The Treewidth of a Linear Code

Navin Kashyap

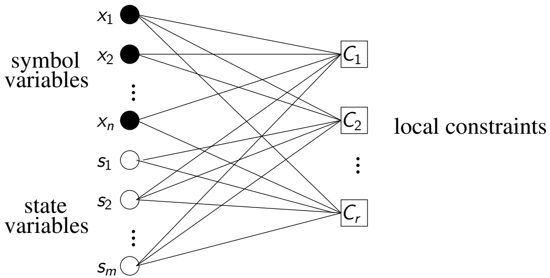
Department of Electrical Communication Engineering
Indian Institute of Science

November 3, 2011

Outline of Talk

General Linear Realizations

Generic Factor Graph :



The state and symbol spaces, as well as the local constraint codes, are all linear.

The full behaviour $\mathcal{B}$ of such a realization is the set of all symbol/state configurations that satisfy all local constraints.

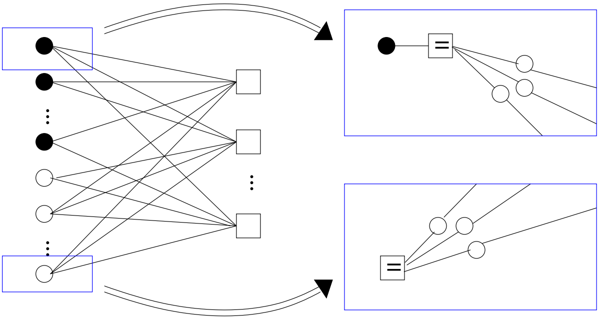
The code realized is the projection of $\mathcal{B}$ onto the symbol vars.

Normal Realizations

In a normal realization,

- all state variables have degree two,
- all symbol variables have degree one.

Any linear realization can be normalized [Forney (2001)].

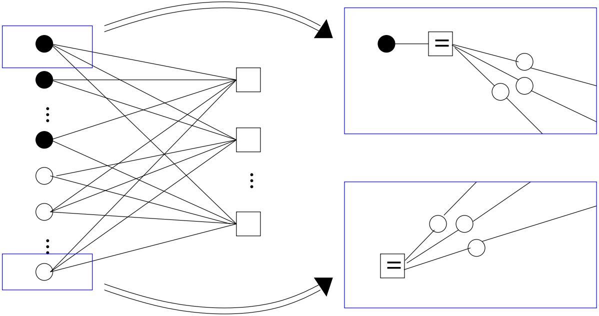


Normal Realizations

In a normal realization,

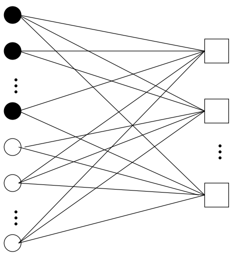
- all state variables have degree two,
- all symbol variables have degree one.

Any linear realization can be normalized [Forney (2001)].

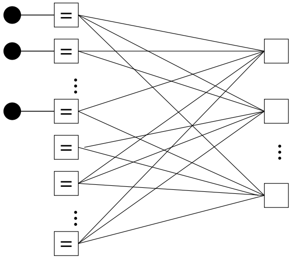


Normalization preserves cycle-free structure.

Normal Realizations



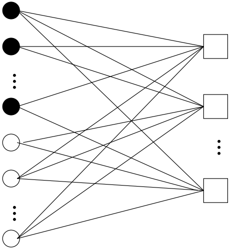
Factor graph



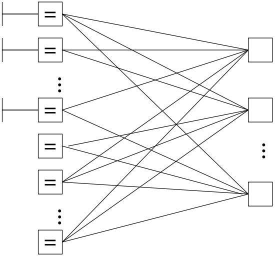
Normal graph

In a normal graph, state variables sit on *edges*.

Normal Realizations



Factor graph



Normal graph

In a normal graph, state variables sit on *edges*, and symbol variables are depicted by “dongles”.

Example: RM(1, 3)

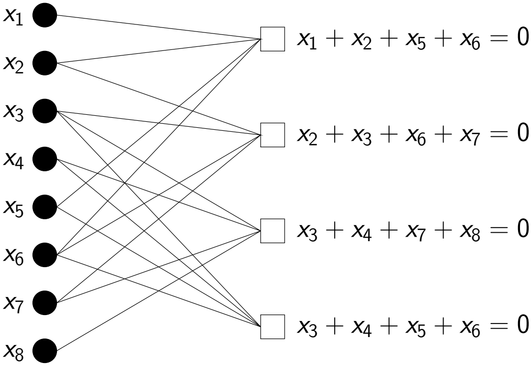
Consider the [8,4] binary Reed-Muller code $\mathrm{RM}(1,3)$ defined to be the nullspace (kernel) of the parity-check matrix

$$H = \begin{bmatrix} 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 \end{bmatrix}$$

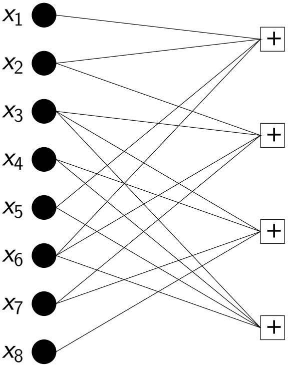
Thus, $\mathrm{RM}(1,3)$ consists of all $(x_1, x_2, \ldots, x_8) \in \{0,1\}^8$ such that:

