A New Channel Allocation Scheme for Vehicle Communication Networks

Jing Xu, Wei Li, Zan Ma, and Shuo Zhang

What is this paper about?

Smart roads provide vehicles with updates on traffic, safety warnings and weather conditions. However, being centralized systems, they can suffer from issues such as overloading and single points of failure.

This paper proposes a distributed peer-to-peer (car-to-car) communication scheme that ensures:

- Crucial and lightweight information (e.g. car accidents, hazardous weather phenomena) travels on dedicated, interference-free primary channels;
- High-demand information (e.g. infotainment services) is transmitted on secondary channels, dynamically assigned based on each vehicle's needs.

brief,packets with high real-time requirements can be sent in $CH_1(u)$,and much quantity in the $CH_2(u)$.

The Model (I)

Model each car as a circle given by its transmission range and build a (dynamic) intersection graph.

Let's consider the generic node u. As in a labeling problem,

- Any node at distance 1 is a first neighbour
- Any node at distance 2 is a second neighbour

that it has been received by sending ACK from node v to u. $N_1(u)$ denotes nodes which are one hop away from node u, and $N_2(u)$ denotes nodes which are two hops from node u, it is obvious that $u \notin N_1(u), u \notin N_2(u)$. In this paper

We put them together to form the interference set $N(u)$ for node u.

- Bandwidth is divided in K channels, and we assume interference may occur only if two different nodes in a neighborhood use the same channel.

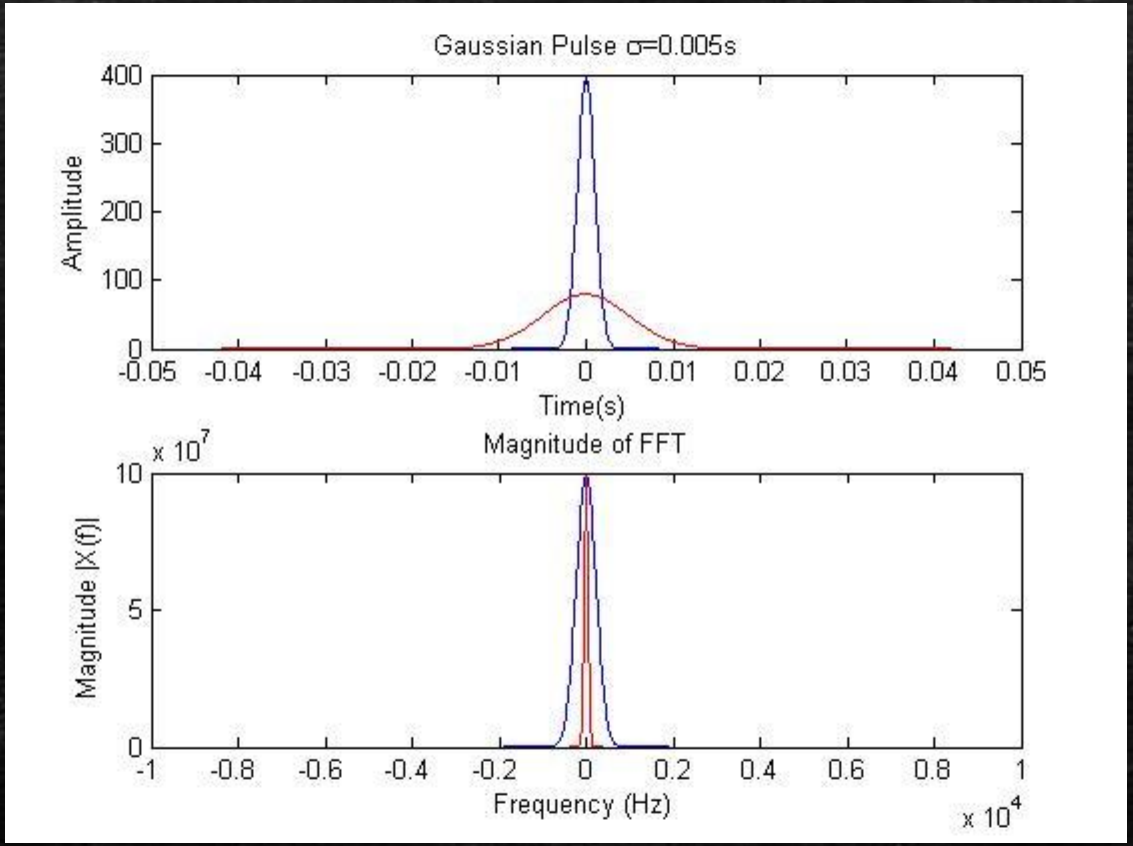
The Model (II)

But that's not a standard L(2, 1) frequency assignment scheme!

