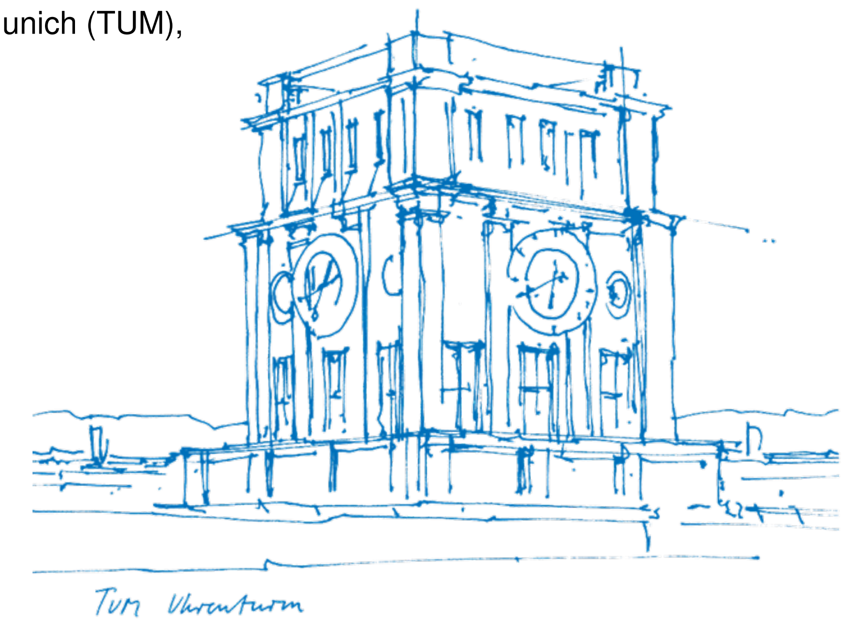
Coding and Bounds for Partially Defect Memory Cells

Haider Al Kim, Sven Puchinger, Antonia Wachter-Zeh
Institute for Communications Engineering, Technical University of Munich (TUM), Germany

NVWM[21] March 9, 2021



(Partially) Stuck[f] Memory Cells (SMC and PSMC)

SMC: Introduced by **[1], then further studied in [2] considering errors as well, and in [3-4].**

PSMC: Introduced only by **[5].**

Related Works

- [1] A. Kuznetsov and B. Tsybakov, “Coding for memories with defective cells,” (in Russian) Problems Inf. Transmiss., vol. 10, no. 2, pp. 52–60, 1974.
- [2] C. Heegard, “Partitioned Linear Block Codes for Computer Memory with’Stuck-at’Defects,” IEEE Transactions on Information Theory, vol. 29, no. 6, pp. 831–842, 1983.
- [3] I. I. Dumer, “Asymptotically optimal codes correcting memory defects of fixed multiplicity,” Problemy Peredachi Informatsii, vol. 25, no. 4, pp. 3–10, 1989.
- [4] I. I. Dumer, “Asymptotically optimal linear codes correcting defects of linearly increasing multiplicity,” Problemy Peredachi Informatsii, vol. 26, no. 2, pp. 3–17, 1990.
- [5] A. Wachter-Zeh and E. Yaakobi, “Codes for Partially Stuck-at Memory Cells,” IEEE Transactions on Information Theory, vol. 62, no. 2, pp. 639–654, 2016.

[f] so called also (*Defective*)

Full Version Papers

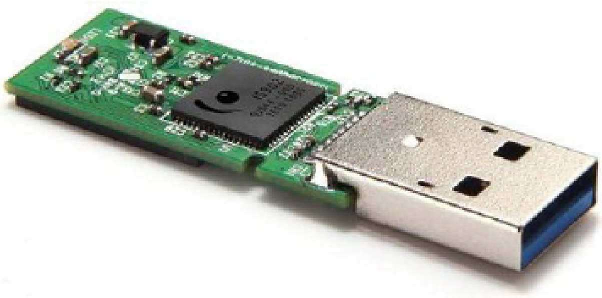
This work partially summarizes our constructions and bounds in [6] and [7], and presents a generalization of the GV bound given in [7].

- [6] H. Al Kim, S. Puchinger, A. Wachter-Zeh “Error Correction for Partially Stuck Memory Cells” Problemy Peredachi Informatsii, XVI International Symposium Problems of Redundancy in Information and Control Systems (Redundancy 2019), 978-1-7281-1944-1 IEEE, pp. 87–92, 2019.
- [7] H. Al Kim, S. Puchinger, A. Wachter-Zeh “Bounds and Code Constructions for Partially Defect Memory Cells” Seventeenth International Workshop on Algebraic and Combinatorial Coding Theory (October 11-17, ACCT 2020), will be published soon in the upcoming proceedings IEEE 2021.

