Trellises, Decision Diagrams and Factor Graphs

John Lafferty

School of Computer Science
Carnegie Mellon University

Joint work with:

Alexander Vardy

Acknowledgements: Randy Bryant and Ed Clarke

Outline

- Background
- BDD-Trellis correspondence
- Alternate representations
- Projection graphs

Remarkably Parallel Histories

- Decision diagrams: Idea recorded in early papers of Lee (1959) and Akers (1978)

- Code trellises: Used as conceptual device for convolutional codes (Forney, 1967) and block codes (Bahl, Cocke, Jelinek, and Raviv, 1974)

- Bryant's 1986 paper initiated a great deal of research on decision diagram representations of Boolean functions.

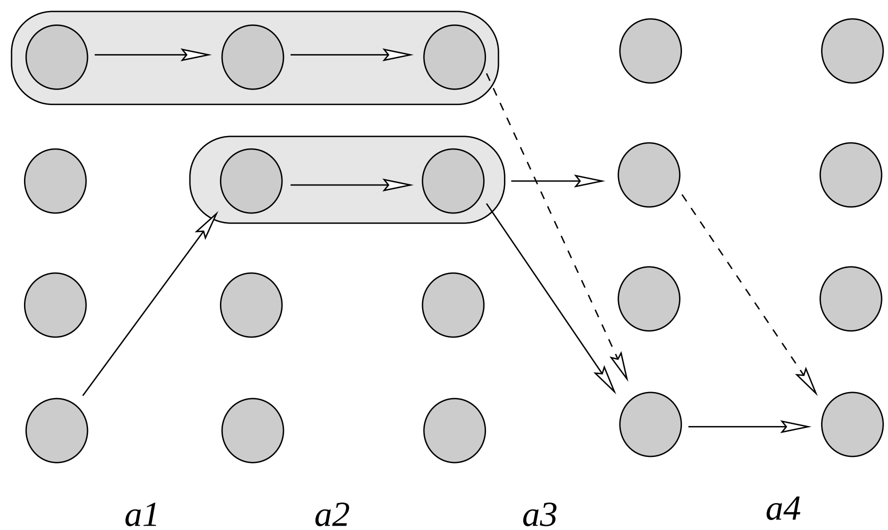
- Papers by Forney (1988) and Muder (1988) led to resurgence of interest in code trellises and a thorough study of their properties.

BCJR Paper

- In 1971–1972 Bahl and Jelinek were working on various aspects of convolutional codes. BCJR was a natural outgrowth; introduction of the minimal trellis was incidental.
- Bitwise optimal decoding algorithm was “academic”
- A reviewer pointed out the related work by L. Baum on the convergence properties of forward-backward algorithm
- Closely related work on speech recognition came shortly after...

Visualizing Hidden Markov Models

By 1974 the IBM group was pioneering a new approach to speech recognition using HMMs and the fundamental notions of information theory.



- Shannon: *"The states will correspond to the "residue of influence" from preceding letters"*

Training and Decoding

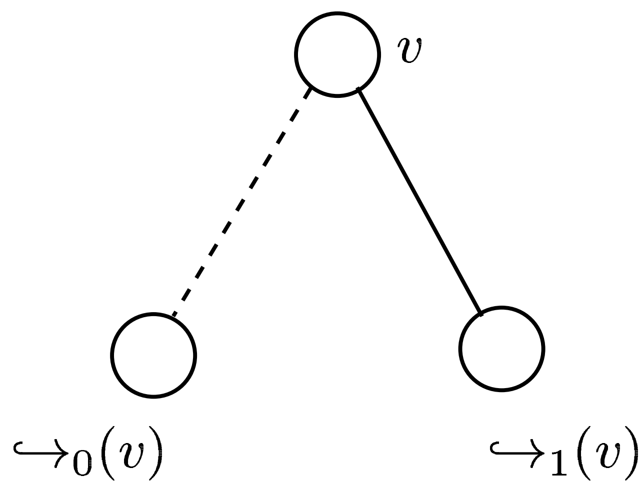
- The Baum-Welch algorithm for training HMMs became the centerpiece of statistical speech processing.
- Large state space and non-uniform prior on “codewords” make Viterbi decoding impractical for speech recognition systems.
- At IBM a version of Jelinek’s *stack decoding* method, conceived of for convolutional codes, was adapted to the decoding problem.
- Similar to A^* and beam search methods — another instance of the striking parallels between AI and information theory.

Connecting Trellises and Decision Diagrams

- The coding and verification communities have been been working independently on closely related problems.
- It's natural to expect that binary decision diagrams and minimal trellises are closely related.
- We've established the correspondence rigorously and have begun to explore its consequences.

Terminology for OBDDs

- Each nonterminal vertex v is labeled by a function variable $\mathsf{var}(v)$ and has two outgoing edges, denoted $\hookrightarrow_0(v)$ and $\hookrightarrow_1(v)$.



Terminology for OBDDs (cont)

- A nonterminal v is said to be a *redundant test* if $\hookrightarrow_0(v) = \hookrightarrow_1(v)$.
- Redundant tests may be *removed*, without altering the function being represented, by deleting v and redirecting all incoming edges to $\hookrightarrow_0(v)$.
- Two nonterminals u and v are said to be *duplicate* if $\hookrightarrow_0(v) = \hookrightarrow_0(u)$, $\hookrightarrow_1(v) = \hookrightarrow_1(u)$, and $\mathsf{var}(v) = \mathsf{var}(u)$.
- Duplicate nonterminals can be *merged* by deleting one of the two vertices and redirecting all incoming edges to the other vertex.

