Improving Belief Propagation via Graphical Model Transformation

Thomas R. Halford
Communication Sciences Institute
University of Southern California

May 1, 2007

Algorithms, Inference, and Statistical Physics

Work supported in part by:
- US Army Research Office, MURI Contract DAAD19-01-1-0477
- TrellisWare Technologies, Inc.
- Powell Foundation

Improving Belief Propagation

Generalized Belief Propagation

↳ *regions of nodes pass messages*

↳ *regions may overlap*

Loop Calculus

↳ *view BP as a truncation of a series expansion for exact inference*

↳ *improve performance by including more terms*

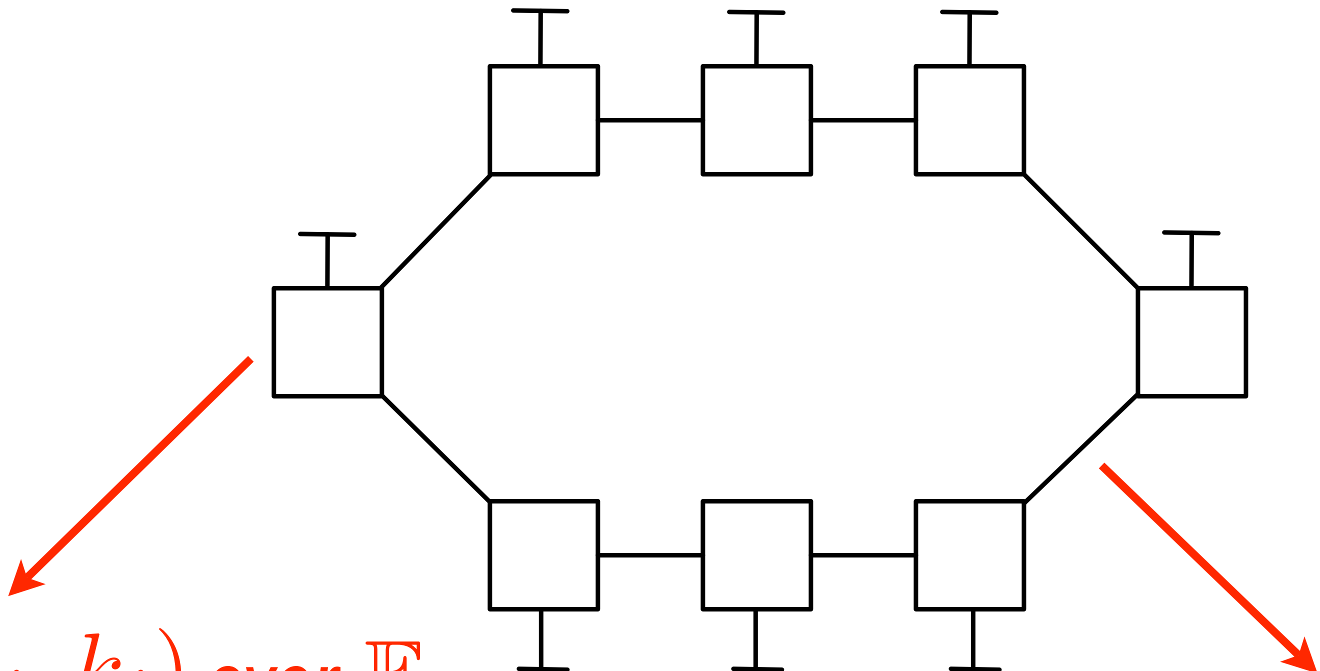
standard model, non-standard BP

Graphical Model Transformation

non-standard model, standard BP

Graphical Models for Codes: Normal Realizations

$\mathcal{C}:(n,k)$ over $\mathbb{F}_q$



$\mathcal{C}_i:(n_i,k_i)$ over $\mathbb{F}_q$

$\mathbb{F}_q^m$

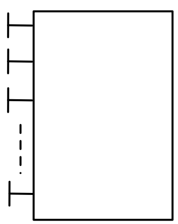
Graphical Model Extraction - One Code Many Models

