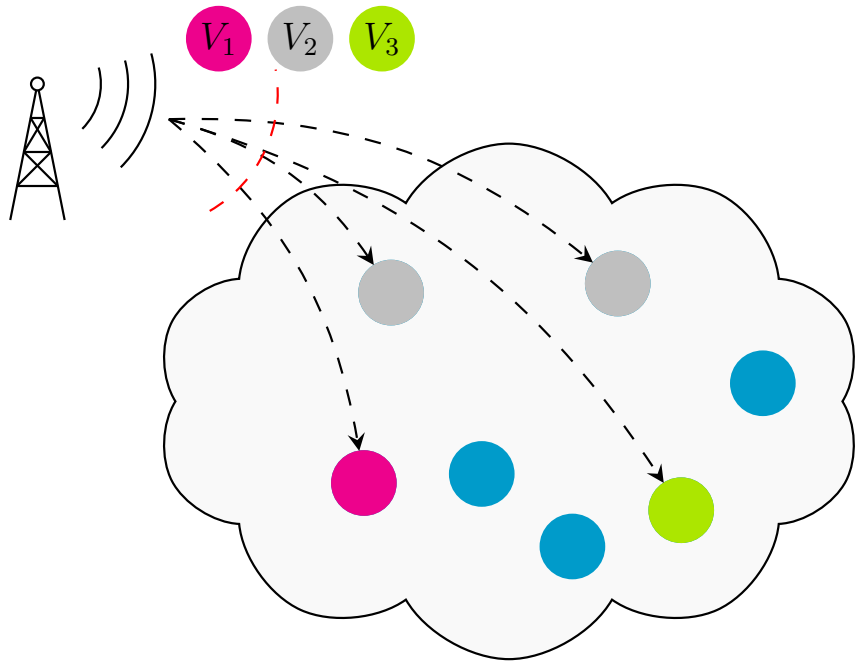
Coding for content delivery: Distributed caching



video on demand

- Mobile data traffic is expected to rise at a rate of 45% per year
- It is forecasted to attain 51 exabytes per month by 2021 [Ericsson]

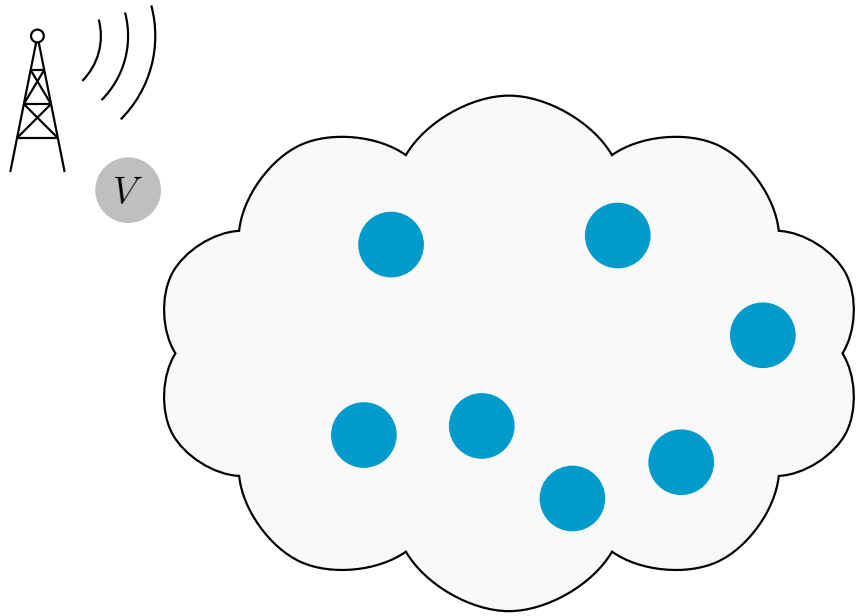
Coding for content delivery: Distributed caching