Terminology for OBDDs (cont)

- The *positive cofactor* f_x of a Boolean function f is the function obtained by replacing variable x by the value 1.
- The *negative cofactor* is the function $f_{\overline{x}}$ that replaces x by the value 0.
- The *Shannon expansion* is the identity $f \;=\; \overline{x} \cdot f_{\overline{x}} \;+\; x \cdot f_x$.
- Essential property of OBDDs: the cofactor operations distribute through the Boolean operations: $(f \oplus g)_x = f_x \oplus g_x$.
- OBDDs are essentially graphical representations of the Shannon expansion.

Terminology for Trellises

- A trellis T is *proper* if the edges beginning at a vertex of T are labeled distinctly.
- A trellis $T_{\mathbb{C}}$ is the *minimal proper trellis* for $\mathbb{C}$ if the number of vertices at each time i in $T_{\mathbb{C}}$ is less than or equal to the number of vertices at time i in any other proper trellis for $\mathbb{C}$.

Construction A

Input: Boolean function $f(x_1, \ldots, x_n)$ and variable ordering $x_1 \prec \cdots \prec x_n$.

Output: Ordered binary decision diagram $\mathcal{D}_f$ for $f(x_1, \ldots, x_n)$.

Algorithm: Starting with the full binary decision tree for $f(x_1, \ldots, x_n)$:

Step 1. Merge duplicate terminals.

Step 2. Merge all duplicate nonterminals.

Step 3. Remove all redundant tests.

Iterate steps **2** and **3** until no duplicate nonterminals or redundant tests remain.

Working Example

We'll use the $[5, 2, 3]$ linear code

$$\mathbb{C} \;=\; \{00000, 11010, 01101, 10111\}$$

given by the parity check matrix

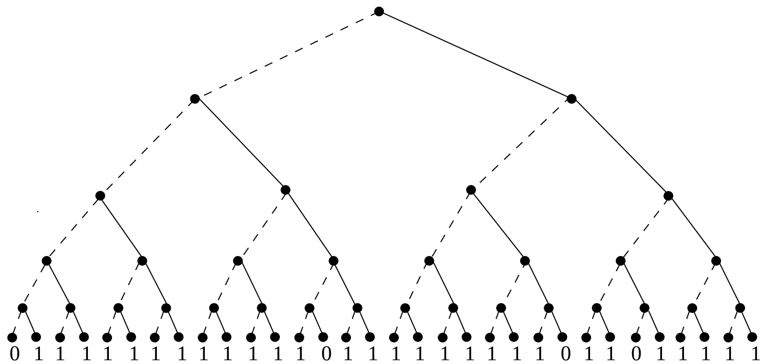
$$H = \begin{bmatrix} 1 & 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 & 1 \end{bmatrix}$$

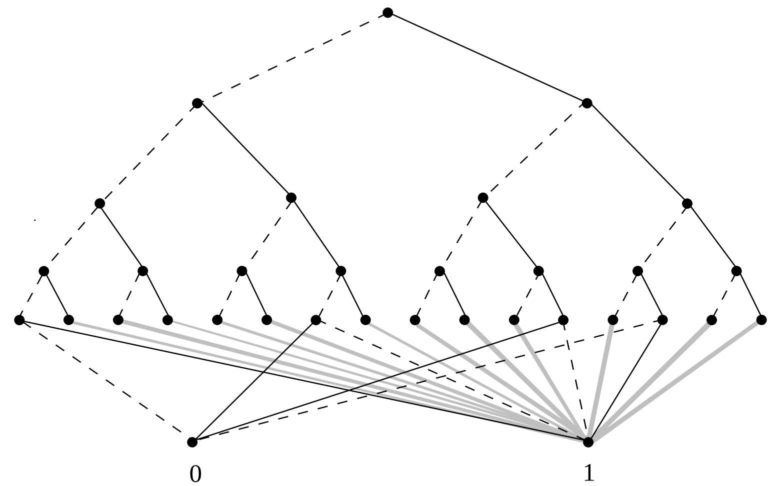
This will also be thought of as the Boolean function

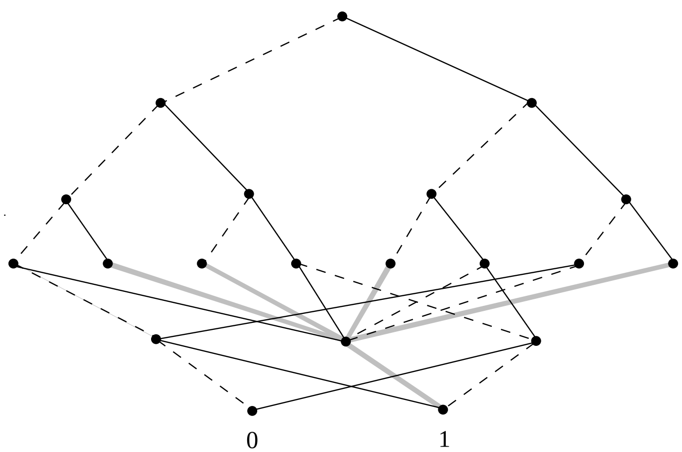
$$f_{\mathbb{C}}(\vec{x}) \stackrel{\text{def}}{=} \begin{cases} 0 & \text{if } (x_1, \ldots, x_n) \in \mathbb{C} \\ 1 & \text{otherwise} \end{cases}$$