- L(1, 1) is computationally easier
- Under certain assumptions, we still have no interference at distance 1
- We might say that L(2, 1) is a "safe mode", while L(1, 1) accepts some risks in order to better exploit the available spectrum

The Model (III)

Under certain assumptions, we still have no interference at distance 1

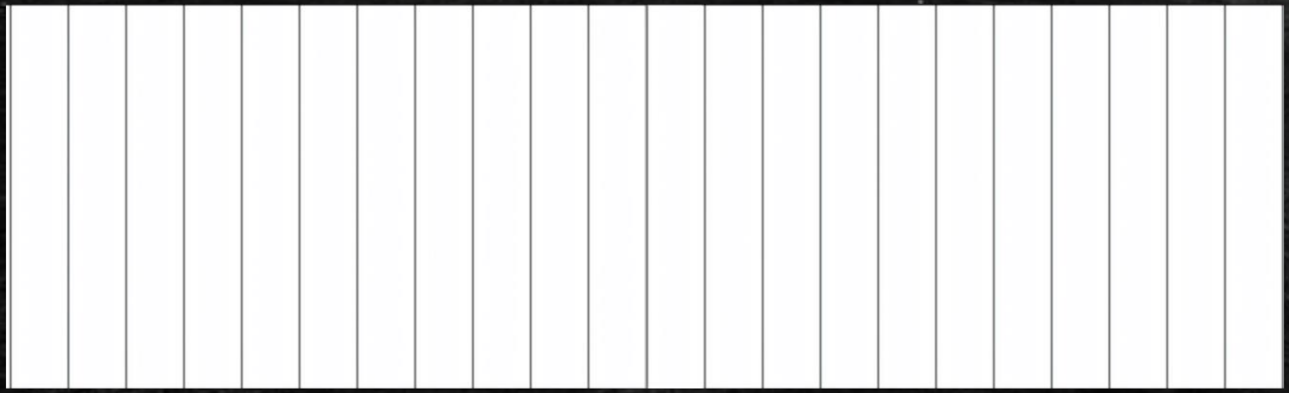


Conceptual Framework

- Primary channels are found by some raw approximation of $L(1, 1)$.
- Secondary channels are then assigned by a greedy algorithm.

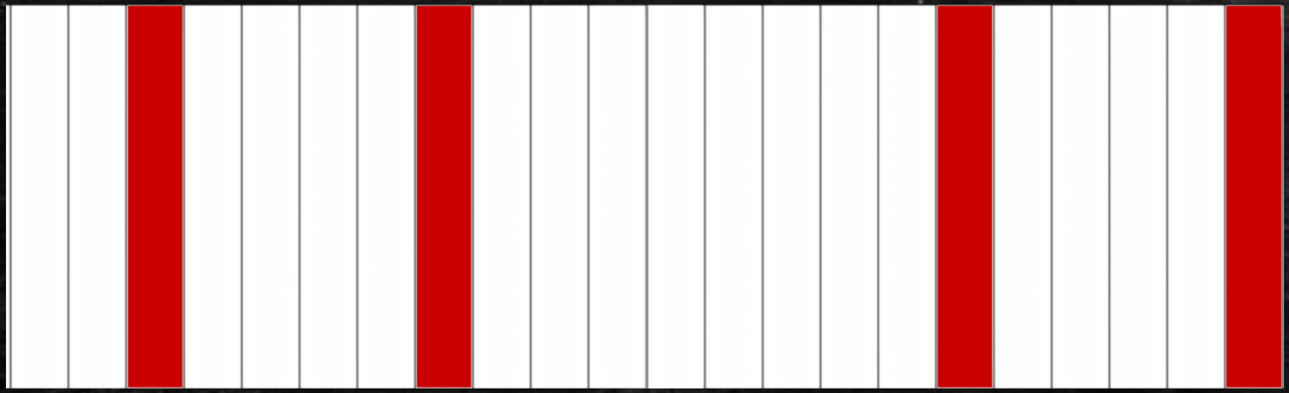
Conceptual Framework

- Primary channels are found by some raw approximation of $L(1, 1)$.
- Secondary channels are then assigned by a greedy algorithm.