Motivation - Codes for Non-Volatile Memories

A non-volatile memory is a memory that stores the information even when powered off.

Multilevel Flash Memories

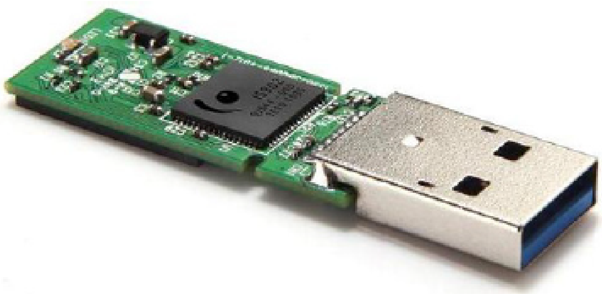


- Electronic charge represents multiple levels

Motivation - Codes for Non-Volatile Memories

A non-volatile memory is a memory that stores the information even when powered off.

Multilevel Flash Memories

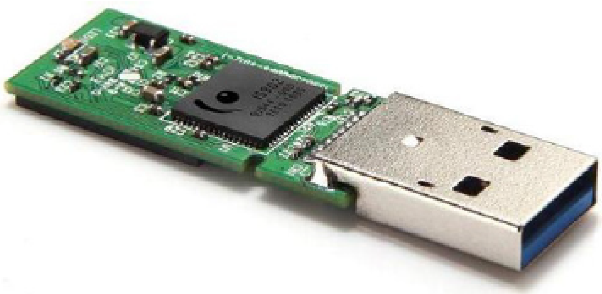


- Electronic charge represents multiple levels
- If charge is trapped, level can only be increased

Motivation - Codes for Non-Volatile Memories

A non-volatile memory is a memory that stores the information even when powered off.

Multilevel Flash Memories



- Electronic charge represents multiple levels
- If charge is trapped, level can only be increased
- Cells can be stuck: cannot change their value

Motivation - Codes for Non-Volatile Memories

A non-volatile memory is a memory that stores the information even when powered off.

Multilevel Flash Memories

- Electronic charge represents multiple levels
- If charge is trapped, level can only be increased
- Cells can be stuck: cannot change their value
- Erasing blocks $\Rightarrow$ (slow and decreases lifespan)
  $\Longrightarrow$ avoid by only increasing levels in a new write

Stuck Vs Partially Stuck Memory Cells

Stuck Cells (Classical Defects)

- Binary cells: cell can be stuck at level 0 or 1
- q-ary cells: cell can be stuck at any level $s \in \{0, \ldots, q-1\}$
- A stuck cell cannot change its level!
- We "mask" our information by assuring the exact s level to match the stuck positions
- Output: vector **c** with $c_{\phi_i} = s_i$, where ϕ_i are the stuck positions $\forall i \in u$ cells.