video on demand

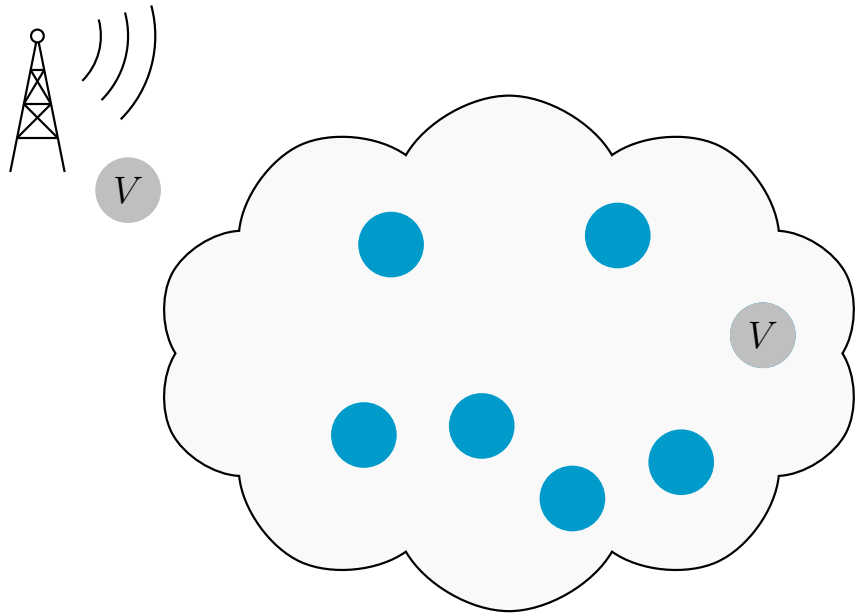
- Mobile data traffic is expected to rise at a rate of 45% per year
- It is forecasted to attain 51 exabytes per month by 2021 [Ericsson]