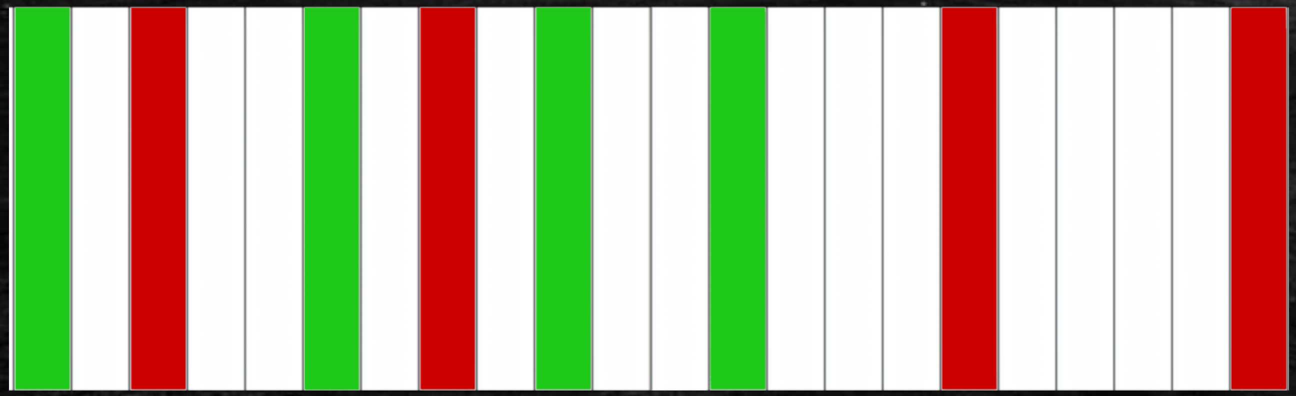
Conceptual Framework

- Primary channels are found by some raw approximation of $L(1, 1)$.
- Secondary channels are then assigned by a greedy algorithm.



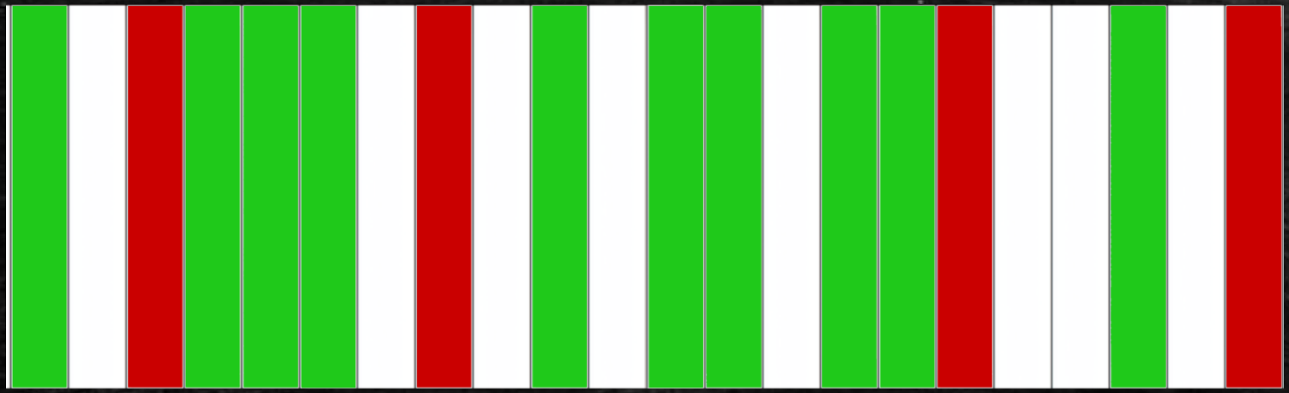
Conceptual Framework

- Primary channels are found by some raw approximation of $L(1, 1)$.
- Secondary channels are then assigned by a greedy algorithm.