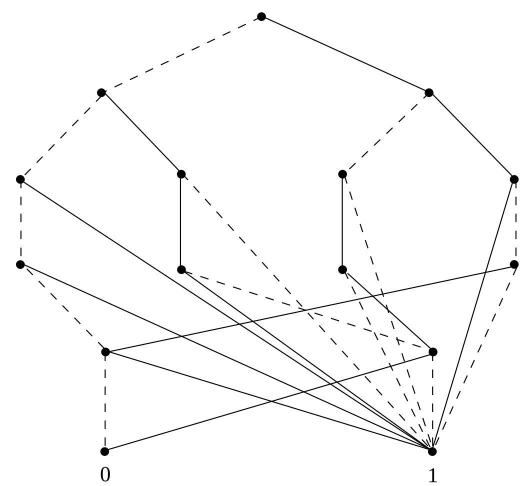
$$f_{\mathbb{C}}(\vec{x}) = (x_1 \oplus x_2 \oplus x_3) + (x_1 \oplus x_4) + (x_1 \oplus x_2 \oplus x_5)$$

Node-Merging Construction of OBDDs

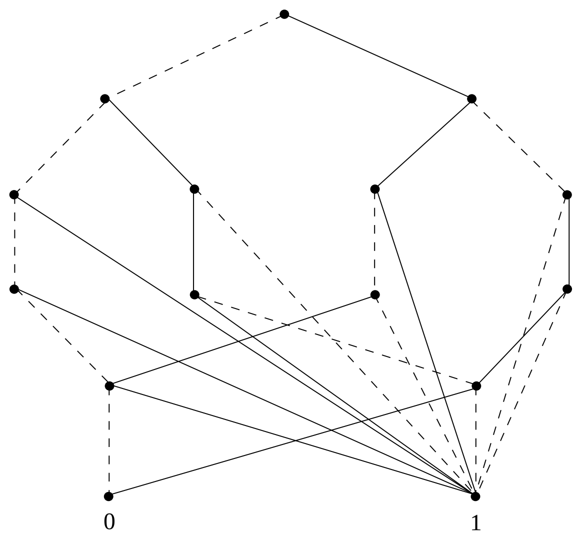


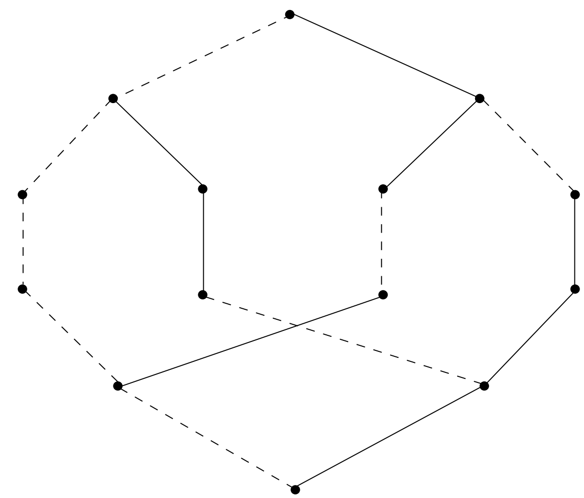






Single-Terminal Decision Diagrams





$$f_{\mathbb{C}} = (x_1 \oplus x_2 \oplus x_3) + (x_1 \oplus x_4) + (x_1 \oplus x_2 \oplus x_5)$$

$$\mathbb{C} = \{00000, 11010, 01101, 10111\}$$

Construction B

Input: Boolean function $f(x_1, \ldots, x_n)$ and variable ordering $x_1 \prec \cdots \prec x_n$.

Output: Ordered binary decision diagram $\mathcal{D}_f$ for $f(x_1, \ldots, x_n)$.

Algorithm: Starting with the full binary decision tree for $f(x_1, \ldots, x_n)$:

Step 1. Merge duplicate terminals.

Step X. Prune away the 1-terminal.

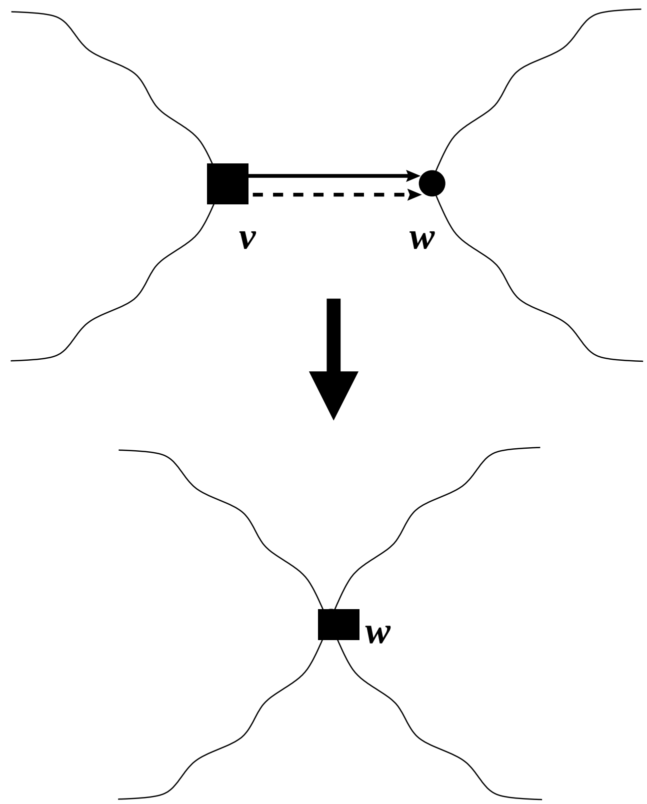
Step 2. Merge all duplicate nonterminals.

Step 3. Remove all redundant tests.

Iterate steps **2** and **3** until no duplicate nonterminals or redundant tests remain.

Minimal Distance Restriction

In Construction B, redundant tests destroy trellis structure:



This cannot happen if minimum distance satisfies $d(\mathbb{C}) > 1$.