Coding for content delivery: Distributed caching



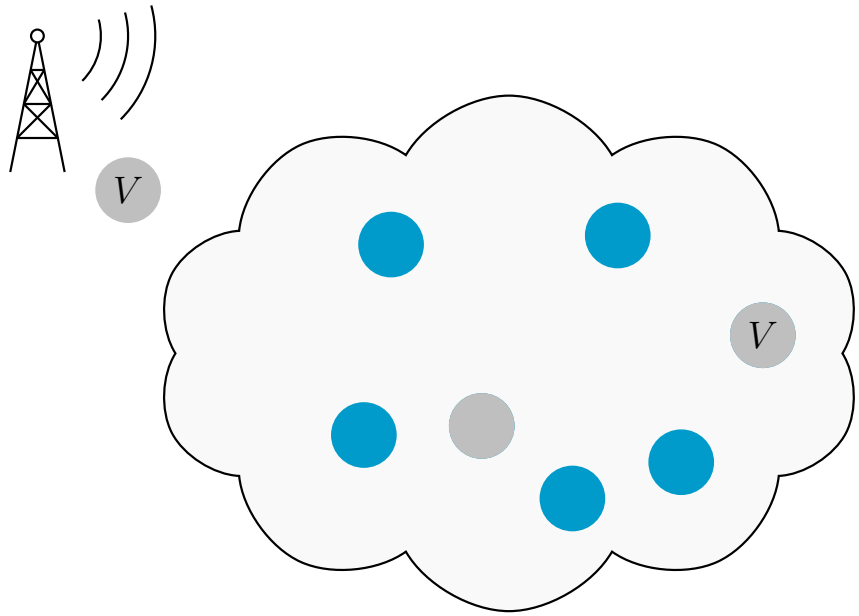
video on demand

Coding for content delivery: Distributed caching



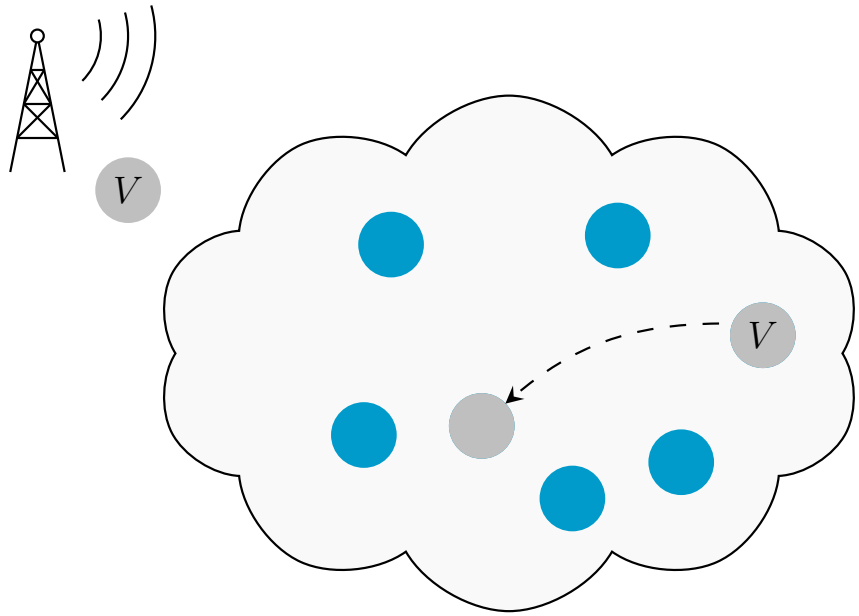
video on demand

Coding for content delivery: Distributed caching



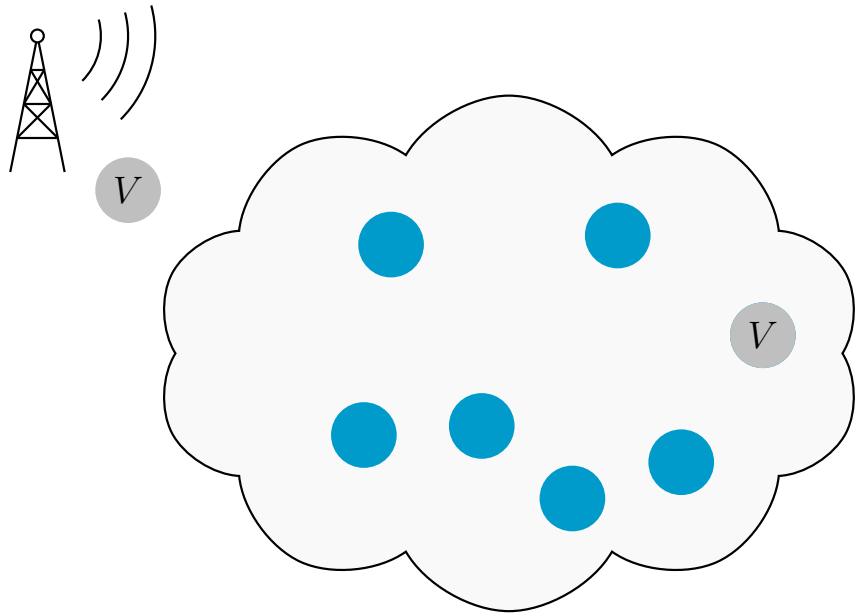
video on demand

Coding for content delivery: Distributed caching



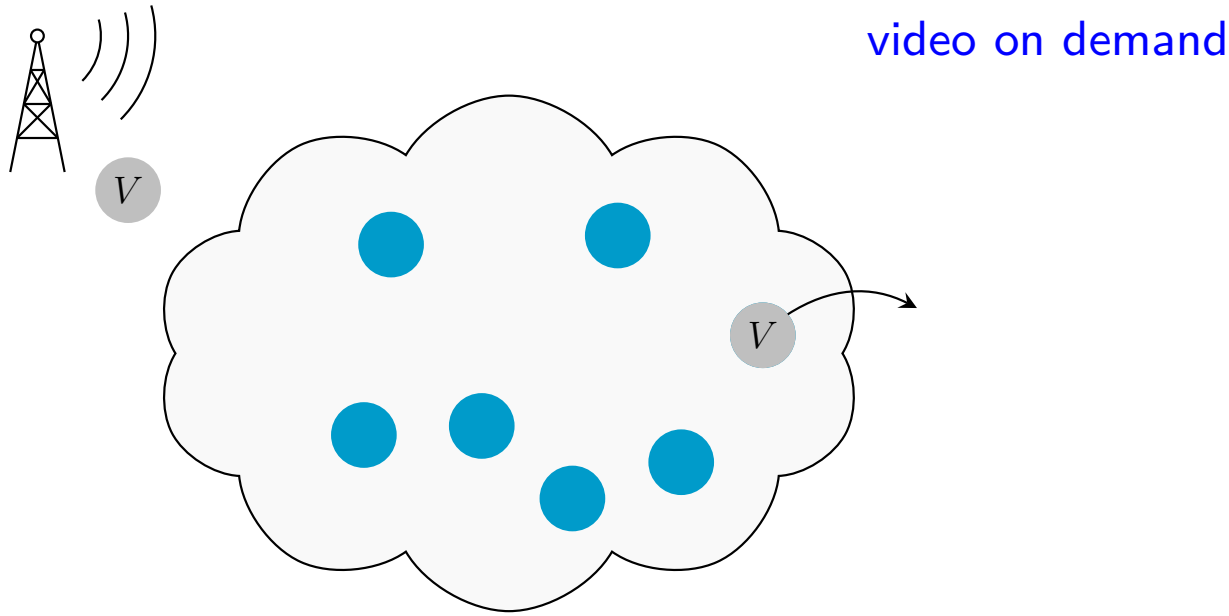
video on demand