Stuck Vs Partially Stuck Memory Cells

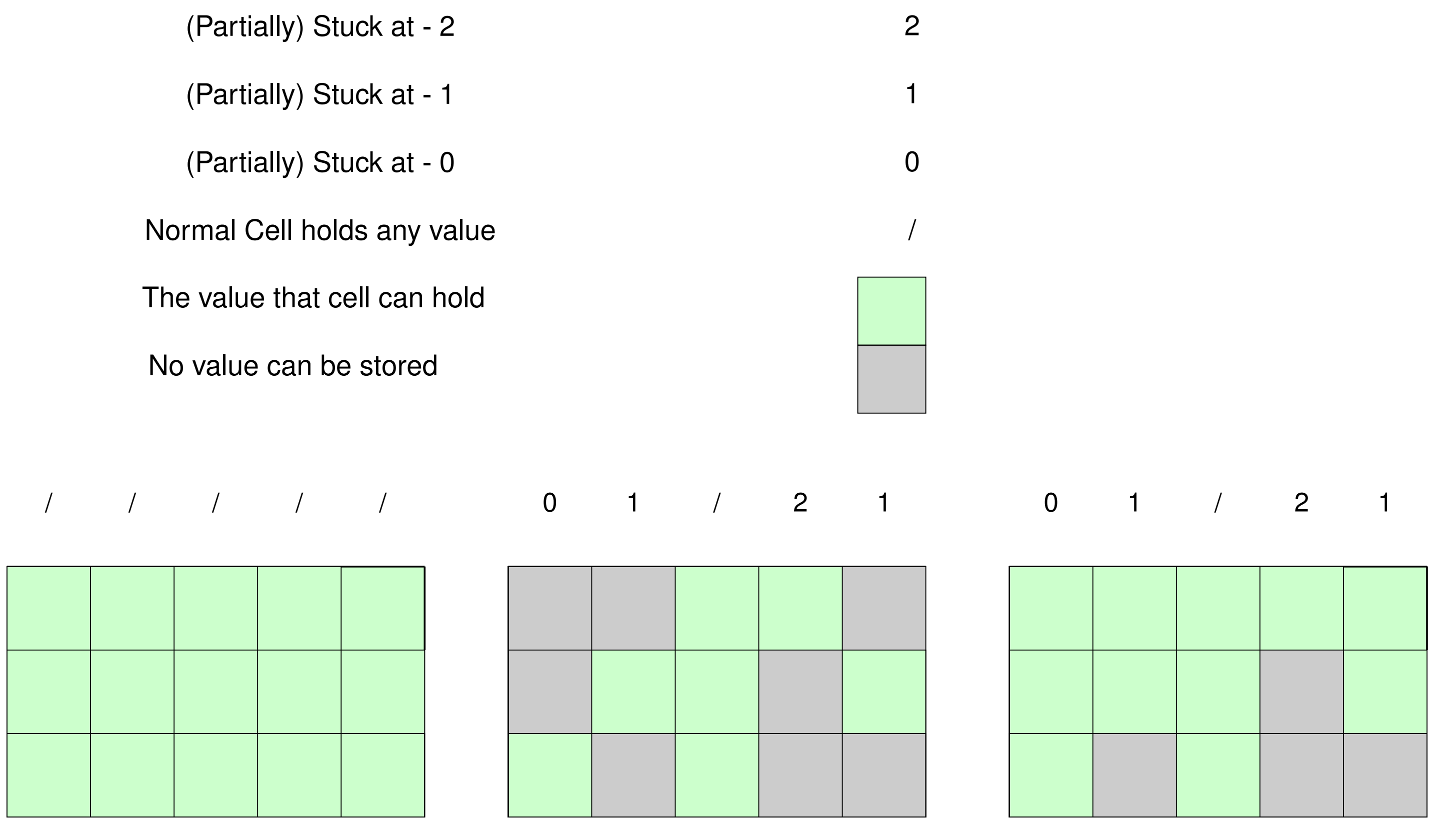
Stuck Cells (Classical Defects)

- Binary cells: cell can be stuck at level 0 or 1
- q-ary cells: cell can be stuck at any level $s \in \{0, \ldots, q-1\}$
- A stuck cell cannot change its level!
- We "mask" our information by assuring the exact s level to match the stuck positions
- Output: vector **c** with $c_{\phi_i} = s_i$, where ϕ_i are the stuck positions $\forall i \in u$ cells.

Partially Stuck Memory Cells

- q-ary cells: cell can be partially stuck at any level $s \in \{0, \ldots, q-1\}$
- A partially stuck cell can only store levels at least s
- If $s = 0$: anything can stored, uncorrupted cell
- Output: vector **c** with $c_{\phi_i} \geq s_i$ at stuck positions, where ϕ_i are the partially stuck positions $\forall i \in u$ cells.

Stuck Vs Partially Stuck Memory Cells



(A) Normal Cells (B) Stuck-at (C) Partially stuck-at

Our Contributions

1. **Code construction**:

.

Our Contributions

1. **Code construction**:
   - Presenting code construction for masking partially stuck cells while additionally correcting errors [6].

Our Contribution

2. **Bounds on the cardinality and the minimum distance**:

Our Contribution

2. **Bounds on the cardinality and the minimum distance**:

- Deriving a **sphere-packing-like bound** (any number of u partially stuck cells),

Our Contribution

2. **Bounds on the cardinality and the minimum distance**:

- Deriving a **sphere-packing-like bound** (any number of u partially stuck cells),
- and generalizing the derived **Gilbert-Varshamov-like bound** ($u < q$ partially stuck cells in [7]) to any number of u partially stuck level

for codes that can mask a certain number of partially stuck cells and correct errors additionally.

Our Contribution

2. **Bounds on the cardinality and the minimum distance**:

- Deriving a **sphere-packing-like bound** (any number of u partially stuck cells),
- and generalizing the derived **Gilbert-Varshamov-like bound** ($u < q$ partially stuck cells in [7]) to any number of u partially stuck level

for codes that can mask a certain number of partially stuck cells and correct errors additionally.