OBDD-Trellis Connection

Theorem. Let $\mathbb{C}$ be an arbitrary binary code with minimum distance $d > 1$. Then the single-terminal OBDD for the Boolean function $f_{\mathbb{C}}$ is isomorphic to the unique minimal proper trellis for $\mathbb{C}$.

$T_{\mathbb{C}} \approx \mathcal{D}_{f_{\mathbb{C}}}$

We'll prove that the minimal trellis $T_{\mathbb{C}}$ is isomorphic to the single-terminal OBDD $\mathcal{D}_{f_{\mathbb{C}}}$ for the off-set function $f_{\mathbb{C}}$ built using Construction B.

The *past at time i* is

$$\mathcal{P}_i(\mathbb{C}) \stackrel{\text{def}}{=} \Big\{(c_1, c_2, \ldots, c_i) \;:\; (c_1, \ldots, c_i, c_{i+1}, \ldots, c_n) \in \mathbb{C} \text{ for some } c_{i+1}, \ldots, c_n \in \mathbb{F}_2\Big\}$$

The *future of $c \in \mathcal{P}_i(\mathbb{C})$* is

$$\mathcal{F}(c) = \{x \in \mathbb{F}_2^{n-i} : (c, x) \in \mathbb{C}\}$$

We say that $c_1, c_2 \in \mathcal{P}_i(\mathbb{C})$ are *future-equivalent* if $\mathcal{F}(c_1) = \mathcal{F}(c_2)$.

$T_{\mathbb{C}} \approx \mathcal{D}_{f_{\mathbb{C}}}$

Lemma (Muder, 1988). A proper trellis T for $\mathbb{C}$ is minimal if and only if for all $i = 1, 2, \ldots, n-1$, the number of vertices at time i in T is equal to the number of future-equivalence classes.

Equivalently, for $v \in V_i$ define $\mathcal{F}_T(v) \subset \mathbb{F}_2^{n-i}$ by

$$\mathcal{F}_T(v) \stackrel{\text{def}}{=} \left\{ x \ : \ x \text{ is a sequence of edge labels along a path in } T \text{ starting at } v \right\}$$

Then a proper trellis T is minimal if and only if for all $i = 1, 2, \ldots, n-1$ and for every pair of vertices $v, v' \in V_i$, we have $\mathcal{F}_T(v) \neq \mathcal{F}_T(v')$.

$T_{\mathbb{C}} \approx \mathcal{D}_{f_{\mathbb{C}}}$

- If $d(\mathbb{C}) > 1$, there can be no redundant tests, so the single-terminal OBDD for $\mathbb{C}$ is a proper trellis.

- Assume that T is *not* minimal. Then there is a pair of distinct vertices $v, v' \in V_i$ with $\mathcal{F}_T(v) = \mathcal{F}_T(v')$

- By Construction B, at least one of $\{\hookrightarrow_0(v), \hookrightarrow_0(v')\}$ or $\{\hookrightarrow_1(v), \hookrightarrow_1(v')\}$ must be a pair of distinct vertices.

- We then obtain distinct vertices $u, u' \in V_{i+1}$ with $\mathcal{F}_T(u) = \mathcal{F}_T(u')$.

- Iterating, we arrive at a contradiction, since V_n is a single vertex.

Alternate Terminology

We can view $\mathbb{C}$ as a regular set in $\mathbb{F}_2^n$. Its minimal deterministic finite-state accepting automaton is the same as the minimal proper trellis.

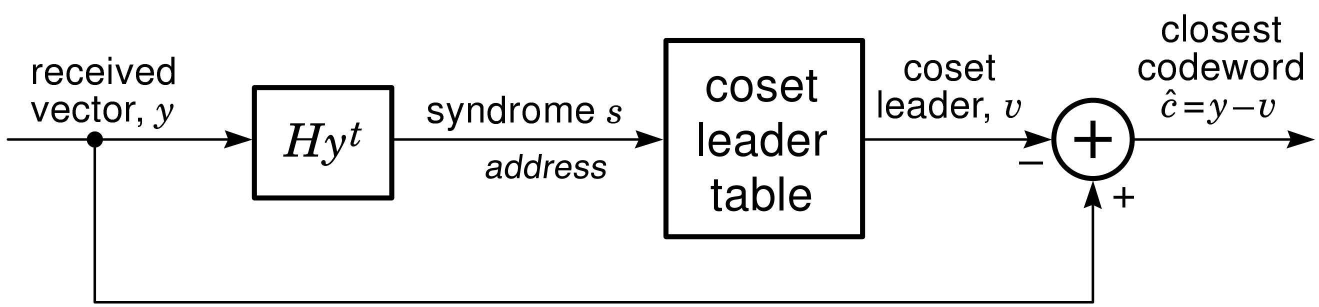
Mentioned in the *“Multilingual Dictionary”* of system theory, coding theory, symbolic dynamics, and automata theory (Forney *et al.* 1995).

Alternate Representations and Transfer of Ideas

- Multi-terminal and spectral decision diagrams
- Functional decision diagrams
- Lower bounds
- For further results and observations, see (Lafferty and Vardy, 1999 [DS #23]).

Multi-Terminal Decision Diagrams

- Represent functions $f : \{0,1\}^n \longrightarrow \mathcal{S}$ for finite set $\mathcal{S}$.
- May have many terminals. Can be used to efficiently represent matrices (Clarke *et al.*, 1993)
- We can use them to design a new data structure for *standard array decoding:*



Syndrome Decision Diagrams

- construct a multi-terminal BDD for the function $h_{\mathbb{C}}(x_1, \ldots, x_n) = Hx^t$.