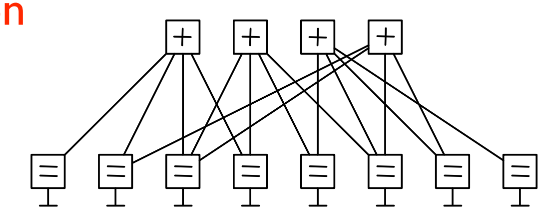
Code Definition

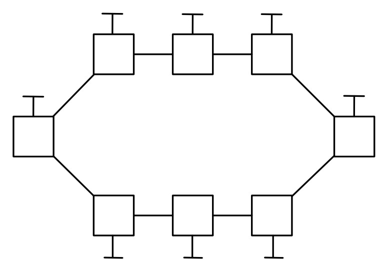
Graphical Models

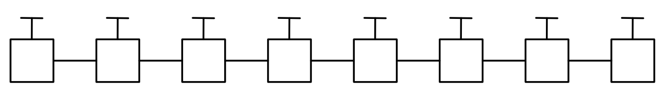
Decoding Algorithms

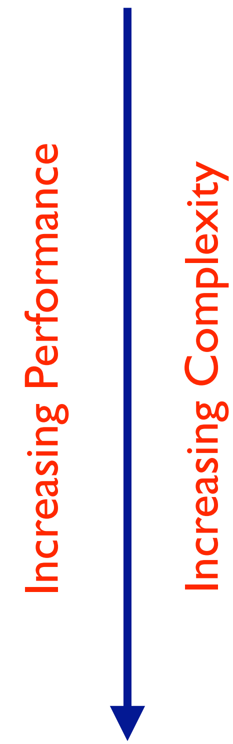
Extraction











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

Towards a Formalization of Extraction

Towards a Formalization of Extraction

Question 1: *Can the space of graphical models for a code be searched?*

Towards a Formalization of Extraction

Question 1: *Can the space of graphical models for a code be searched?*

YES - We'll show how

Towards a Formalization of Extraction

Question 1: *Can the space of graphical models for a code be searched?*

YES - We'll show how

Question 2: *What are meaningful optimization constraint functions?*

Towards a Formalization of Extraction

Question 1: *Can the space of graphical models for a code be searched?*

YES - We'll show how

Question 2: *What are meaningful optimization constraint functions?*

Inclusion in some model complexity class

Towards a Formalization of Extraction

Question 1: *Can the space of graphical models for a code be searched?*

YES - We'll show how

Question 2: *What are meaningful optimization constraint functions?*

Inclusion in some model complexity class

Question 3: *What are meaningful optimization cost functions?*

Towards a Formalization of Extraction

Question 1: *Can the space of graphical models for a code be searched?*

YES - We'll show how

Question 2: *What are meaningful optimization constraint functions?*

Inclusion in some model complexity class

Question 3: *What are meaningful optimization cost functions?*

Cost function ⇔ *good* graphical model

Towards a Formalization of Extraction