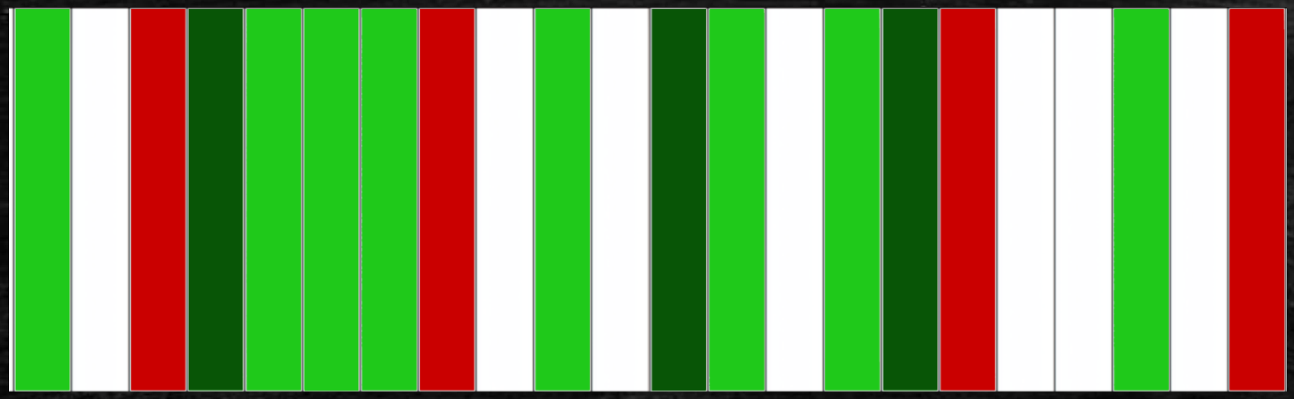
Conceptual Framework

- Primary channels are found by some raw approximation of $L(1, 1)$.
- Secondary channels are then assigned by a greedy algorithm.



Conceptual Framework

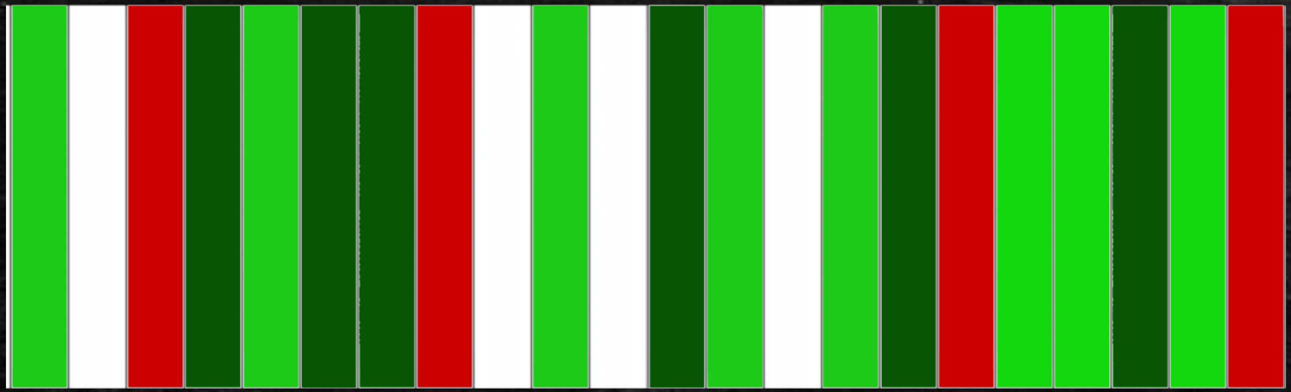
- Primary channels are found by some raw approximation of $L(1, 1)$.
- Secondary channels are then assigned by a greedy algorithm.



A single secondary channel may be chosen by more than one node…

Conceptual Framework

- Primary channels are found by some raw approximation of $L(1, 1)$.
- Secondary channels are then assigned by a greedy algorithm.



... while some others may remain unused.

Conceptual Framework

- Primary channels are found by some raw approximation of $L(1, 1)$.
- Secondary channels are then assigned by a greedy algorithm.

If a secondary channel is chosen by more than one node, we perform

Frequency Division Multiplexing (OFDMA)

on that channel, furtherly dividing it into sub-channels.

Tools for choosing the Primary Channels (I)

- Primary channels are found by some raw approximation of $L(1, 1)$.

Suppose to have a K-bit long vector. We have 2^K possible words w_i such that if we define the boolean sum as a bitwise OR operation we have that

- $BooleanSum(w_i, w_j)$ is still a word;
- It may happen that $BooleanSum(w_i, w_j) = w_i$. We say w_i covers w_j.