Numerical Analysis :

Our Contribution

2. **Bounds on the cardinality and the minimum distance**:

- Deriving a **sphere-packing-like bound** (any number of u partially stuck cells),
- and generalizing the derived **Gilbert-Varshamov-like bound** ($u < q$ partially stuck cells in [7]) to any number of u partially stuck level

for codes that can mask a certain number of partially stuck cells and correct errors additionally.

Numerical Analysis :

1. We compare how close our constructions for any u are to sphere-packing bound.

Our Contribution

2. **Bounds on the cardinality and the minimum distance**:

- Deriving a **sphere-packing-like bound** (any number of u partially stuck cells),
- and generalizing the derived **Gilbert-Varshamov-like bound** ($u < q$ partially stuck cells in [7]) to any number of u partially stuck level

for codes that can mask a certain number of partially stuck cells and correct errors additionally.

Numerical Analysis :

1. We compare how close our constructions for any u are to sphere-packing bound.
2. We check if our constructions for any u partially stuck cells $\leq n$ satisfy Gilbert-Varshamov bound.

Code Construction over $\mathbb{F}_q$

Code Construction over $\mathbb{F}_q$

Construction 1

Let n, u, t, k, k_1, r, l be positive integers with $u, t \leq n$, $k_1 = n - l - r$ and $k = k_1 + l$. Suppose that there are matrices $\boldsymbol{P} \in \mathbb{F}_q^{k_1 \times r}$ and $\boldsymbol{H}_0 \in \mathbb{F}_q^{(n-k_1-r)\times n}$ is a systematic parity-check matrix of a code $\mathcal{C}_0$ with parameters $[n, k_1 + r, d_0 \geq u - q + 3]_q$, such that

$$\boldsymbol{G} = \begin{bmatrix} \boldsymbol{G}_1 \\ \boldsymbol{H}_0 \end{bmatrix} = \begin{bmatrix} \boldsymbol{0}_{k_1 \times l} & \boldsymbol{I}_{k_1} & \boldsymbol{P}_{k_1 \times r} \\ & \boldsymbol{H}_0 & \end{bmatrix}$$

is a generator matrix of a $[n, k = l + k_1, d \geq 2t + 1]_q$ code $\mathcal{C}$.
Based on these definitions, we define a coding scheme in Algorithm 1 and Algorithm 2.

Theorem 1

The coding scheme in Construction 1 is a (u, t)-PSMC with $s_{\phi_i} = 1$ of length n and cardinality $\mathcal{M}_{u,t} = q^{k_1}$.

Encoding and Decoding - Theorem 1

Algorithm 1: Encoding ($\boldsymbol{m}; \phi$)

Input:

- Message: $\boldsymbol{m} = (m_0, m_1, \ldots, m_{k-1}) \in \mathbb{F}_q^{k_1}$ and,
- Positions of partially stuck-at-($s = 1$) cells: $\phi \subseteq \{0, \ldots, n\}$

1. Find $\boldsymbol{z} = (z_0, z_1, \cdots z_{l-1}) \in \mathbb{F}_q$ as explained in the proof of Theorem 7[a]
2. Compute $\boldsymbol{c} = ((\boldsymbol{m} \cdot \boldsymbol{G}_1) + (\boldsymbol{z} \cdot \boldsymbol{H}_0)) \mod q$

Output: Codeword $\boldsymbol{c} \in \mathbb{F}_q^n$

[a] A. Wachter-Zeh and E. Yaakobi, 2016

Proof of Theorem 1

- Encoding

1. Algorithm 1 finds $\boldsymbol{z}$ similar to Algorithm 7 in [a]
2. The proof follows a detailed proof of Theorem 8 in [a] for the masking part which replaces the all-ones vector with an $(n-k) \times n$ parity-check matrix $\boldsymbol{H}$.
3. Instead of $\boldsymbol{H}$, Theorem 1 uses $\boldsymbol{H}_0$ which is $(n-k_1-r) \times n$.
4. The error correction part is related to $[n, k_1]_q$ code $\mathcal{C}_1$ and matrix $\boldsymbol{G}_1$.
5. Hence, we need to choose r between $1 \leq r < n-1$ such that the supercode $\mathcal{C}$ can correct t errors.
6. r should be the smallest value such that an $[n, n-r, \delta_1]_q$ code exists.
7. So $\boldsymbol{P}$ is hard to find in general.
8. One way to construct $\boldsymbol{P}$ is to start with a generator matrix of a well-known code (e.g. certain cyclic codes) and has a large enough minimum distance.
9. Then, we apply elementary row operations and column permutations to have the desired form given in Construction 1.