Question 1: *Can the space of graphical models for a code be searched?*

YES - We'll show how

Question 2: *What are meaningful optimization constraint functions?*

Inclusion in some model complexity class

Question 3: *What are meaningful optimization cost functions?*

Cost function ⇔ *good* graphical model

Difficult and open problem in general...
Our Approach - *Short Cycle Structure*

Towards a Formalization of Extraction

Question 1: *Can the space of graphical models for a code be searched?*

YES - We'll show how

Question 2: *What are meaningful optimization constraint functions?*

Inclusion in some model complexity class

Question 3: *What are meaningful optimization cost functions?*

Cost function $\Leftrightarrow$ *good* graphical model

Difficult and open problem in general...
Our Approach - *Short Cycle Structure*

Question 4: *What are good heuristics for this hard combinatorial optimization?*

Towards a Formalization of Extraction

Question 1: *Can the space of graphical models for a code be searched?*

YES - We'll show how

Question 2: *What are meaningful optimization constraint functions?*

Inclusion in some model complexity class

Question 3: *What are meaningful optimization cost functions?*

Cost function ⇔ *good* graphical model

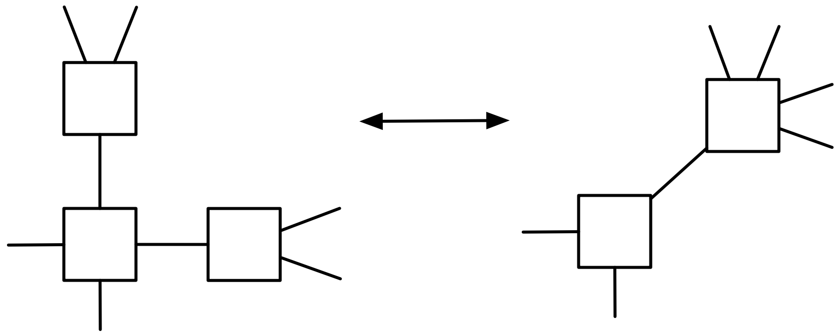
Difficult and open problem in general...
Our Approach - *Short Cycle Structure*

Question 4: *What are good heuristics for this hard combinatorial optimization?*

Difficult and open problem in general...
Our Approach - *Greedy Extraction Heuristic*

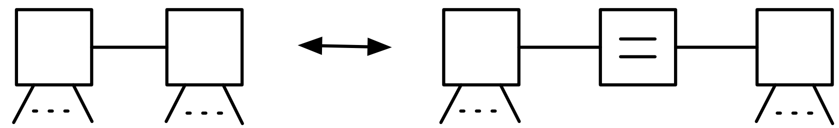
Searching the Model Space: Basic Operations

Constraint Merging / Splitting

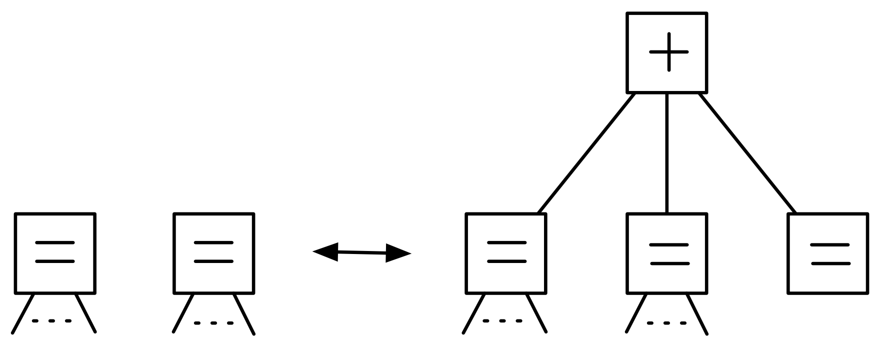


[Pe88], [Fo01], [KsFrLo01]