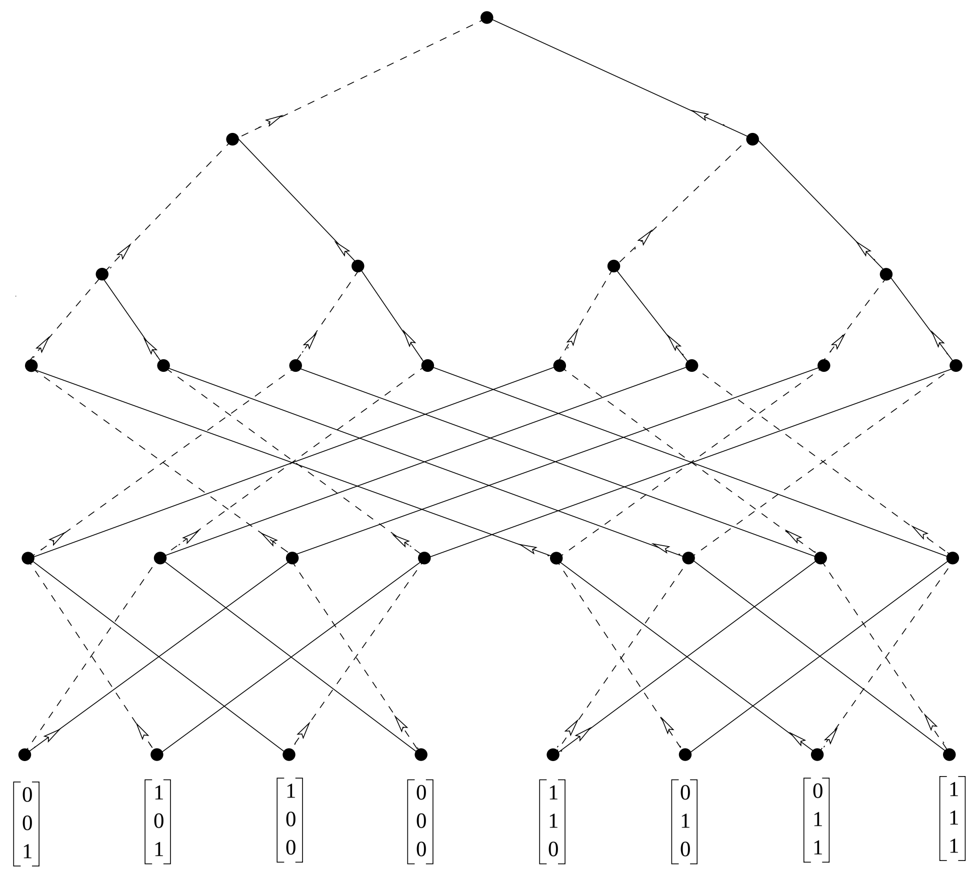
- Use a procedure analogous to the BCJR construction:

$$V_i \stackrel{\text{def}}{=} \left\{ x_1 h_1 + \cdots + x_i h_i \ : \ (x_1, \ldots, x_i) \in \mathbb{F}_2^i \right\}$$

- Carry out dynamic programming during construction to calculate smallest weight path to a vertex.

- Similar constructions have been given (Kschischang, 1996; Ytrehus, 1997)

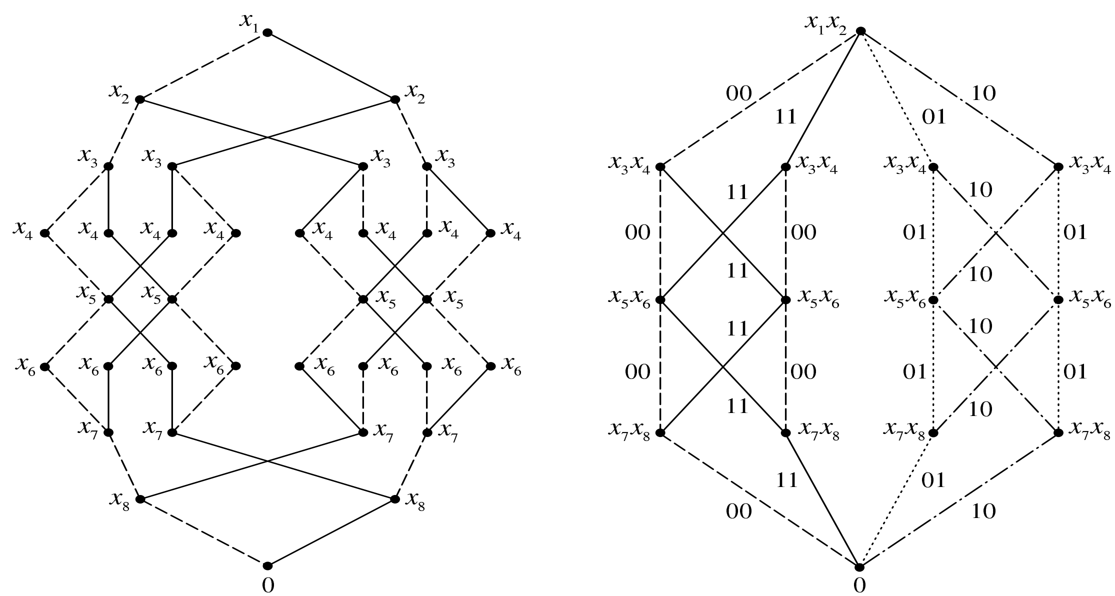
Syndrome Decision Diagrams



The syndrome decision diagram for $\mathbb{C} = \{00000, 11010, 01101, 10111\}$

Gives linear time decoding algorithm.