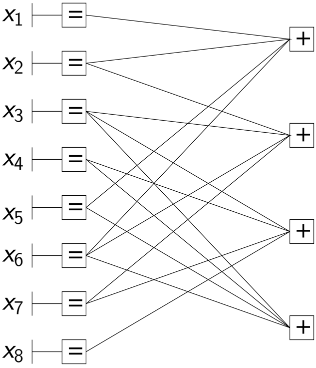
$$\begin{aligned} x_1 + x_2 + x_5 + x_6 &= 0 \\ x_2 + x_3 + x_6 + x_7 &= 0 \\ x_3 + x_4 + x_7 + x_8 &= 0 \\ x_3 + x_4 + x_5 + x_6 &= 0 \end{aligned}$$

RM(1, 3): Tanner Graph



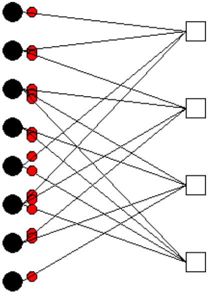
RM(1, 3): Normal Realization





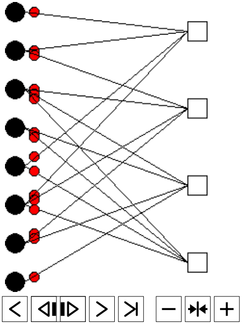
Message-Passing Decoder

A Tanner graph realization admits an iterative message-passing decoding algorithm for the code.



Message-Passing Decoder

A Tanner graph realization admits an iterative message-passing decoding algorithm for the code.

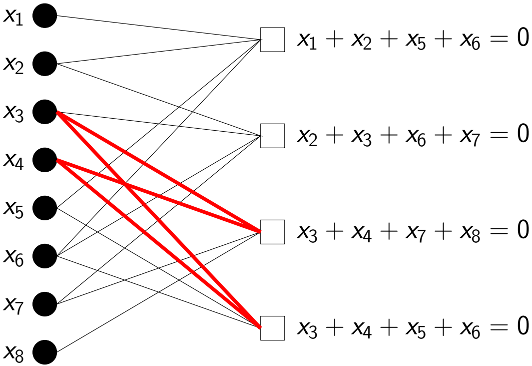


Goal of decoding algorithm (given received word $\mathbf{y}$):

recover, at each coordinate x_i, a vector proportional to the a-posteriori probability (APP) vector

$[p(x_i = 0 \mid \mathbf{y}), \; p(x_i = 1 \mid \mathbf{y})]$.

RM(1, 3) Tanner Graph has Cycles



On a Tanner graph with cycles, iterative message-passing decoding is not guaranteed to produce the correct output, or even to converge.

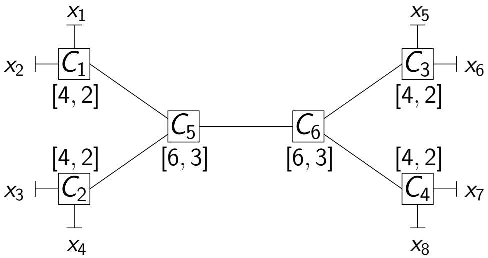
Cycle-Free Tanner Graphs

Iterative message-passing decoding algorithms are guaranteed to converge to the correct output on a cycle-free graph.

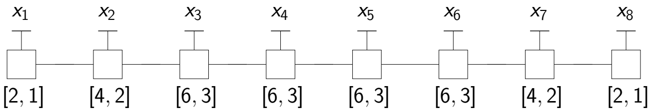
$RM(1,3)$ has no cycle-free Tanner graph
[by a result of Etzion-Trachtenberg-Vardy (1999)].

Cycle-Free Realizations of RM(1, 3)

Tree realization:

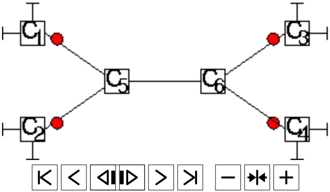


Trellis realization:

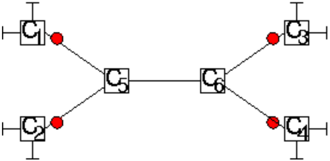


(State variables on edges are not shown.)

Message-Passing Decoding



Message-Passing Decoding



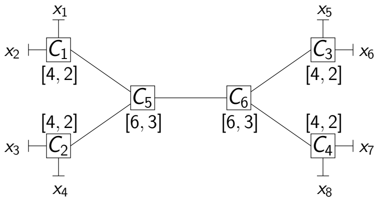
The computational complexity of the message-passing decoding algorithm is proportional to

$$\sum_i \deg(v_i)\,|C_i| = \sum_i \deg(v_i)\,2^{\dim(C_i)}$$

where v_i is the vertex of the graph within which C_i sits.

[Aji-McEliece (2001), Forney (2001)]

Constraint Complexity

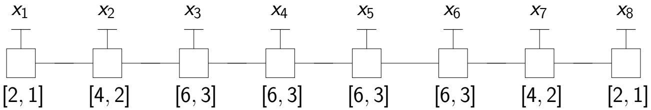


$\sum_i \deg(v_i)\, 2^{\dim(C_i)}$ is dominated by the $2^{\max_i \dim(C_i)}$ terms.

We call $\max_i \dim(C_i)$ the constraint complexity of the realization.

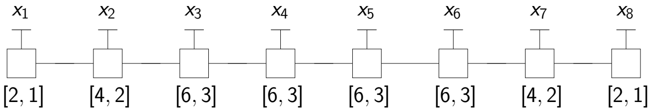
For the realization shown, the constraint complexity is 3.