Tools for choosing the Primary Channels (II)

Given a set W of binary strings of length K, the goal of a Superimposed Code (SC) is to provide a code (i.e. a set $M \subset W$) that is as robust as possible against coverings (i.e. we build a set of codewords that minimize covering phenomena).

It is defined as a triple (s, L, K) where L and s are integer numbers such that if I sum s codewords in M, the result r will cover at most $L - 1$ codewords among the remaining $|M| - s$.

i.e. there are at most $L - 1$ codewords in the subset $\{M - \{s\}\}$ such that $r \text{ OR } w = r$.

> Let N, t, s, and L be integers such that $1 < s < t$, $1 \leq L \leq t - s$, and $N > 1$. Given a $N \times t$ binary matrix $\mathcal{X}$, denote the ith column of $\mathcal{X}$ by $X(i)$, where $X(i) = (x_1(i), x_2(i), \cdots, x_N(i))'$. We call $X(i)$ a codeword i of $\mathcal{X}$ with a length N. In other words, $\mathcal{X}$ is a *binary code* with each column corresponding to a codeword.

Xing et al - Channel Assignment via Superimposed Code (2007)

Tools for choosing the Primary Channels (III)

- L is called Reliability (or Overlap Bound). Why?

  - Choosing big values of L means admitting several coverages.

    $L = |M| - s + 1$ means admitting the coverage of at most the whole complementary set.

    Computationally easy, but useless.

  - The optimal value is $L = 1$. (no codeword in the complementary set is ever covered). Such a Superimposed Code is called s-disjunctcode, and from now on we will assume to be given such a code.

Tools for choosing the Primary Channels (IV)

- s is called Strength. Why?

  - Choosing small values of s means that the sum of "few" codewords will cover at most one codeword among the remaining "many";

  - Choosing big values of s means that the sum of "many" codewords will cover at most one codeword among the remaining "few".

  $s = |M| - 2$ means that the sum of all the codewords but two will cover at most one of the remaining 2 codewords in the complementary set.

Tools for choosing the Primary Channels (V)

LEMMA 4.1. *Given an $(s, 1, N)$ superimposed code $\mathcal{X}$, for any s-subset of the codewords of $\mathcal{X}$, there exists at least one row at which all codewords in the s-subset contains the value 0.*

PROOF. For contradiction we assume that there is no row at which all codewords in the s-subset contain a common value 0. Then the Boolean sum of the s codewords equals $(1, 1, \cdots, 1)'$, which can cover all other codewords in $\mathcal{X}$, contradicting to the fact that $\mathcal{X}$ is a superimposed s-disjunct code. $\square$

Xing et al - Channel Assignment via Superimposed Code (2007)

Tools for choosing the Primary Channels (VI)

$$\begin{pmatrix}
1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 \\
1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\
0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 \\
1 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 1 \\
1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1
\end{pmatrix}$$

Figure 1: An example of a superimposed $(3, 1, 13)$-code of size 13

Xing et al - Channel Assignment via Superimposed Code (2007)

A (3, 1, 13)-code is a 3-disjunct code, meaning you can sum any subset of up to s=3 columns of length N=13 and be sure the sum won't cover any of the remaining t-s = 13-3 = 10 codewords.

N=t is not by chance: for small s w.r.t. N, we tend to have t proportional to N.

Choosing the Primary Channels (I)

Algorithm 1. A distributed wireless channel allocation scheme for vehicle networks

Input: The initial information of each node u: C, $ChEstimate(u)$, $N_1(u)$, $NumCh(u)$, $R(u)$, $Rate(u)$.

Output: Each node u chooses its primary channel $CH_1(u)$ to send important information and the channel set $CH_2(u)$ to deliver large amounts of packets.

step 1: Each node broadcasts its ID and forward the received neighbor ID once, thus everyone will get the $\mathcal{N}(u)$.

step 2: $\forall u \in V, CH_1(u) = Channels(BoolSum(\mathcal{X}(\mathcal{N}(u) \bigcup \{u\}))) \bigoplus BoolSum(\mathcal{X}(\mathcal{N}(u)))$

▷ find the primary channels for u, and secondary channels for $\mathcal{N}(u)$, then choose one to be the $CH_1(u)$.

Every node broadcasts its ID (i.e. its codeword) with TTL = 2.