Sectionalization



a. b.

- Main results due to Lafourcade and Vardy (1996), using dynamic programming methods.
- Related techniques for decision diagrams?

MTBDDs and Spectral Decision Diagrams

Want to represent a function

$$f : G_1 \times G_2 \times \cdots \times G_n \longrightarrow \mathbb{F}_2$$

- Usual case for Boolean functions is $G_i = \mathbb{F}_2$
- Idea: Consider different factorizations of the product and use Fourier transform to represent function.

Spectral Decision Diagrams

- A *representation* ρ of a group G is an assignment of an invertible matrix $\rho\,(s)$ to each group element $s \in G$ so that $\rho\,(s\,t) = \rho(s)\ \rho(t)$
- The representation is *irreducible* if it has no notrivial subrepresentations
- Degrees satisfy $\sum_\rho d_\rho^2 = |\,G\,|$
- *Fourier transform* of f at a representation ρ is the matrix

$$\widehat{f}(\rho) = \sum_{s \in G} f(s)\,\rho\,(s).$$

Spectral Decision Diagrams

- Represent a function $f : G_1 \times \cdots \times G_n \longrightarrow \mathbb{F}_2$ in terms of a *Fourier decision tree*: each path from root to leaf specfies a representation $\rho_1 \otimes \cdots \otimes \rho_n$

- Leaf labeled by the *matrix* $\widehat{f}(\rho_1 \otimes \cdots \otimes \rho_n)$.

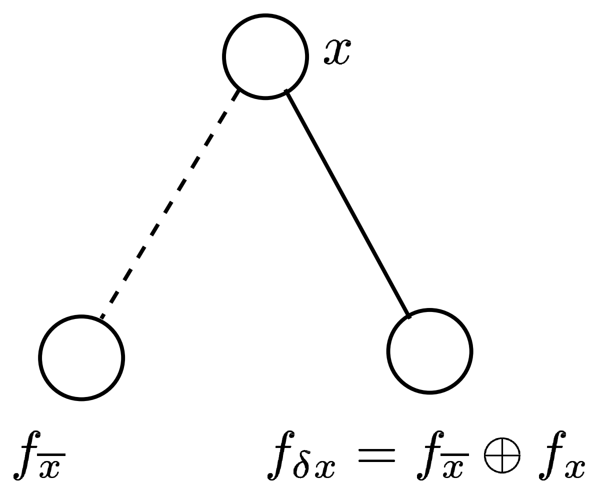
- Amount of data stored at the leaves is the same, since $|G| = \sum_\rho d_\rho^2$

- E.g., the group $\mathbb{Z}_2^4$ can be represented as $\mathbb{Z}_4 \times \mathbb{Z}_4$ or as $\mathbb{Z}_2 \times \mathbb{Q}_2$. Three-bit multiplier on $\mathbb{Z}_2^6$ represented using $\mathbb{Q}_2 \times \mathbb{Q}_2$ yields smaller decision diagram (Stanković, 1999)