Coding for content delivery: Distributed caching

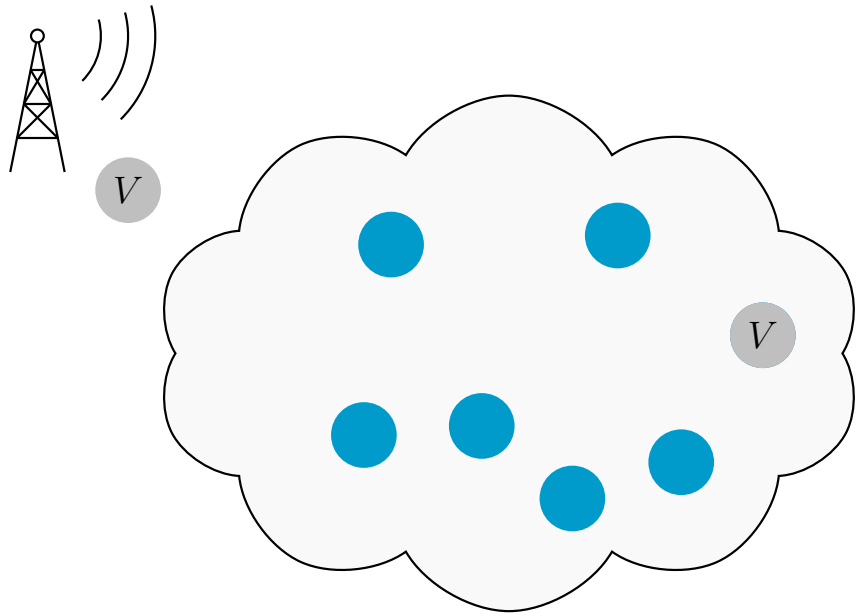


video on demand

Coding for content delivery: Distributed caching

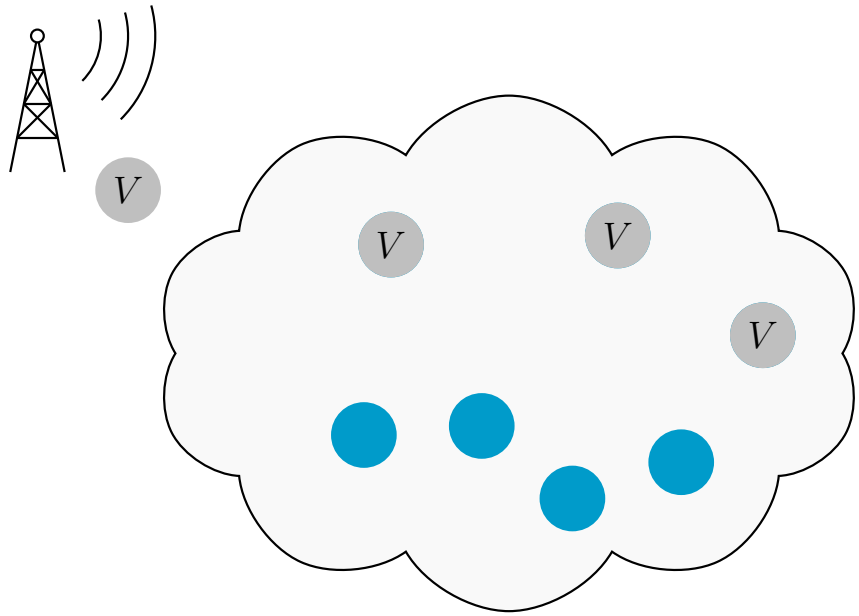


Coding for content delivery: Distributed caching



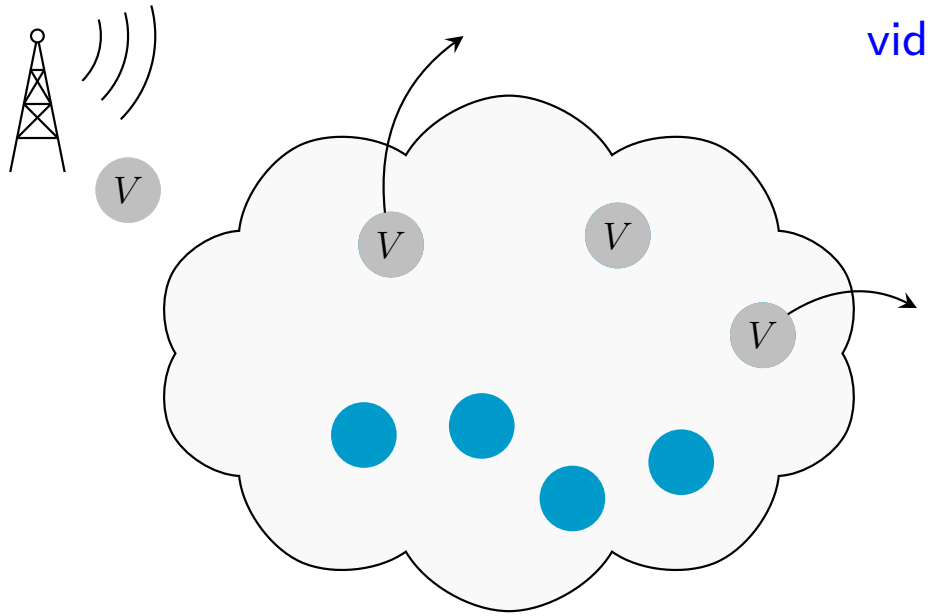
video on demand

Coding for content delivery: Distributed caching



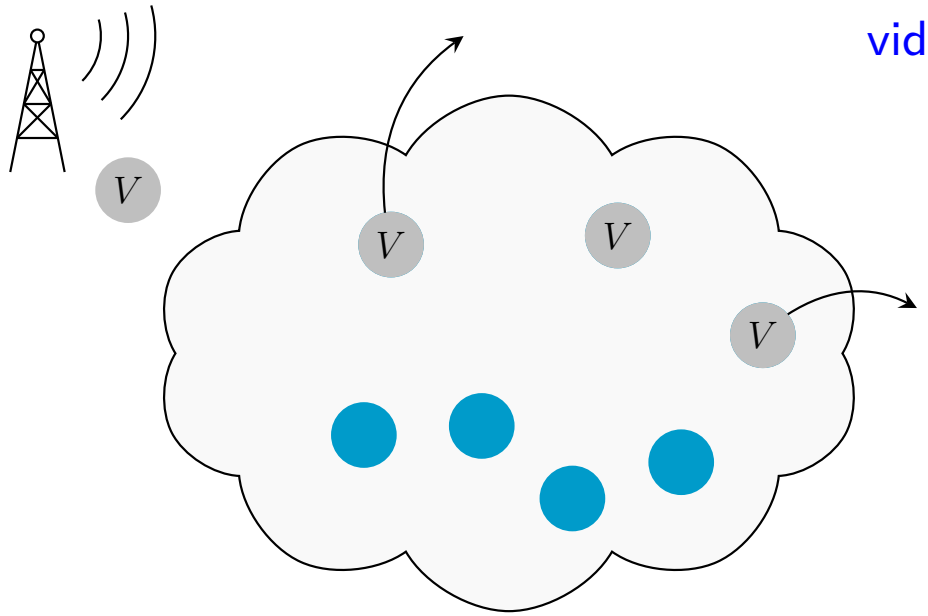
video on demand