Another Cycle-Free Realization of RM(1, 3)



The constraint complexity is again 3.

Another Cycle-Free Realization of RM(1, 3)



The constraint complexity is again 3.

We will define the treewidth of a code (e.g. RM(1, 3)) to be the least constraint complexity of any of its cycle-free realizations.

The treewidth of a code estimates the complexity of implementing optimum (maximum-likelihood) decoding as a message-passing algorithm on the best cycle-free realization of the code.

Definitions: Tree and Trellis Realizations

Definition

A normal realization is called

- a tree realization if its underlying graph is a tree (a tree is a cycle-free connected graph).
- a trellis realization if its underlying graph is a path (a path is a tree in which all vertices lie in a straight line).

Definitions: Treewidth and Trelliswidth

Definition

The constraint complexity of a normal realization is the maximum dimension among its local constraint codes.

Definition

The treewidth (resp. trelliswidth) of a code is the least constraint complexity among its tree (resp. trellis) realizations.

- $\kappa_{\text{tree}}(\mathcal{C}) :=$ treewidth of code $\mathcal{C}$
- $\kappa_{\text{trellis}}(\mathcal{C}) :=$ trelliswidth of code $\mathcal{C}$

Comparing κ_{tree} and κ_{trellis}

Since a trellis realization is a special type of tree realization, for any code $\mathcal{C}$, we have

$$\kappa_{\text{tree}}(\mathcal{C}) \leq \kappa_{\text{trellis}}(\mathcal{C}).$$

Comparing κ_{tree} and κ_{trellis}

Since a trellis realization is a special type of tree realization, for any code $\mathcal{C}$, we have

$$\kappa_{\text{tree}}(\mathcal{C}) \le \kappa_{\text{trellis}}(\mathcal{C}).$$

Theorem (K. (2009))

For a linear code $\mathcal{C}$ of length $n > 1$,

$$\frac{\kappa_{\textit{trellis}}(\mathcal{C})}{\kappa_{\textit{tree}}(\mathcal{C})} \le 2\log_2(n-1)+3.$$

Comparing κ_{tree} and κ_{trellis}

Since a trellis realization is a special type of tree realization, for any code $\mathcal{C}$, we have

$$\kappa_{\text{tree}}(\mathcal{C}) \leq \kappa_{\text{trellis}}(\mathcal{C}).$$

Theorem (K. (2009))

For a linear code $\mathcal{C}$ of length $n > 1$,

$$\frac{\kappa_{\mathit{trellis}}(\mathcal{C})}{\kappa_{\mathit{tree}}(\mathcal{C})} \leq 2\log_2(n-1) + 3.$$

This bound on the ratio is the best possible, up to the constants involved. It is known [K. (2007)] that a sequence of codes $\mathcal{C}^{(i)}$, $i = 1, 2, \ldots$, of length n_i, exists such that

$$\frac{\kappa_{\text{trellis}}(\mathcal{C}^{(i)})}{\kappa_{\text{tree}}(\mathcal{C}^{(i)})} \approx \frac{1}{4}\log_2 n_i.$$

Parametrized Complexity of ML decoding

Theorem (K. (2009))

The complexity of maximum-likelihood decoding of a length-n linear code $\mathcal{C}$ over $\mathbb{F}_q$ is $O(nq^t)$, where t is the treewidth of $\mathcal{C}$.

As a corollary, we see that

codes of bounded treewidth are linear-time decodable.

Parametrized Complexity of ML decoding

Theorem (K. (2009))

The complexity of maximum-likelihood decoding of a length-n linear code $\mathcal{C}$ over $\mathbb{F}_q$ is $O(nq^t)$, where t is the treewidth of $\mathcal{C}$.

As a corollary, we see that

codes of bounded treewidth are linear-time decodable.

However, codes of bounded treewidth do not have good minimum distance, and so may not be good from an error-correcting perspective.

Proof Sketch

Forney (2003) made the following observation:

A length-n linear code $\mathcal{C}$ has an optimal tree realization in which the underlying tree

(a) is cubic (i.e., all internal nodes have degree 3), and
(b) has n leaves

Here, "optimal" means that the constraint complexity of the realization equals the treewidth of the code.

Proof Sketch

Forney (2003) made the following observation:

> A length-n linear code $\mathcal{C}$ has an optimal tree realization in which the underlying tree
> (a) is cubic (i.e., all internal nodes have degree 3), and
> (b) has n leaves
>
> Here, "optimal" means that the constraint complexity of the realization equals the treewidth of the code.

The computational complexity of message-passing decoding on such an optimal tree realization T is proportional to

$$\sum_{v \in V(T)} \deg(v)\, q^{\dim(C_v)} \le \sum_{v \in V(T)} 3 \cdot q^{\kappa_{\text{tree}}(\mathcal{C})} = 3(2n-2)q^{\kappa_{\text{tree}}(\mathcal{C})}$$

the last equality because a cubic tree with n leaves has exactly $(2n-2)$ vertices.

Computing Treewidth

Theorem (from Hliněný & Whittle (2008))

Computing the treewidth of a linear code is NP-hard.

Computing Treewidth

Theorem (from Hliněný & Whittle (2008))

Computing the treewidth of a linear code is NP-hard.