Then, every u is able to perform the XOR operation to find the channels which are primary to u and not primary to any neighbor.

Choosing the Primary Channels (II)

- First, we equip every node in the network with an ***s-disjunct code*** $\mathcal{X}$. We set the length K of the codeword equal to the number of orthogonal channels k_i;
- A generic node u randomly selects a codeword $\mathcal{X}(u) = \vec{c_u}$ from the $|\mathcal{X}|$ available options, e.g.

$$\vec{c_u} = (100100101\cdots)$$

- The element $\vec{c_u}(i)$ represents channel k_i. Specifically, $k_i = 1$ means that k_i is a candidate to become the primary channel $\mathrm{CH}_1(u)$ for node u;
- Each node broadcasts its pair {`ID` (e.g. a `MAC Address`), $\vec{c_u}$}, and forwards the pairs of its neighbors so that every neighbor reaches is whole neighborhood.
  - It's more or less like a broadcast with `TTL = 2`.
- At this point, each node u knows the `ID` and the codeword chosen by every node in its neighborhood, which forms its ***interference set*** $\mathcal{N}(u)$, i.e., the nodes that must choose disjoint CH_1 to avoid mutual interference (i.e. a labeling $L(1,1)$).

Choosing the Primary Channels (III)

- It then calculates two elements:
  - $\mathrm{BoolSum}(\mathcal{X}(\mathcal{N}(u) \cup u))$ - All possible CH_1 in the neighborhood, ***including*** u;
  - $\mathrm{BoolSum}(\mathcal{X}(\mathcal{N}(u)))$ - All possible CH_1 in the neighborhood, ***excluding*** u.
- Finally, the list of channels $\mathrm{CH}_1(u)$ is obtained from the ***XOR*** of these two

$$\mathrm{BoolSum}(\mathcal{X}(\mathcal{N}(u) \cup \{u\}) \oplus \mathrm{BoolSum}(\mathcal{X}(\mathcal{N}(u))$$

Or, in a simpler way,

$$\mathcal{X}(u)\ \mathrm{AND}\ (\mathrm{NOT}\ \mathrm{BoolSum}(\mathcal{X}(\mathcal{N}(u)))$$

Choosing the Primary Channels (III)

LEMMA 5.1. *If $CH_1(u) \neq \emptyset$, node u does not interfere with any other node in $\mathcal{N}(u)$.*

PROOF. When $CH_1(u) \neq \emptyset$, node u picks up channels from $CH_1(u)$, a subset of u's primary channel set, for transmission. $CH_1(u)$ contains channels that are primary to u but secondary to all nodes in $\mathcal{N}(u)$. For $\forall v \in \mathcal{N}(u)$, v can't use any channel from $CH_1(u)$ based on Algorithm 1 since v is assigned with either its own primary channels (from $CH_1(v)$ or $CH_3(v)$), which can't be in $CH_1(u)$, or channels that are secondary to all interferers in $\mathcal{N}(v)$ ($CH_2(v)$), which are secondary to u too since $u \in \mathcal{N}(v)$. $\square$

Xing et al - Channel Assignment via Superimposed Code (2007)

How can we be sure we'll find at least one CH1?

How do we know CH1 will be non-empty?

Choosing the Primary Channels (IV)

Lemma 1. *If $s \geq |\mathcal{N}(u)|$ and $\mathcal{N}(u)$ is the complete set of interferers of u for any node,the $CH_1(u)$ exists surely.*

Proof. Since $\mathcal{X}$ is an s-disjunct code,$BoolSum$ $(\mathcal{X}(\mathcal{N}(u)))$ dose not cover $\mathcal{X}(u)$,which means that there exists at least one row in $\mathcal{X}$ at which $\mathcal{X}(u)$ has the value 1 and all $\mathcal{X}(\mathcal{N}(u))$ have the value 0.Therefore the conclusion $CH_1(u) \neq \varnothing$ holds.