Inserting / Removing Degree-2 Repetition Constraint



Inserting / Removing Isolated Partial Parity Constraints



Generalized Parity-Check Matrices

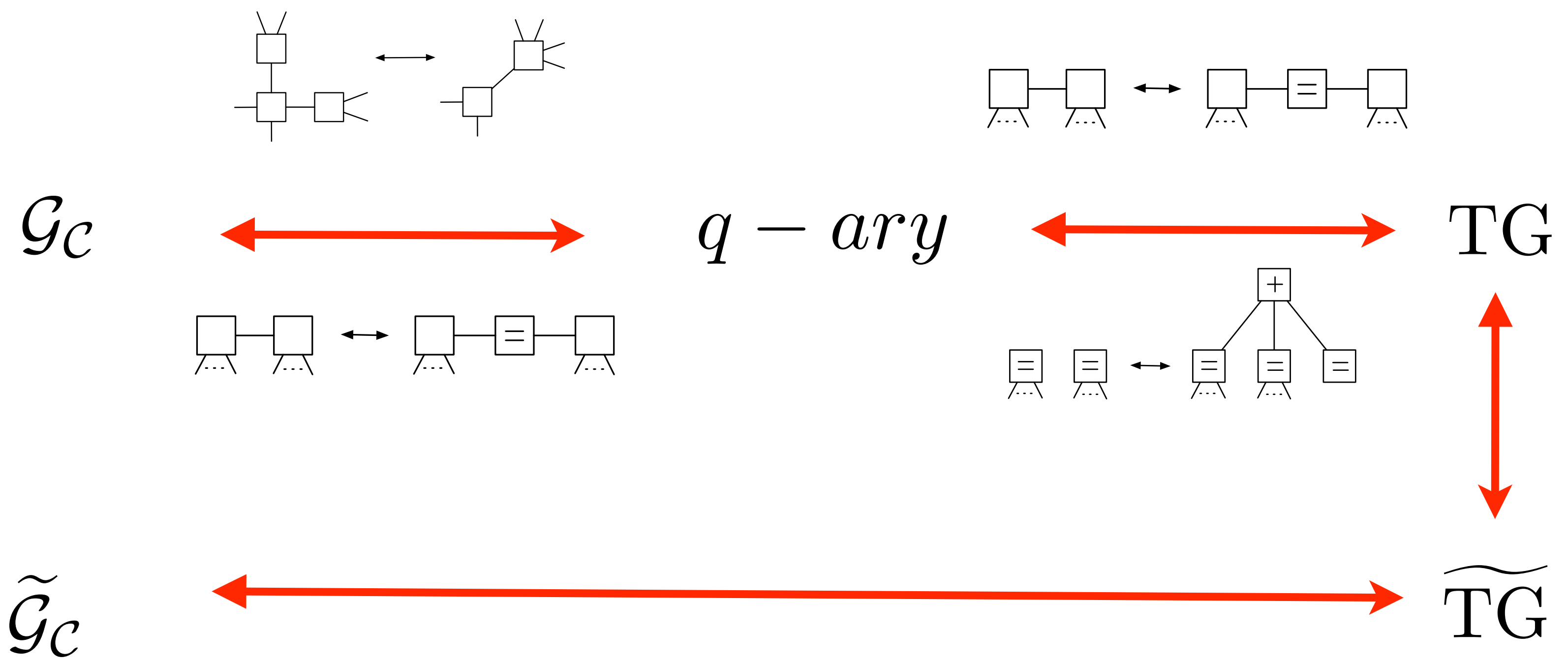
Inserting / Removing Trivial Constraints

Redundant Parity-Check Matrices

Searching the Model Space: Main Result

Theorem: Let $\mathcal{G}_\mathcal{C}$ and $\widetilde{\mathcal{G}}_\mathcal{C}$ be two graphical models for $\mathcal{C}$. Then $\mathcal{G}_\mathcal{C}$ can be transformed into $\widetilde{\mathcal{G}}_\mathcal{C}$ via a *finite* number of basic operations.

Proof:



Constraint Functions - q^m-ary Graphical Models

1) Maximum hidden variable alphabet size: q^m.

2) Each local constraint $\mathcal{C}_i$ satisfies:

$$\min\left(k_i, n_i - k_i\right) \leq m$$

Wolf's bound on local trellis complexity

(or is a direct product of codes which do).

Cost Functions - Short Cycle Structure

Candidate Proxies:

- stopping sets
- trapping / absorbing sets
- pseudo-codewords
- short cycles

Cost Functions - Short Cycle Structure

Candidate Proxies:

- stopping sets ✘
- trapping / absorbing sets
- pseudo-codewords
- short cycles

Cost Functions - Short Cycle Structure

Candidate Proxies:

- stopping sets ✗
- trapping / absorbing sets ✗
- pseudo-codewords
- short cycles

Cost Functions - Short Cycle Structure

Candidate Proxies:

- stopping sets ✗
- trapping / absorbing sets ✗
- pseudo-codewords ✗
- short cycles

Cost Functions - Short Cycle Structure

Candidate Proxies:

- stopping sets ✘
- trapping / absorbing sets ✘
- pseudo-codewords ✘
- short cycles

➘ Can count cycles of length g, $g+2$ and $g+4$ in time $O(gn^3)$.

(Halford & Chugg, "An algorithm for counting short cycles in bipartite graphs", *IEEE Trans. IT,* 52(1) 2006.)

A Greedy Heuristic for Model Extraction

Motivation: Tanner graphs for many block codes *necessarily* contain *many* short cycles

Halford, Grant & Chugg, "Which codes contain 4-cycle-free Tanner graphs?", *IEEE Trans. IT,* 52(9) 2006.