[a] A. Wachter-Zeh and E. Yaakobi, 2016

Encoding and Decoding - Theorem 1

Algorithm 2: Decoding

Input: $\boldsymbol{y} = \boldsymbol{c} + \boldsymbol{e} \in \mathbb{F}_q^n$, where $\boldsymbol{c}$ is a valid output of Algorithm 1 and $\boldsymbol{e}$ is an error of Hamming weight at most t.

- $\hat{\boldsymbol{c}} \leftarrow$ decode $\boldsymbol{y}$ in the code $\mathcal{C}$
- Unmasking Process:
  1. $\hat{\boldsymbol{z}} \leftarrow (\hat{c}_0, \hat{c}_1, \cdots \hat{c}_{l-1})$
  2. $\hat{\boldsymbol{w}} = (\hat{w}_0, \hat{w}_1, \cdots, \hat{w}_{n-1}) \leftarrow (\hat{\boldsymbol{c}} - \hat{\boldsymbol{z}} \cdot \boldsymbol{H}_0) \mod q$
  3. $\hat{\boldsymbol{m}} \leftarrow (\hat{w}_l, \hat{w}_{l+1} \ldots, \hat{w}_{n-r-1})$

Output: Message vector $\hat{\boldsymbol{m}} \in \mathbb{F}_q^{k_1}$

Proof of Theorem 1

Decoding ($\boldsymbol{y}$)

1. **Error correction**: the decoder (Algorithm 2) gets $\boldsymbol{y}$, which is $\boldsymbol{c}$ corrupted by at most t errors. Since $\mathcal{C}$ has minimum distance $d \geq 2t + 1$, we can correct these errors and obtain $\boldsymbol{c}$.
2. **Unmasking**:
   - Due to the structure of the matrix $\boldsymbol{G}$, the left most significant positions of $\boldsymbol{c}$ equals the vector $\boldsymbol{z}$.
   - Hence, we can compute $\hat{\boldsymbol{w}} = \boldsymbol{m} \cdot \boldsymbol{G}_1$ (cf. Algorithm 2) and then the message vector $\hat{\mathbf{m}} = \mathbf{m}$.

Sphere-Packing Bound on PSMCs

(Necessary Condition)

Sphere-Packing Bound on PSMCs

Theorem 2

Any (u,t)-PSMC over $\mathbb{F}_q$ of cardinality $\mathcal{M}_{u,t}$ satisfies:

- for non-overlapping, i.e., the errors are only allowed to happen at positions $\Psi = \{\Psi_0, \Psi_1, \ldots, \Psi_{n-u-1}\} = [n] \setminus \phi$, then we have:

$$\mathcal{M}_{u,t} \cdot \sum_{j=0}^{t} \binom{n-u}{j} (q-1)^j \leq q^{n-u} \prod_{i=0}^{u-1} (q - s_{\phi_i}), \quad (1)$$

- for overlapping, i.e., the errors are allowed to happen in any cell, i.e., $\Psi \subseteq [n]$ under the assumption that $s_{\phi_i} < q-1$, for $i = 0, \ldots, u-1$, then we have:

$$\mathcal{M}_{u,t} \cdot \sum_{j=0}^{t} \sum_{j_1=0}^{j} \binom{n-u}{j_1} (q-1)^{j_1} \binom{u}{j-j_1} \prod_{i \in \mathcal{J}} (q-1-s_{\phi_i}) \leq q^{n-u} \prod_{i=0}^{u-1} (q - s_{\phi_i}), \quad (2)$$

where $\mathcal{J}$ denotes the set of cardinality $j - j_1$ of stuck cells that is affected by errors.

Proof of Theorem 2

Non-overlapping

1. For any reliable memory cells, the cardinality: $\mathcal{M} = q^k \leq q^n$.
2. Theorem 2 in [a] (PSMC only) proofs that at most $\mathcal{M}_u$ can be stored:

$$\mathcal{M}_u \leq q^{n-u} \prod_{i=1}^{u} (q - s_{\phi_i}). \tag{3}$$

3. (3) gives the RHS of (1).
4. Around $n - u$ normal cells, there is a decoding sphere of radius $t = \lfloor \frac{d-1}{2} \rfloor$.
5. Follows SPB proof, there are $\mathcal{B}_{t,n-u} := \sum_{j=0}^{t} \binom{n-u}{j} (q-1)^j$ words in distance exactly j from a fixed word.
6. A sphere $\mathcal{B}_{t,n-u}(\boldsymbol{a})$ of radius at most t is: $\mathcal{B}_{t,n-u}(\boldsymbol{a}) := \{\boldsymbol{b} \in \mathbb{F}_q^{n-u} : d(\boldsymbol{a}, \boldsymbol{b}) \leq t\}$.
7. The total number of words (LHS of (1)) is at most the size of the whole space (RHS of (1)).