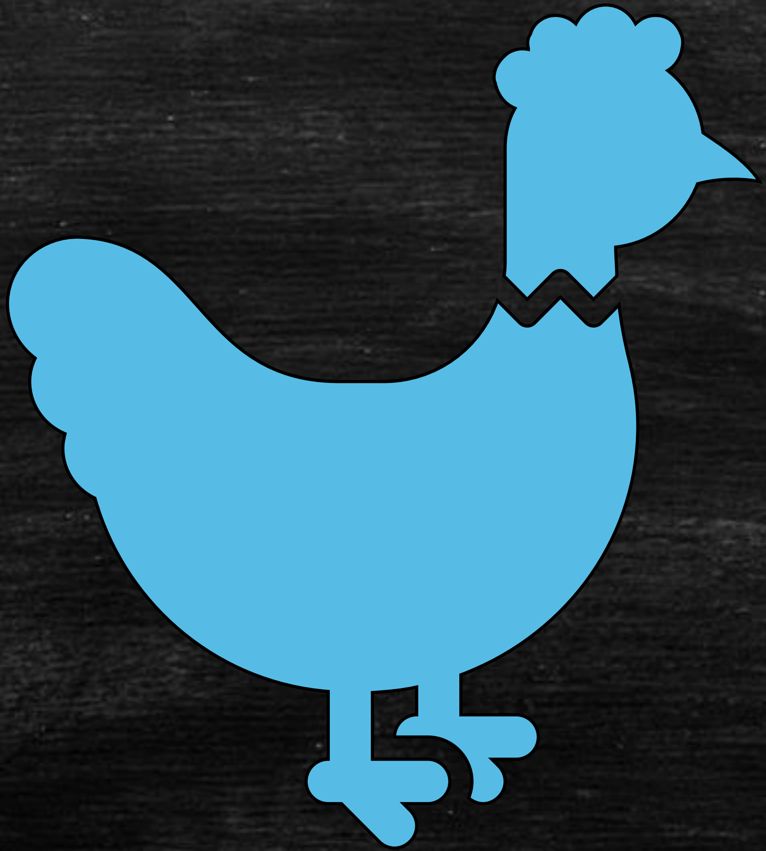
Xu et al - A New Channel Allocation Scheme for Vehicle Communication Networks (2014)

THEOREM 5.2. *If $s \geq |\mathcal{N}(u)|$ and $\mathcal{N}(u)$ is the complete set of interferers of u for $\forall u$ in G, the channel assignment based on Algorithm 1 guarantees interference free communications in the network.*

PROOF. Since $\mathcal{X}$ is an s-disjunct code, $BoolSum(\mathcal{X}(\mathcal{N}(u)))$ does not cover $X(u)$, which means that there exists at least one row in $\mathcal{X}$ at which $X(u)$ has the value 1 and all $\mathcal{X}(\mathcal{N}(u))$ have the value 0 (see Lemma 4.1). Therefore condition $CH_1(u) \neq \emptyset$ holds. Based on Theorem 5.1, the claim holds. □

Xing et al - Channel Assignment via Superimposed Code (2007)

Uhm...



A New Channel Allocation Scheme for Vehicle Communication Networks

Jing Xu, Wei Li, Zan Ma, and Shuo Zhang (2014)

… also known as...

Superimposed Code Based Channel Assignment in Multi-Radio Multi-Channel Wireless Mesh Networks

Kai Xing, Xiuzhen Chen, Liran Ma, Qilian Liang (2007)

At least (more or less) regarding the primary channels.

Primary Channels (Xu et al, 2014)

Algorithm 1. A distributed wireless channel allocation scheme for vehicle networks

Input: The initial information of each node u: C, $ChEstimate(u)$, $N_1(u)$, $NumCh(u)$, $R(u)$, $Rate(u)$.

Output: Each node u chooses its primary channel $CH_1(u)$ to send important information and the channel set $CH_2(u)$ to deliver large amounts of packets.

step 1: Each node broadcasts its ID and forward the received neighbor ID once, thus everyone will get the $\mathcal{N}(u)$.

step 2: $\forall u \in V, CH_1(u) =$
$Channels(BoolSum(\mathcal{X}(\mathcal{N}(u) \bigcup \{u\}))) \bigoplus BoolSum(\mathcal{X}(\mathcal{N}(u)))$

▷ find the primary channels for u, and secondary channels for $\mathcal{N}(u)$, then choose one to be the $CH_1(u)$.

Like we said, every node broadcasts its ID (i.e. its codeword) with TTL = 2.

Then, every u is able to perform the XOR operation to find the channels which are primary to u and not primary to any neighbor.

But this works only if we have at most s neighbors.

What if we happen to have more?

Primary Channels (Xing et al, 2007)

Here the idea is the following: we don't know if the numer of neighbors is lesser or equal than s. If that's not the case, we're not sure to find CH1 as we discussed so far.