Theorem (K. and Thangaraj (2011))

For an $[n, k]$ MDS code,

$$\textit{treewidth} = \textit{trelliswidth} = \min(k, n-k+1).$$

For the Reed-Muller code $RM(r, m)$,

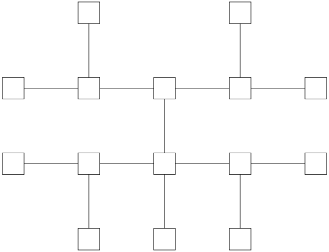
$$\textit{treewidth} = \textit{trelliswidth} = \begin{cases} \sum_{j=0}^{r} \binom{m-2j-1}{r-j} & \text{if } m \geq 2r+1 \\ 1 + \sum_{j=0}^{m-r-1} \binom{m-2j-1}{r-j} & \text{if } m < 2r+1 \end{cases}$$

Constructing Tree Realizations

Given: a code $\mathcal{C}$ of length n

Select:

- T - a tree

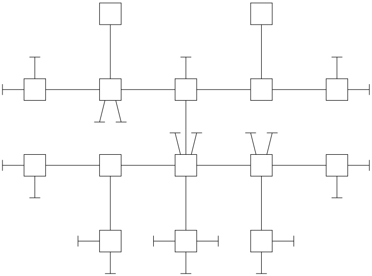


Constructing Tree Realizations

Given: a code $\mathcal{C}$ of length n

Select:

- T - a tree, and
- ω - an assignment of the n coordinates of $\mathcal{C}$ (i.e. the symbol variables) to $V(T)$

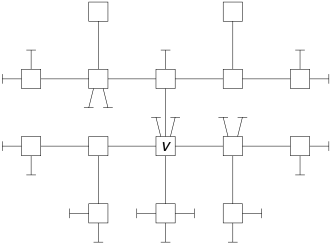


Local Constraint Codes [Forney (2003)]

Notation: For $J \subseteq [n]$, $\mathcal{C}_J := \{\mathbf{c} \in \mathcal{C} : \mathbf{c}|_{J^c} = \mathbf{0}\}$.

Local Constraint Codes [Forney (2003)]

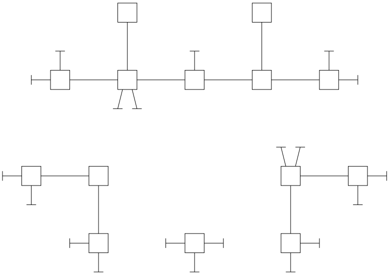
Notation: For $J \subseteq [n]$, $\mathcal{C}_J := \{\mathbf{c} \in \mathcal{C} : \mathbf{c}|_{J^c} = \mathbf{0}\}$.



For a node $v \in V(T)$ with degree δ:

Local Constraint Codes [Forney (2003)]

Notation: For $J \subseteq [n]$, $\mathcal{C}_J := \{\mathbf{c} \in \mathcal{C} : \mathbf{c}|_{J^c} = \mathbf{0}\}$.

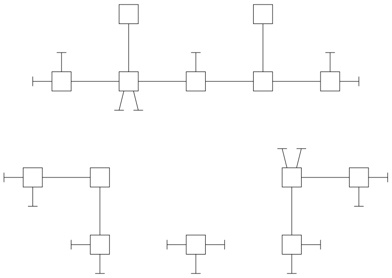


For a node $v \in V(T)$ with degree δ:

- the removal of v from T yields a graph whose components, $T_1, \ldots, T_\delta$, are subtrees of T

Local Constraint Codes [Forney (2003)]

Notation: For $J \subseteq [n]$, $\mathcal{C}_J := \{\mathbf{c} \in \mathcal{C} : \mathbf{c}|_{J^c} = \mathbf{0}\}$.

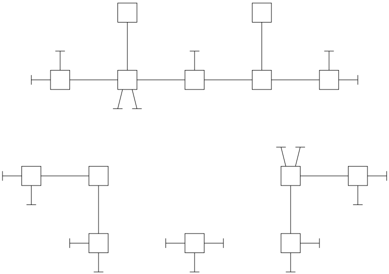


For a node $v \in V(T)$ with degree δ:

- the removal of v from T yields a graph whose components, $T_1, \ldots, T_\delta$, are subtrees of T
- for $i = 1, \ldots, \delta$, set $J_i = \omega^{-1}(V(T_i))$

Local Constraint Codes [Forney (2003)]

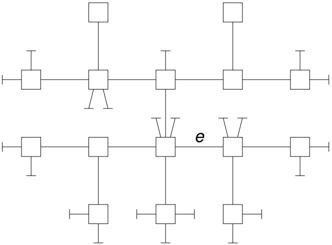
Notation: For $J \subseteq [n]$, $\mathcal{C}_J := \{\mathbf{c} \in \mathcal{C} : \mathbf{c}|_{J^c} = \mathbf{0}\}$.



For a node $v \in V(T)$ with degree δ:

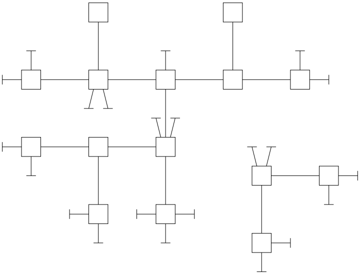
- the removal of v from T yields a graph whose components, $T_1, \ldots, T_\delta$, are subtrees of T
- for $i = 1, \ldots, \delta$, set $J_i = \omega^{-1}(V(T_i))$
- $C_v :=$ local constraint code at v $= \mathcal{C} / \bigoplus_{i=1}^{\delta} C_{J_i}$

State Spaces [Forney (2003)]



For an edge $e \in E(T)$:

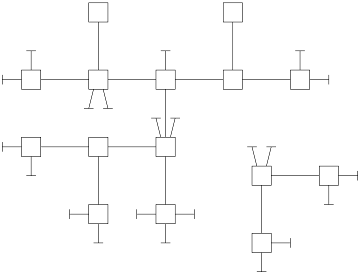
State Spaces [Forney (2003)]



For an edge $e \in E(T)$:

- the removal of e from T yields a graph with two components, T' and T''

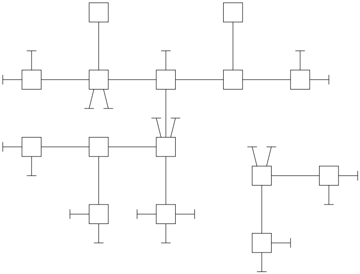
State Spaces [Forney (2003)]



For an edge $e \in E(T)$:

- the removal of e from T yields a graph with two components, T' and T''
- set $J' = \omega^{-1}(V(T'))$ and $J'' = \omega^{-1}(V(T''))$

State Spaces [Forney (2003)]



For an edge $e \in E(T)$:

- the removal of e from T yields a graph with two components, T' and T''
- set $J' = \omega^{-1}(V(T'))$ and $J'' = \omega^{-1}(V(T''))$
- $S_e :=$ state space at e $= \mathcal{C}/(\mathcal{C}_{J'} \oplus \mathcal{C}_{J''})$

Constraint Complexity

For each $v \in V(T)$ in Forney's tree realization,

$$\dim(C_v) = \dim(\mathcal{C}) - \sum_{i=1}^{\delta} \dim(\mathcal{C}_{J_i})$$

Hence, the constraint complexity of the realization is

$$\kappa(\mathcal{C}; T, \omega) := \max_{v \in V(T)} \left[\dim(\mathcal{C}) - \sum_{i=1}^{\delta} \dim(\mathcal{C}_{J_i}) \right]$$

Constraint Complexity

For each $v \in V(T)$ in Forney's tree realization,

$$\dim(C_v) = \dim(\mathcal{C}) - \sum_{i=1}^{\delta} \dim(\mathcal{C}_{J_i})$$

Hence, the constraint complexity of the realization is

$$\kappa(\mathcal{C}; T, \omega) := \max_{v \in V(T)} \left[\dim(\mathcal{C}) - \sum_{i=1}^{\delta} \dim(\mathcal{C}_{J_i}) \right]$$

Theorem

Given a tree T and a mapping $\omega : [n] \to V(T)$, $\kappa(\mathcal{C}; T, \omega)$ is the minimum constraint complexity among all tree realizations on T with coordinate assignment ω.

Treewidth

For a linear code $\mathcal{C}$,

$$\begin{aligned} \kappa_{\text{tree}}(\mathcal{C}) &= \min_{(T,\omega)} \kappa(\mathcal{C}; T, \omega) \\ &= \min_{(T,\omega)} \max_{v \in V(T)} \left[\dim(\mathcal{C}) - \sum_{i=1}^{\delta} \dim(\mathcal{C}_{J_i}) \right] \end{aligned}$$

Treewidth

For a linear code $\mathcal{C}$,

$$\begin{aligned}\kappa_{\text{tree}}(\mathcal{C}) &= \min_{(T,\omega)} \kappa(\mathcal{C}; T, \omega) \\ &= \min_{(T,\omega)} \max_{v \in V(T)} \left[\dim(\mathcal{C}) - \sum_{i=1}^{\delta} \dim(\mathcal{C}_{J_i})\right]\end{aligned}$$

The minimization above may be taken over (T, ω) such that

- T is a cubic tree with n leaves ($n =$ blocklength of $\mathcal{C}$), and
- ω is a 1-1 assignment of coordinates of $\mathcal{C}$ to the leaves of T.

Generalization to Matroids

The definition of treewidth for a linear code is based heavily on the work of Forney (2001,2003).

Around 2005, Jim Geelen independently defined a notion of treewidth for matroids, as a generalization of a well-established definition of treewidth for graphs.

Remarkably, Geelen's definition of matroid treewidth, when applied to the special case of vector matroids, reduces precisely to the definition for linear codes.

What is a Matroid?

Definition

A matroid consists of a finite set E together with a function $r : 2^E \to \mathbb{Z}^+$ having the following properties:

(M1) $0 \leq r(A) \leq |A|$ for all $A \subseteq E$

(M2) if $A \subseteq B$, then $r(A) \leq r(B)$ [monotonicity]

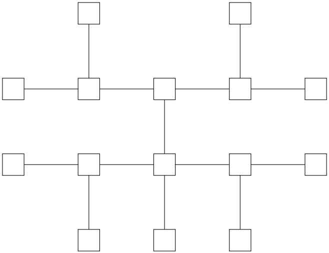
(M3) $r(A \cup B) + r(A \cap B) \leq r(A) + r(B)$ for all $A, B \subseteq E$ [submodularity]

The set E is called the ground set and the function r is called the rank function of the matroid.