[a] A. Wachter-Zeh and E. Yaakobi, 2016

Proof of Theorem 2

Overlapping

1. Similar proof for the RHS of (1) applies for (2).
2. Errors can happen either in the $n-u$ non-stuck cells or in the u stuck cells such that $s_{\phi_i} + e_i \leq q-1$.
3. There are $\sum_{j_1=0}^{j} \binom{n-u}{j_1}\binom{u}{j-j_1}$ possibilities for j erroneous positions.
4. There are $\mathcal{B}_{t,u,n} := \sum_{j=0}^{t}\sum_{j_1=0}^{j}\binom{n-u}{j_1}(q-1)^{j_1}\binom{u}{j-j_1}\prod_{i\in\mathcal{J}}(q-1-s_{\phi_i})$.
5. $\mathcal{J}$ denotes the set of cardinality $j-j_1$ of stuck cells that is affected by errors.
6. $(\mathcal{M}_{u,t}\cdot\mathcal{B}_{t,u,n})$ is at most the total number of possibilities and the statement follows ($\Rightarrow$ the LHS of (2)).

Analysis and Comparison

(Sphere-Packing Bound on PSMCs)

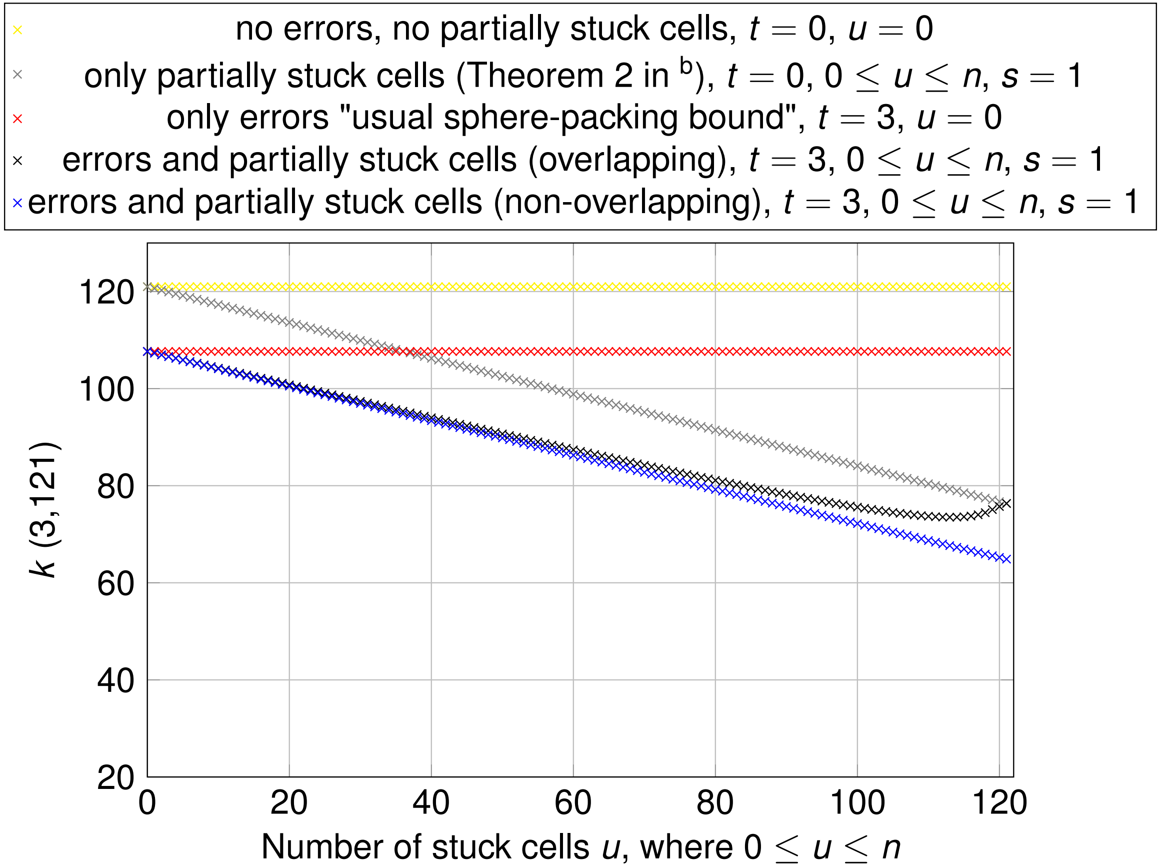


Figure: The chosen parameters are $q = 3$, and $n = ((q^5 - 1)/(q - 1))$.