In case this first approach fails, u may look for secondary channels which are secondary also to all its neighbors.

In case this second approach also fails, rely on the "lesser evil" choosing the "least-interference channel"

Algorithm 1 Channel Assignment for Node u

Input: Codewords $X(u)$ and $\mathcal{X}(\mathcal{N}(u))$.
Output: $CH(u)$, the set of channels assigned to u.

function $CH(u)$=**ChannelSelect**($X(u)$, $\mathcal{X}(\mathcal{N}(u))$)
  $CH_1(u) \leftarrow Channels(BoolSum(\mathcal{X}(\mathcal{N}(u) \cup \{u\})) \oplus BoolSum(\mathcal{X}(\mathcal{N}(u))))$ ▷ Find the set of primary channels that are secondary to all nodes in $\mathcal{N}(u)$.
  if $CH_1(u) \neq \emptyset$ **then**
    $CH(u) \leftarrow CH_1(u)$
  else
    $CH_2(u) \leftarrow Channels(\overline{BoolSum(\mathcal{X}(\mathcal{N}(u) \cup \{u\}))})$ ▷ Find the set of secondary channels that are secondary to all nodes in $\mathcal{N}(u)$.
    if $CH_2(u) \neq \emptyset$ **then**
      $CH(u) \leftarrow CH_2(u)$
    else
      $CH_3(u) \leftarrow$ Select $Channels(X(u))$ with the smallest row weight in $\mathcal{X}(\mathcal{N}(u))$ ▷ Select the primary channels with the least row weight in $\mathcal{N}(u)$.
      $CH(u) \leftarrow CH_3(u)$
    end if
  end if
end function

What about the Secondary Channels?

We were supposed to assign some secondary channels via some greedy algorithm.

Xing et al (2007) never mentions such an assignment.

step 3: $\forall u \in V, AvaliableCH(u) = C - \sum_{v\in(\{u\}\bigcup\mathcal{N}(u))} CH_1(v)$. ($C$ represents the whole channel set)
step 4: $\forall u \in V, Priority(u) = \frac{R(u)}{Rate(u)}$, and send it to their $\mathcal{N}(u)$.
step 5: $\forall u \in V, Sort(Priority(v)), v \in (u\bigcup\mathcal{N}(u))$, thus we can get each node u's priority order $Seq(u)$ among the nodes in $\mathcal{N}(u)$.
step 6: $Token = 1$;
step 7: $\forall u \in V$, **if** $Seq(u) == Token : CH_2(u) =$ the highest $NumCh(u)$ channels on the value of $Estimate(u)$ among the $Available(u)$.

$$\textbf{if} \quad |CH_2(u)| < NumCh(u)$$

$$\textbf{then} \quad Pritority(u) \quad Add \quad \textbf{endif}$$

$Token = Token + 1$ **endif**.
step 8: **if** there exists any node u which has not been involved in the allocation scheme, **turn to step 7**;
else **break**;

Choosing the Secondary Channels (I)

The idea behind the assignment is the following.

Every node has an idea of its throughput needs and of its transmission speed. Based on this, we can compute a *Priority* value for choosing the best secondary channels among the available ones.

The evaluation of the "best" is based on some metrics given by a function called *ChEstimate* (or sometimes, randomly, just *Estimate*).

Then, we initialise a global variable named *Token* and we loop over all the possible nodes u of the graph. Each node loops over *Token*, so that when the *Sequence* value (computed by means of *Priority*) is equal to the *Token* I can choose the best channels available. Somehow, it may happen that a node can't get enough channels. In this case, we just *Priority Add* (whatever it means, since it's the only occurrence in the whole paper). Even though this should be the core operation

Choosing the Secondary Channels (II)

In brief, despite optimizing the assigment of the secondary channels is a good idea, the second part of the Xu et al (2014) paper has a bunch of problems.

If we go back to Xing et al (2007), like we said, there's no such an assignment.

We can simply choose not to assign the secondary channels.

Each node can rely on them for transmissions of lesser importance (e.g. infotainment), accepting the risk of interference.

Conclusions

A simple, light and distributed scheme for finding an approximate solution to the L(1,1) interference-free primary channel assignment is achieved by means of an s-disjunct code respecting some further conditions.

Secondary channels for high throughput transmissions have to deal with some interference.