Tree Decompositions of Matroids

A tree decomposition of a matroid $M = (E, r)$ consists of

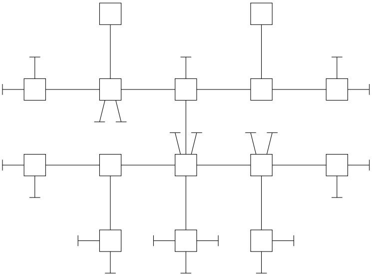
- a tree T



Tree Decompositions of Matroids

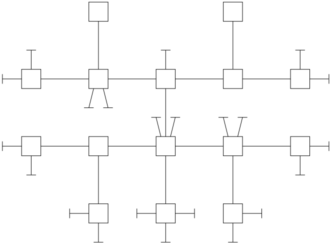
A tree decomposition of a matroid $M = (E, r)$ consists of

- a tree T, and
- a mapping $\omega : E \to V(T)$



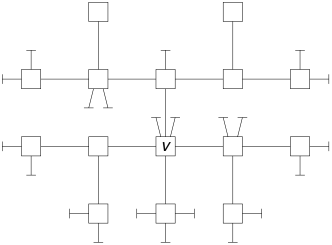
Width of a Tree Decomposition

Let (T, ω) be a tree decomposition of a matroid $M = (E, r)$.



Width of a Tree Decomposition

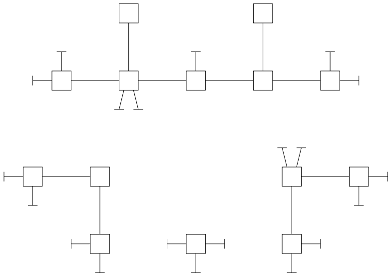
Let (T, ω) be a tree decomposition of a matroid $M = (E, r)$.



For a node $v \in V(T)$ with degree δ:

Width of a Tree Decomposition

Let (T, ω) be a tree decomposition of a matroid $M = (E, r)$.

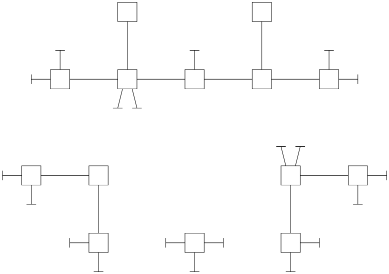


For a node $v \in V(T)$ with degree δ:

- the removal of v from T yields a graph whose components, $T_1, \ldots, T_\delta$, are subtrees of T
- for $i = 1, \ldots, \delta$, set $J_i = \omega^{-1}(V(T_i))$
- node-width(v) $:=$ $r(E) - \sum_{i=1}^{\delta} [r(E) - r(E \setminus J_i)]$

Width of a Tree Decomposition

Let (T, ω) be a tree decomposition of a matroid $M = (E, r)$.



For a node $v \in V(T)$ with degree δ:

- the removal of v from T yields a graph whose components, $T_1, \ldots, T_\delta$, are subtrees of T
- for $i = 1, \ldots, \delta$, set $J_i = \omega^{-1}(V(T_i))$
- node-width(v) $:= \; r(E) - \sum_{i=1}^{\delta} [r(E) - r(E \setminus J_i)]$

Then, width$(T, \omega) = \max_{v \in V(T)}$ node-width(v)

Matroid Treewidth

Definition (J.F. Geelen (unpublished); Hliněný and Whittle (2006))

The treewidth of M is the least width of any of its tree decompositions.

Matroid Treewidth

Definition (J.F. Geelen (unpublished); Hliněný and Whittle (2006))

The treewidth of M is the least width of any of its tree decompositions.

Hliněný and Whittle (2006,2008) showed that the definition, when applied to graphic matroids, reduces to the standard definition of graph treewidth.

Graph Treewidth

Let $\mathcal{G}$ be a graph with vertex set $V(\mathcal{G})$.

A tree decomposition of $\mathcal{G}$ consists of a tree T, and an ordered collection $\mathcal{V} = (V_x,\ x \in V(T))$ of subsets of $V(\mathcal{G})$, satisfying

- $\bigcup_{x \in V(T)} V_x = V$;
- for each $v \in V(\mathcal{G})$, the subgraph of T induced by $\{x \in V(T) : v \in V_x\}$ is connected; and
- for each pair of adjacent vertices $u, v \in V(\mathcal{G})$, we have $\{u, v\} \subseteq V_x$ for some $x \in V(T)$.

We then define $\mathrm{width}(T, \mathcal{V}) \triangleq \max_{x \in V(T)} |V_x| - 1$.

Graph Treewidth

Let $\mathcal{G}$ be a graph with vertex set $V(\mathcal{G})$.

A tree decomposition of $\mathcal{G}$ consists of a tree T, and an ordered collection $\mathcal{V} = (V_x,\ x \in V(T))$ of subsets of $V(\mathcal{G})$, satisfying

- $\bigcup_{x \in V(T)} V_x = V$;
- for each $v \in V(\mathcal{G})$, the subgraph of T induced by $\{x \in V(T) : v \in V_x\}$ is connected; and
- for each pair of adjacent vertices $u, v \in V(\mathcal{G})$, we have $\{u, v\} \subseteq V_x$ for some $x \in V(T)$.

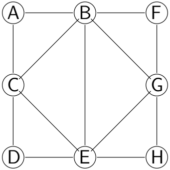
We then define $\mathrm{width}(T, \mathcal{V}) \triangleq \max_{x \in V(T)} |V_x| - 1$.

Definition (Robertson & Seymour (1983))

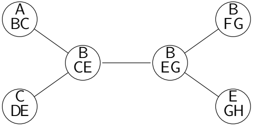
The treewidth of $\mathcal{G}$ is defined to be the least width of any tree decomposition of $\mathcal{G}$; denoted by $\kappa_{\text{tree}}(\mathcal{G})$.

Some Examples

- For any tree T, $\kappa_{\text{tree}}(T) = 1$.
- If $\mathcal{G}$ is a cycle on at least three vertices, then $\kappa_{\text{tree}}(\mathcal{G}) = 2$.
- The graph $\mathcal{G}$ shown below also has treewidth 2.