Coding for content delivery: Distributed caching

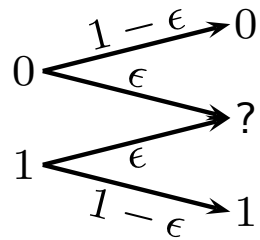


video on demand

Coding for content delivery: Distributed caching

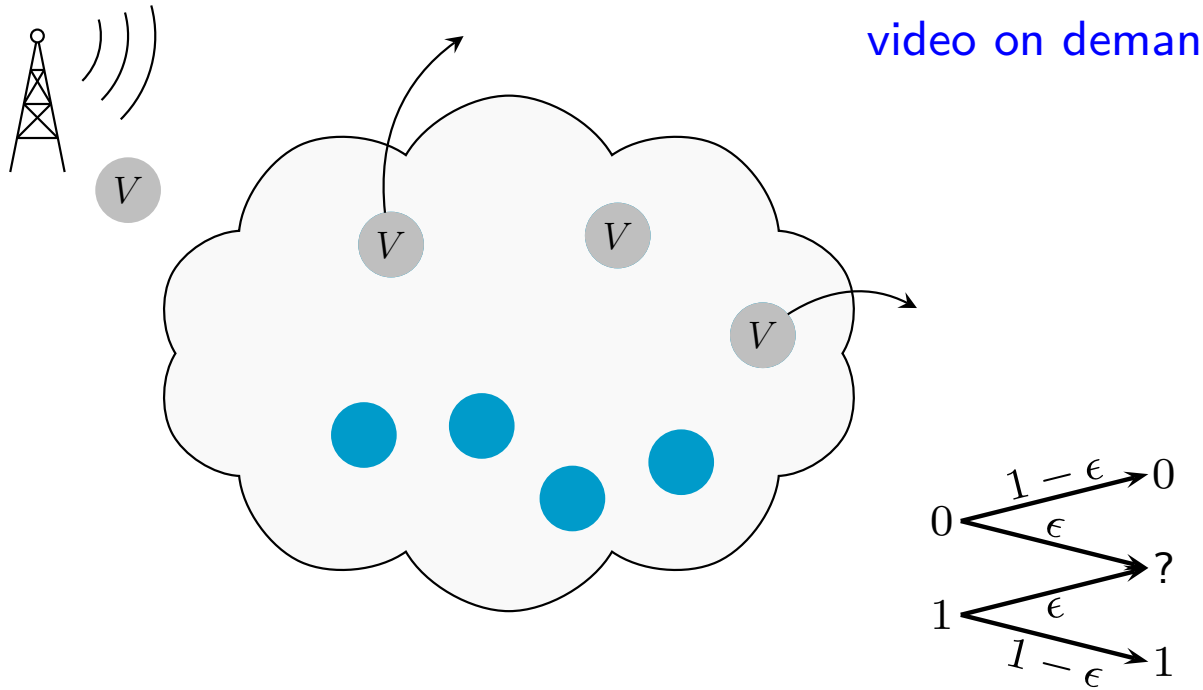


video on demand



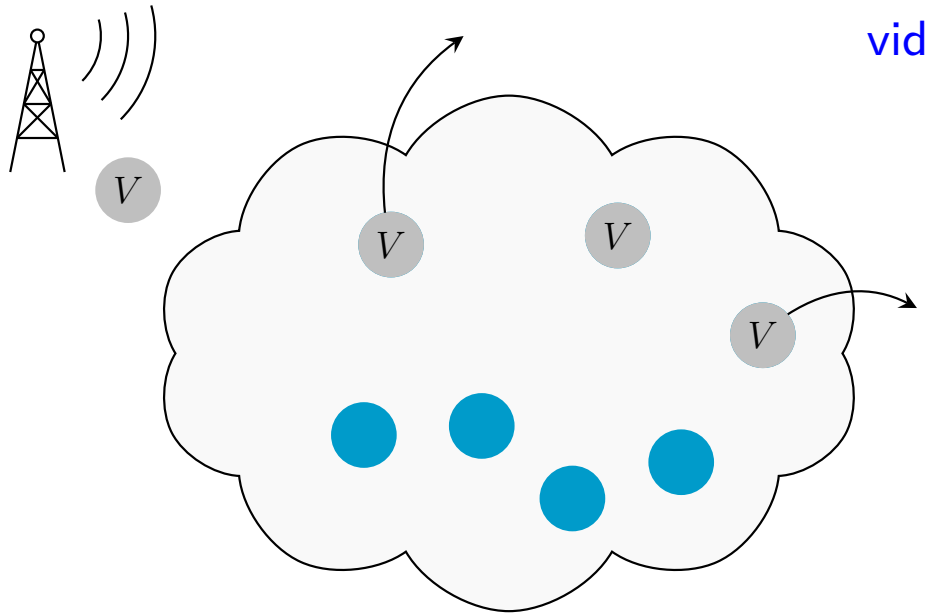
- Node departures can be seen as erasures in a BEC

Coding for content delivery: Distributed caching

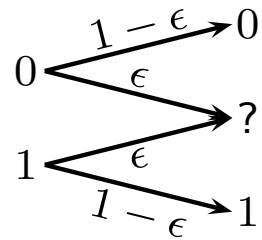


- Node departures can be seen as erasures in a BEC