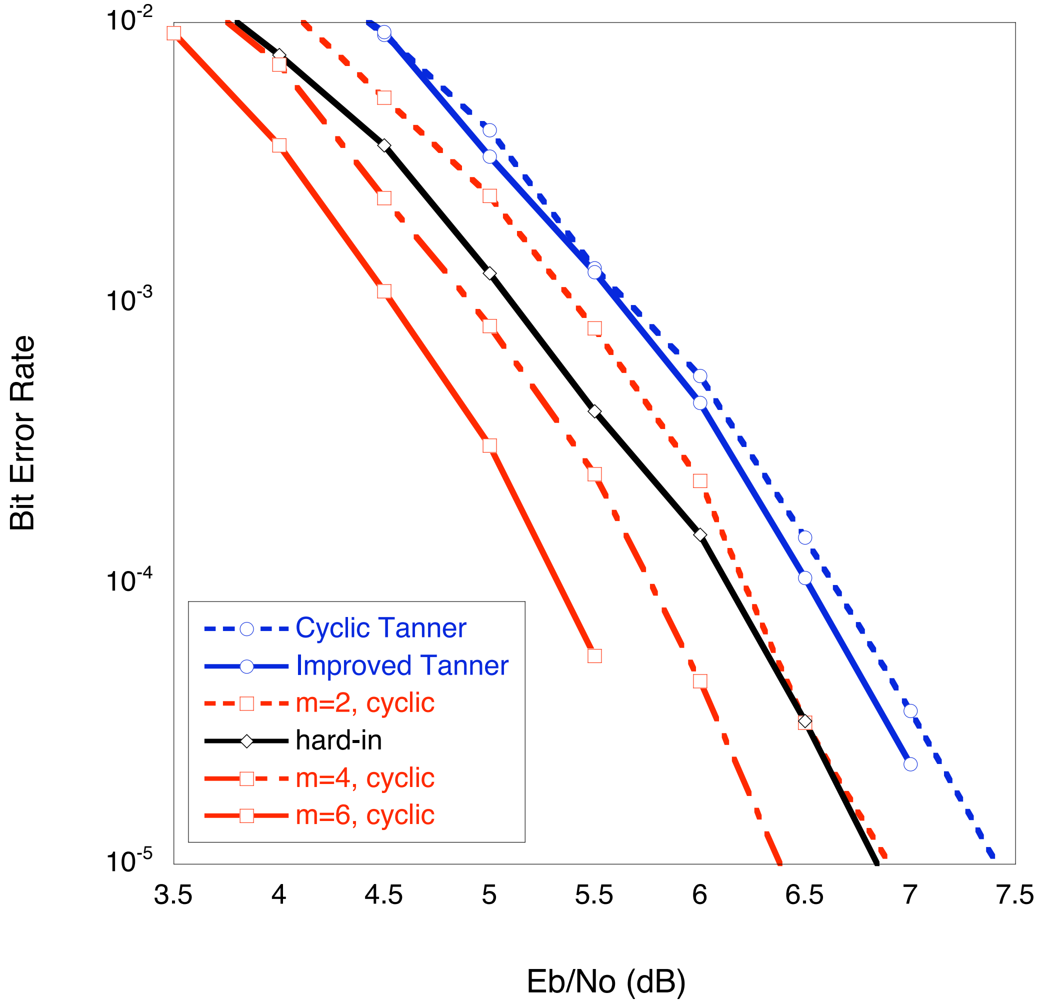
Idea: Greedily reduce cycles via model transformation

Allowed Moves: 1) Tanner graph search - *row operations*

2) 2^m-ary search - *local constraint merging*

Cost Function: Short cycle structure (N_4, N_6, N_8)

Greedy Heuristic: Experimental Results



Model	N_4	N_6	N_8
Cyclic Tanner	7251	717 K	74 M
Improved Tanner	5415	466 K	43 M
$m = 2$, cyclic	3465	230 K	15 M
$m = 4$, cyclic	706	16 K	292 K
$m = 6$, cyclic	126	657	0

(63,45) BCH Code

[HaCh06b], [JiNa06]

Synthesis & Open Problems

GBP vs. Model Tx:

- *similar if GBP regions don't overlap*
- *model transformation allows redundancy*

Loop Calculus vs. Model Tx:

- *similar problem of how to transform / what loop terms to use*
- *model transformation improves dense models, loop calculus improves sparse models*

Major Open Problems:

- *better cost functions & search heuristics*
- *model transformation + GBP / loop calculus*