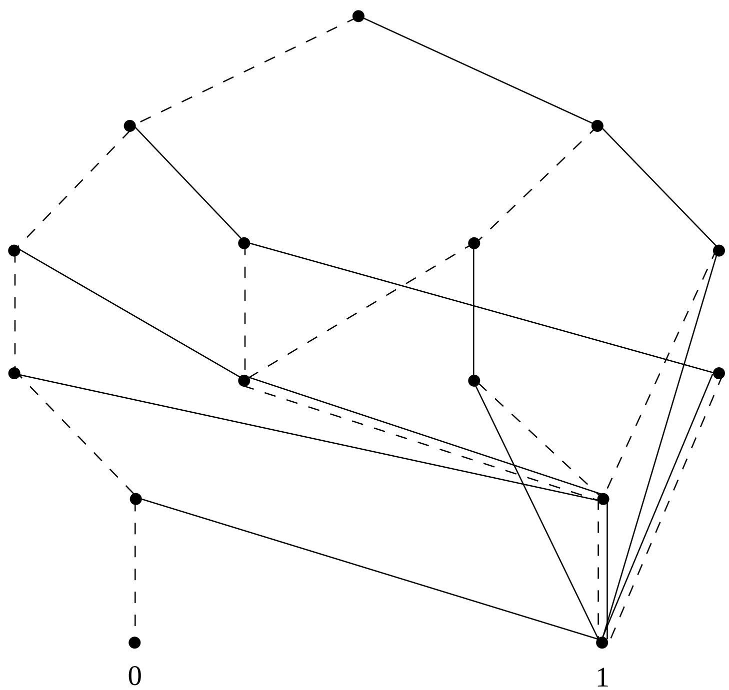
Functional Decision Diagrams

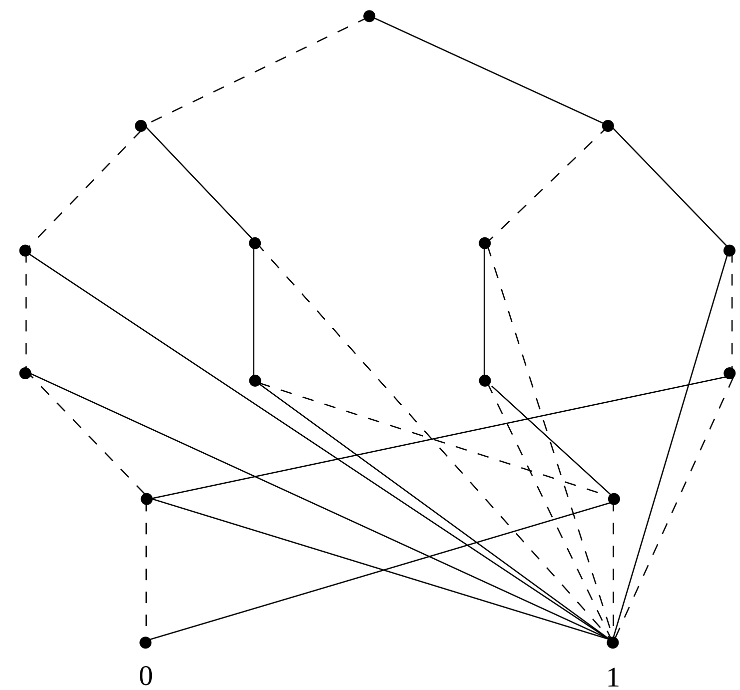
- Change representation of nodes to use *Reed-Muller* or *negative Davio* expansion:

$$f = f_{\overline{x}} \oplus (x \cdot f_{\delta x})$$



Functional Decision Diagrams





The OFDD and OBDD for $\mathbb{C} = \{00000, 11010, 01101, 10111\}$

Functional Decision Diagrams

- Many properties in common with OBDDs (Kebschull *et al.*, 1992)
- Representation is canonical
- Can be exponentially smaller—or larger—than OBDDs
- Are they useful for representing error-correcting codes?

Functional Decision Diagrams

Can we carry out a forward-backward type of algorithm on an OFDD?

- Restrict to BSC:

$$p_\theta(y|x) \;=\; \theta^{d(x,y)}\,(1-\theta)^{n-d(x,y)}$$

- Need to compute

$$\mathcal{S}_0(x_i) \stackrel{\text{def}}{=} \sum_{\substack{x\in\mathbb{C}\\ x_i=0}} p_\theta(y|x) \qquad \text{and} \qquad \mathcal{S}_1(x_i) \stackrel{\text{def}}{=} \sum_{\substack{x\in\mathbb{C}\\ x_i=1}} p_\theta(y|x)$$

- Algorithm must work for any $\theta \in [0,1]$.

Functional Decision Diagrams

This approach will not work:

- Suppose we can compute $\mathcal{S}_0(x_i)$ and $\mathcal{S}_1(x_i)$.
- Setting $\theta = 0.5$ we can then compute

$$\mathcal{S}_0(x_i) + \mathcal{S}_1(x_i) = \frac{|\mathbb{C}|}{2^n}$$

- However, the problem of computing $\left|\mathbb{C}_f\right|$ using OFDDs is #P-complete (Wercherner, *et al.*, 1996 – reduction from 3CNF).

Conclusion: FDDs are not well-suited to coding calculations

Lower Bounds – From Coding Theory

For a subset of indices $\mathcal{J} = j_1, j_2, \ldots j_m$, define a random variable $\mathcal{X}_{\mathcal{J}}$:

$$\Pr\{\mathcal{X}_{\mathcal{J}} = (a_1, \ldots, a_m)\} \stackrel{\text{def}}{=} \frac{\left| \mathbb{C}_{f|_{x_{j_1}, \cdots, x_{j_m} = a_1, \cdots, a_m}} \right|}{\left| \mathbb{C}_f \right|}$$

Now define the *entropy profile* $H_1(f), H_2(f), \ldots H_n(f)$:

$$H_i(f) \stackrel{\text{def}}{=} \min_{\mathcal{J}} H(\mathcal{X}_{\mathcal{J}})$$

where the minimum is taken over all index subsets of size i.

Lower Bounds – From Coding Theory

Theorem (Reuven and Be'ery, 1998). Let $f(x_1, \ldots, x_n)$ be a Boolean function such that $d(\mathbb{C}_f) > 1$. Then the number of vertices at level i in the OBDD for $f(x_1, \ldots, x_n)$ is bounded from below by

$$\frac{2^{H_i(f)} \cdot 2^{H_{n-i}(f)}}{|\,\mathbb{C}_f\,|}$$

- Recent work at CMU attempts to find a good variable ordering by using machine learning techniques based on mutual information (decision tree induction) to find groups of variables that are highly "coupled."

Lower Bounds – From Formal Verification

- (Thathachar, 1998) establishes lower bounds for many functions of practical interest, across a large class of decision diagram representations.
- Introduces a general graphical model called *binary linear diagrams* that includes OBDDs, OFDDs, MTBDDs, etc.
- Transforms decision diagram to an automaton on bitstrings, and uses *fooling set* arguments to lower bound rank of transition matrix.

Automata and Rank Bounds

- Let π denote an ordering $x_{\pi(1)} \prec x_{\pi(2)} \cdots \prec x_{\pi(n)}$ of the variables.
- Let M_k^π denote the $2^k \times 2^{n-k}$ matrix with rows and columns indexed by bit strings, whose (s, t) entry given by $f(s \cdot t)$.

Theorem. The number of vertices r in any decision diagram that computes f is bounded from below by

$$r \geq \min_{\pi} \max_{k} \text{rank}(M_k^\pi)$$

Fooling Set Lower Bounds

Divide the variables $x_{\pi(1)} \prec x_{\pi(2)} \cdots \prec x_{\pi(n)}$ into two pieces, L and R.
A *fooling set* is a set of pairs $\mathcal{S} = \{(s,t)\}$ satisfying

- $f(s \cdot t) = 0$ for each $(s,t) \in \mathcal{S}$.
- For (s,t) and $(s',t') \in \mathcal{S}$: $f(s \cdot t') \neq f(s' \cdot t)$.