[b] A. Wachter-Zeh and E. Yaakobi, 2016

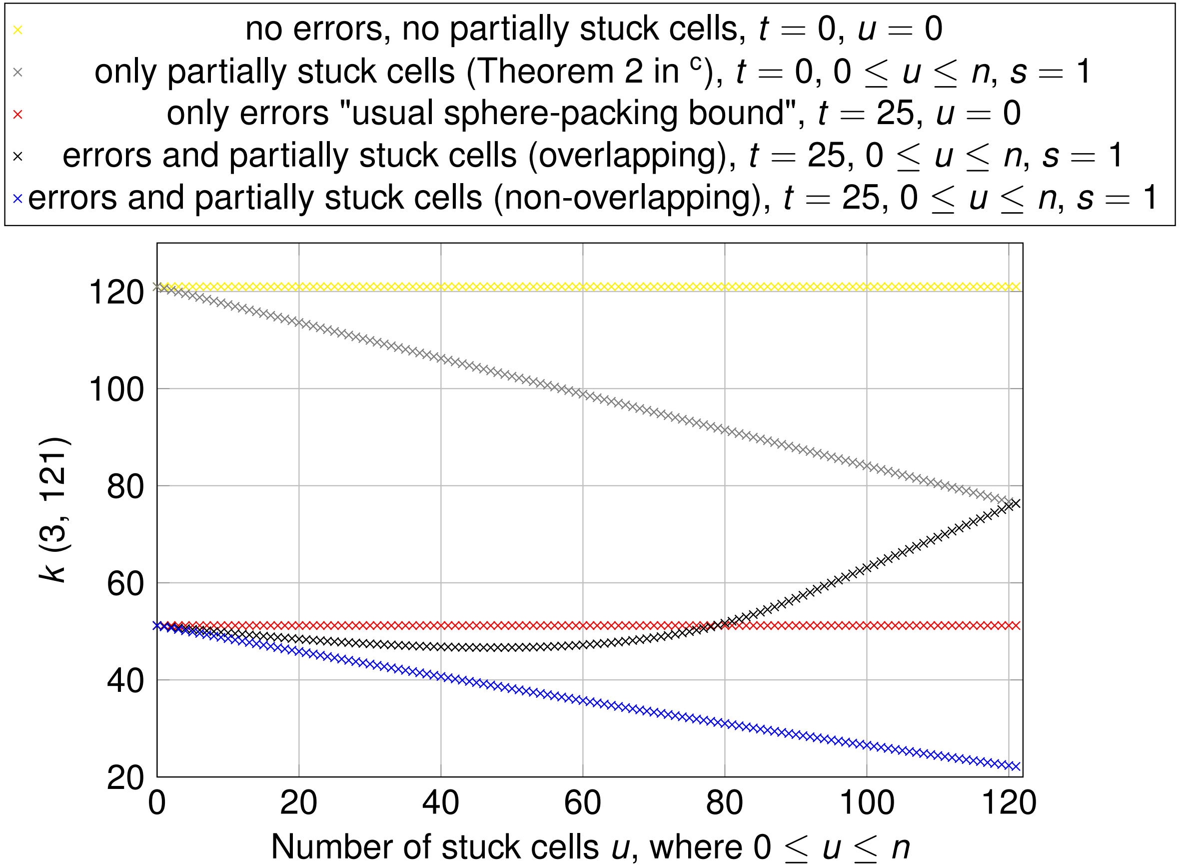


Figure: The chosen parameters are $q = 3$, and $n = ((q^5 - 1)/(q - 1))$.

[c] A. Wachter-Zeh and E. Yaakobi, 2016

Gilbert-Varshamov-like bound on PSMCs

(Sufficient Condition)

Gilbert–Varshamov Bound on PSMCs

Theorem 3

Suppose there are positive integers u, t, n, $d_0^{\perp} \geq d \geq 2t+1$, $d_0 \geq u-q+3$, $l \leq k-1$, and $q \geq 2$ that satisfy:

$$q^{l+1} \cdot \sum_{i=0}^{2t} \binom{n}{i} (q-1)^i \leq q^n,$$

$$q^{n-l+1} \cdot \sum_{j=0}^{u-q+2} \binom{n}{j} (q-1)^j \leq q^n,$$

$$q^k \cdot \sum_{i=0}^{2t} \binom{n}{i} (q-1)^i \leq q^n.$$

Then, there is an $[n, k, \geq 2t+1]_q$ code $\mathcal{C}$ that has a subcode $\mathcal{C}_0^{\perp}$ $[n, l]_q$ whose dual distance is $d_0 \geq u-q+3$.