$\mathcal{G}$



An optimal tree decomposition of $\mathcal{G}$

Relating Graph and Code Treewidth (via Matroids)

Hliněný and Whittle (2006,2008) showed that the treewidth of a graph equals the treewidth of its cycle matroid.

The cycle matroid of a graph corresponds to its cut-set code.

Definition

The cut-set code of a graph $\mathcal{G} = (V, E)$ is the binary code $\mathcal{C}[\mathcal{G}]$ generated by the $|V| \times |E|$ vertex-edge incidence matrix of $\mathcal{G}$.

Relating Graph and Code Treewidth (via Matroids)

Hliněný and Whittle (2006,2008) showed that the treewidth of a graph equals the treewidth of its cycle matroid.

The cycle matroid of a graph corresponds to its cut-set code.

Definition

The cut-set code of a graph $\mathcal{G} = (V, E)$ is the binary code $\mathcal{C}[\mathcal{G}]$ generated by the $|V| \times |E|$ vertex-edge incidence matrix of $\mathcal{G}$.

Theorem (Hliněný and Whittle (2006,2008))

$$\kappa_{tree}(\mathcal{G}) = \kappa_{tree}(\mathcal{C}[\mathcal{G}]) \text{ for any graph } \mathcal{G}.$$

Relating Graph and Code Treewidth (via Matroids)

Hliněný and Whittle (2006,2008) showed that the treewidth of a graph equals the treewidth of its cycle matroid.

The cycle matroid of a graph corresponds to its cut-set code.

Definition

The cut-set code of a graph $\mathcal{G} = (V, E)$ is the binary code $\mathcal{C}[\mathcal{G}]$ generated by the $|V| \times |E|$ vertex-edge incidence matrix of $\mathcal{G}$.

Theorem (Hliněný and Whittle (2006,2008))

$\kappa_{tree}(\mathcal{G}) = \kappa_{tree}(\mathcal{C}[\mathcal{G}])$ *for any graph* $\mathcal{G}$.

Computing the treewidth of a graph is NP-hard.

[Arnborg, Corneil and Proskurowski (1987)]

Hence, computing the treewidth of a linear code is also NP-hard.

Treewidth of MDS and Reed-Muller Codes

Theorem (K. and Thangaraj (2011))

For an $[n, k]$ MDS code,

$$\textit{treewidth} = \textit{trelliswidth} = \min(k, n - k + 1).$$

For the Reed-Muller code $RM(r, m)$,

$$\textit{treewidth} = \textit{trelliswidth} = \begin{cases} \sum_{j=0}^{r} \binom{m-2j-1}{r-j} & \text{if } m \geq 2r+1 \\ 1 + \sum_{j=0}^{m-r-1} \binom{m-2j-1}{r-j} & \text{if } m < 2r+1 \end{cases}$$

Proof Strategy

Recall that for a linear code $\mathcal{C}$,

$$\begin{aligned}\kappa_{\text{tree}}(\mathcal{C}) &= \min_{(T,\omega)} \kappa(\mathcal{C}; T, \omega) \\ &= \min_{(T,\omega)} \max_{v \in V(T)} \left[\dim(\mathcal{C}) - \sum_{i=1}^{\delta} \dim(\mathcal{C}_{J_i}) \right]\end{aligned}$$

and the minimization above may be taken over (T, ω) such that

- T is a cubic tree with n leaves ($n =$ blocklength of $\mathcal{C}$), and
- ω is a 1-1 assignment of coordinates of $\mathcal{C}$ to the leaves of T.

Proof Strategy

Recall that for a linear code $\mathcal{C}$,

$$\begin{aligned}\kappa_{\text{tree}}(\mathcal{C}) &= \min_{(T,\omega)} \kappa(\mathcal{C}; T, \omega) \\ &= \min_{(T,\omega)} \max_{v \in V(T)} \left[\dim(\mathcal{C}) - \sum_{i=1}^{\delta} \dim(\mathcal{C}_{J_i})\right]\end{aligned}$$

and the minimization above may be taken over (T, ω) such that

- T is a cubic tree with n leaves ($n =$ blocklength of $\mathcal{C}$), and
- ω is a 1-1 assignment of coordinates of $\mathcal{C}$ to the leaves of T.

Our proof strategy is to

- first compute κ_{trellis} for MDS and RM codes;
- then show that if $\mathcal{C}$ is an MDS or RM code, then for any (T, ω) as above, we have $\kappa(\mathcal{C}; T, \omega) \geq \kappa_{\text{trellis}}(\mathcal{C})$.

Computation of κ_{trellis}

Computing $\kappa_{\text{trellis}}(\mathcal{C})$ is a matter of finding a coordinate ordering of $\mathcal{C}$ that yields an optimal trellis realization.

Computation of κ_{trellis}

Computing $\kappa_{\text{trellis}}(\mathcal{C})$ is a matter of finding a coordinate ordering of $\mathcal{C}$ that yields an optimal trellis realization.

Dimension of local constraint code at node h $(1 \leq h \leq n)$ is

$$\dim(\mathcal{C}) - \dim(\mathcal{C}_{\{1,2,\ldots,h-1\}}) - \dim(\mathcal{C}_{\{h+1,h+2,\ldots,n\}}).$$

Computation of κ_{trellis}

Computing $\kappa_{\text{trellis}}(\mathcal{C})$ is a matter of finding a coordinate ordering of $\mathcal{C}$ that yields an optimal trellis realization.