- Replication is a trivial erasure correcting code

Coding for content delivery: Distributed caching

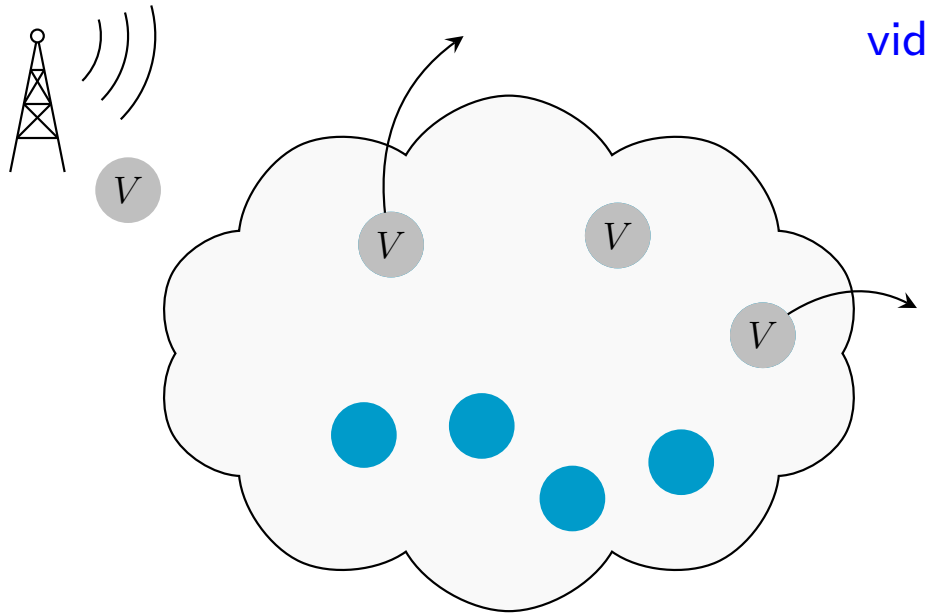


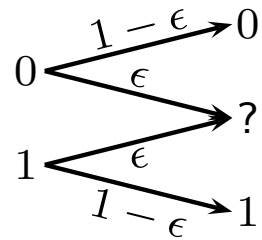
video on demand



- Node departures can be seen as erasures in a BEC
- Replication is a trivial erasure correcting code $\rightarrow$ distributed caching can be seen as a coding problem!

Coding for content delivery: Distributed caching





- Node departures can be seen as erasures in a BEC
- Replication is a trivial erasure correcting code $\rightarrow$ distributed caching can be seen as a coding problem!

Idea

Use erasure correcting codes to store content!

And many more applications...

And many more applications...

- Distributed computing
- Distributed storage
- Security and privacy
- Quantum key distribution
- Quantum error correction
- ...