Proof of GV Bound on PSMCs

Proof of Theorem 3

- First, we use a *probabilistic argument* for the GV bound as in [b] to prove that the code $\mathcal{C}_0^{\perp}$ of parameters $[n, l, d_0^{\perp}]_q$ generated by a matrix $\boldsymbol{H}_0 \in \mathbb{F}_q^{l \times n}$ whose dual code $\mathcal{C}_0$ of minimum distance $d_0 \geq u - q + 3$ simultaneously exists.

- Second, we need to show that we can extend $\mathcal{C}_0^{\perp}$ to obtain a code $\mathcal{C}$ of parameters $[n, k, d]_q$ generated by a full rank matrix of larger dimension $l + 1 \leq k$ so that we apply the method given in [c] with a suitable adaptation which is a restricted version for linear codes from the suggested method in [c] to extend $\mathcal{C}_0^{\perp}$.

[b] R. R. Varshamov, 1957
[c] E. N. Gilbert, 1952

Analysis and Comparison

(Gilbert-Varshamov-like bound on PSMCs)

Analysis and Comparison

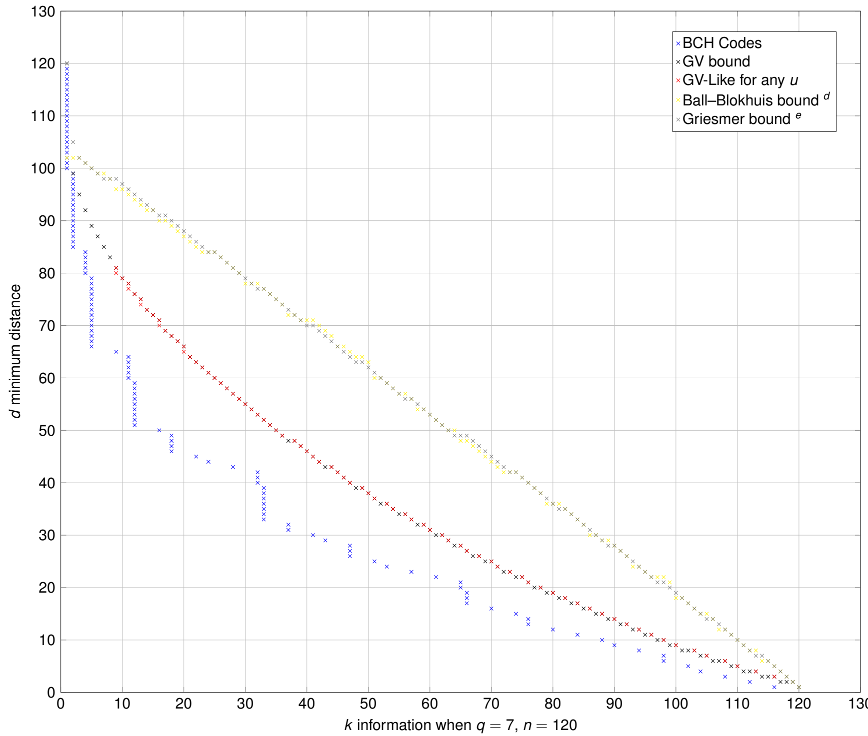


Figure: Comparisons of bounds of our new GV-Like-Bound for $n = 120$ and $q = 7$. It shows there exist error correcting and masking codes that satisfy GV bound starting after a certain dimension ($k = 9$) and particular minimum distance ($d = 81$). For this plot, $9 \leq \lceil \zeta(n, d_0^{\perp}, q) \rceil + 1 \leq 116$ while $0 \leq \lfloor \xi(n, h_q(\delta_0^{\perp}), \varepsilon) \rfloor \leq 110$.

[d] S. Ball and A. Blokhuis, 2013

[e] J. H. Griesmer, 1960

Summary

- Proposing a code construction for combined masking and error-correcting code for PSMC.

- Deriving bounds on the cardinality and the minimum distance.
  - Necessary Condition: a sphere-packing bound (for any u PSMC).
  - Sufficient Condition: a Gilbert–Varshamov-like bound (for any u PSMC).

- Numerical Analysis and comparison.
  - For sphere-packing bound (for any u PSMC).
  - For Gilbert–Varshamov-like bound (for any u PSMC).

Thank You

Any questions ... ?