Dimension of local constraint code at node h $(1 \leq h \leq n)$ is

$$\dim(\mathcal{C}) - \dim(\mathcal{C}_{\{1,2,\ldots,h-1\}}) - \dim(\mathcal{C}_{\{h+1,h+2,\ldots,n\}}).$$

- For an $[n, k]$ MDS code, $\dim(\mathcal{C}_J)$ depends only on $|J|$:

  $$\dim(\mathcal{C}_J) = \max\{0, |J| - (n-k)\}$$

  Hence, constraint complexity of trellis realization is independent of coordinate order.

  Routine computations show $\kappa_{\text{trellis}} = \min\{k, n-k+1\}$.

Computation of κ_{trellis} for RM Codes

For RM codes,

- an optimal coordinate ordering has been determined by Kasami et al. (1993);
- methods developed by Blackmore and Norton (2000) easily yield an expression for κ_{trellis}.

Showing $\kappa(\mathcal{C}; T, \omega) \geq \kappa_{\text{trellis}}$

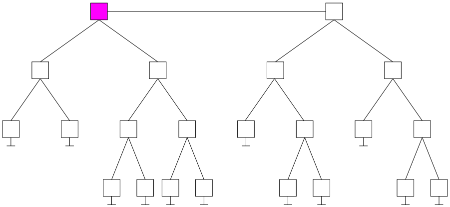
$$\kappa(\mathcal{C}; T, \omega) = \max_{v \in V(T)} \underbrace{\left[\dim(\mathcal{C}) - \sum_{i=1}^{\delta} \dim(\mathcal{C}_{J_i})\right]}_{\kappa_v}$$

Let $\mathcal{C}$ be an MDS or RM code, and let T be any cubic tree.

It can be shown that there exists a vertex $v \in V(T)$ such that no matter what the coordinate assignment ω, we have $\kappa_v \geq \kappa_{\text{trellis}}$.

Hence, $\kappa(\mathcal{C}; T, \omega) \geq \kappa_{\text{trellis}}$ for any cubic tree T and coord map ω.

Choice of v for an MDS Code



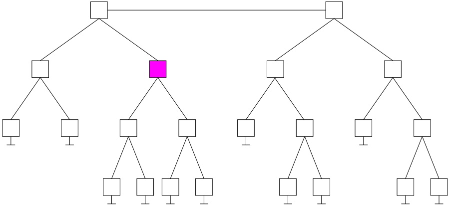
Theorem (C. Jordan (1869))

In any tree T with n leaves, there exists a node v such that each component of $T - v$ has at most $n/2$ leaves.

A node v as above is called a centroid of the tree.
There can be at most two centroids in a tree.

If $\mathcal{C}$ is an MDS code, and T any cubic tree, then taking v to be a centroid of T, we are guaranteed $\kappa_v \geq \kappa_{\text{trellis}}(\mathcal{C})$.

Choice of v for an RM Code



For an internal node v in a cubic tree T, let $n_1 \leq n_2 \leq n_3$ denote the number of leaves in the three components of $T - v$.

Theorem (folklore?)

In any cubic tree T with n leaves, there exists an internal node v such that $n/2 \leq n_3 \leq 2n/3$.

If $\mathcal{C}$ is an RM code, and T any cubic tree, then among the nodes satisfying the theorem, take v to be one with largest n_3.
For this choice of v, we have $\kappa_v \geq \kappa_{\text{trellis}}(\mathcal{C})$.

Some Open Problems

- What other NP-hard problems for codes (e.g., computing minimum distance) become tractable for codes of bounded treewidth?

Some Open Problems

- What other NP-hard problems for codes (e.g., computing minimum distance) become tractable for codes of bounded treewidth?
- Is it true that $|\kappa_{\text{tree}}(\mathcal{C}^{\perp}) - \kappa_{\text{tree}}(\mathcal{C})| \leq 1$ for any code $\mathcal{C}$?

  It can be shown that $\frac{1}{2}\,\kappa_{\text{tree}}(\mathcal{C}) \leq \kappa_{\text{tree}}(\mathcal{C}^{\perp}) \leq 2\,\kappa_{\text{tree}}(\mathcal{C})$.

Some Open Problems

- What other NP-hard problems for codes (e.g., computing minimum distance) become tractable for codes of bounded treewidth?
- Is it true that $|\kappa_{\text{tree}}(\mathcal{C}^{\perp}) - \kappa_{\text{tree}}(\mathcal{C})| \leq 1$ for any code $\mathcal{C}$?
  It can be shown that $\frac{1}{2}\,\kappa_{\text{tree}}(\mathcal{C}) \leq \kappa_{\text{tree}}(\mathcal{C}^{\perp}) \leq 2\,\kappa_{\text{tree}}(\mathcal{C})$.
- Can the treewidth of a linear code be efficiently approximated within some constant factor?

References

[1] N. Kashyap, "On Minimal Tree Realizations of Linear Codes," *IEEE Trans. Inform. Theory*, vol. 55, no. 8, pp. 3501–3519, Aug. 2009.

[2] N. Kashyap, "Constraint Complexity of Realizations of Linear Codes on Arbitrary Graphs," *IEEE Trans. Inform. Theory*, vol. 55, no. 11, pp. 4864–4877, Nov. 2009.

[3] N. Kashyap and A. Thangaraj, "The Treewidth of MDS and Reed-Muller Codes," arXiv:1102.2734, Feb. 2011.