Theorem. (Dietzfelbinger *et al.*, 1994) If $\mathcal{S}$ is a fooling set for f of size s then

$$\text{rank}\left(M^{L:R}\right) \;\geq\; \sqrt{s} - 1$$

Fooling Set Lower Bounds (cont)

- (Thathachar, 1998) finds explicit fooling sets, and then uses this result to get lower founds for a wide range of functions of practical interest.

- Bounds hold for broad class of decision diagram representations

- Extends a line of work initiated by Bryant (1991), who showed how to find fooling sets for the middle bit of the product of two n-bit numbers.

From Decision Diagrams to Factor Graphs

- OBDDs and trellises are powerful tools—but exponential blowup limits their usefulness
- In LDPCCs, expander codes, and turbo codes, the graph comes from the definition of the code itself
- How might we begin to apply iterative decoding techniques to more general classes of codes / Boolean functions?

Fighting Randomness with Randomness

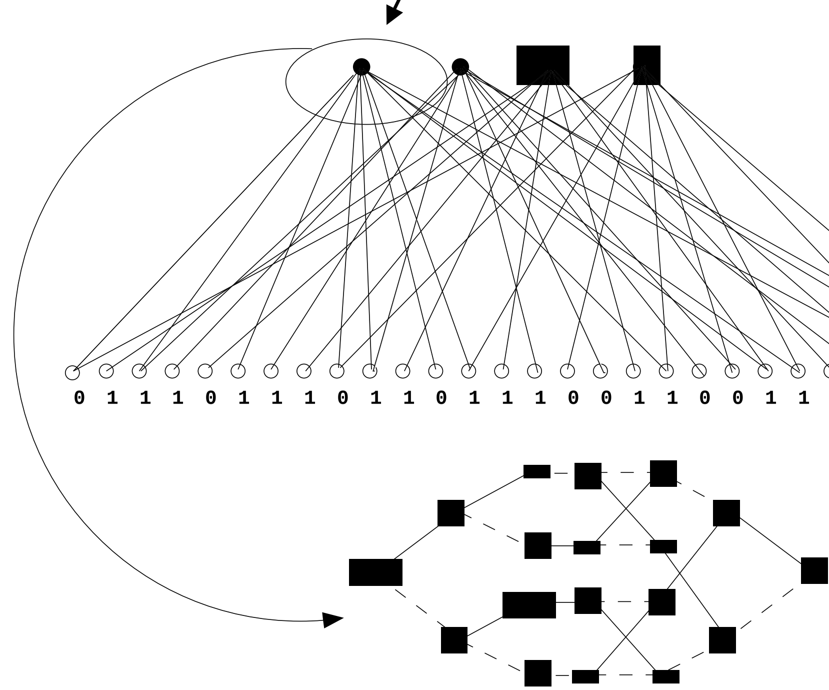
- For many applications, the “code” is given to us—e.g. in formal verification and statistical inference

- Idea: randomize the *decoder* rather than the code

- Randomization is the dominant theme in recent work in the theoretical CS community on approximate NN search (Kleinberg 1997, Kushilevitz *et al.*, 1998...)

Projection Decoding

$$G_{k\times n} = \begin{pmatrix} 1\,0\,1\,0\,0\,1\,1\,1\,0\,0\,\overbrace{1\,0\,1\,0\,0\,1\,1\,1\,0}^{\mathcal{S}_{k\times k+\varepsilon}}\,0\,1\,1\,1\,0\,0\,1\,0\,1 \\ 0\,1\,1\,0\,0\,0\,1\,0\,0\,0\,1\,0\,1\,0\,0\,1\,1\,0\,1\,0\,1\,0\,1\,1\,1\,0\,0\,1 \\ \vdots \qquad\qquad\qquad \vdots \\ 0\,1\,0\,1\,1\,1\,0\,1\,1\,1\,1\,1\,0\,1\,0\,0\,1\,1\,0\,1\,0\,0\,0\,0\,1\,1\,0\,0 \end{pmatrix}$$



How Many Projections are Required?

$$Prob\left(\text{dim } \left(\cap_{i=1}^{m}\mathbb{C}_i\right) \ > \ \mathbb{C}\right) = Prob\left(\bigcup_{x\in\mathbb{F}_2{}^n/\mathbb{C}} \cap_{i=1}^{m}\mathbb{C}_i \text{ contains } x\right)$$

$$\leq \sum_{x\in\mathbb{F}_2{}^n/\mathbb{C}} Prob\left(\cap_{i=1}^{m}\mathbb{C}_i \text{ contains } x\right)$$

$$= (2^{n-k}-1)\left(\frac{2^{n-k-\epsilon}-1}{2^{n-k}-1}\right)^m \approx 2^{n-k-\epsilon m}$$

using the Gaussian binomial coefficients

$$V(n,l) \stackrel{\text{def}}{=} \frac{(2^n-1)(2^n-2)\cdots(2^n-2^{l-1})}{(2^l-1)(2^l-2)\cdots(2^l-2^{l-1})}$$

Complexity of Decoding

- $Prob\left(\dim\left(\cap_{i=1}^{m}\mathbb{C}_i\right) > k\right) \approx 2^{n-k-\epsilon m}$
- Each decoding iteration will have complexity $O(n^2)$
- Graph will have many short cycles and large degree
- But, will be a good expander

Related to (Dumer, 1991-96; Barg, Krouk and van Tilborg, 1999)

Ongoing work with Dan Rockmore

Summary

- Ordered binary decision diagrams $\approx$ minimal trellises
- Minimum distance restriction is of little significance
- Coding theory has emphasized linear codes
- Decoding problems seem to restrict useful representations
- The two communities have much to learn from each other