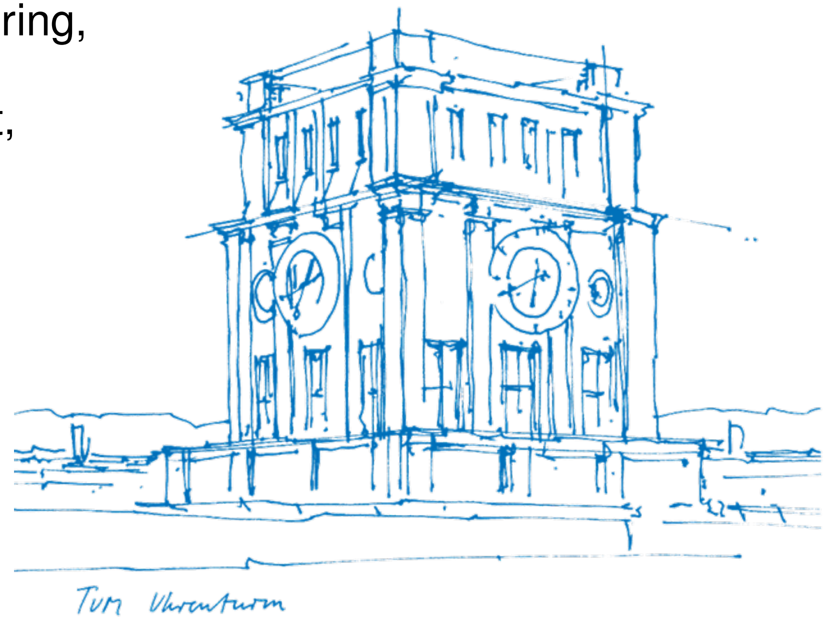
Codes for Cost-Efficient DNA Synthesis

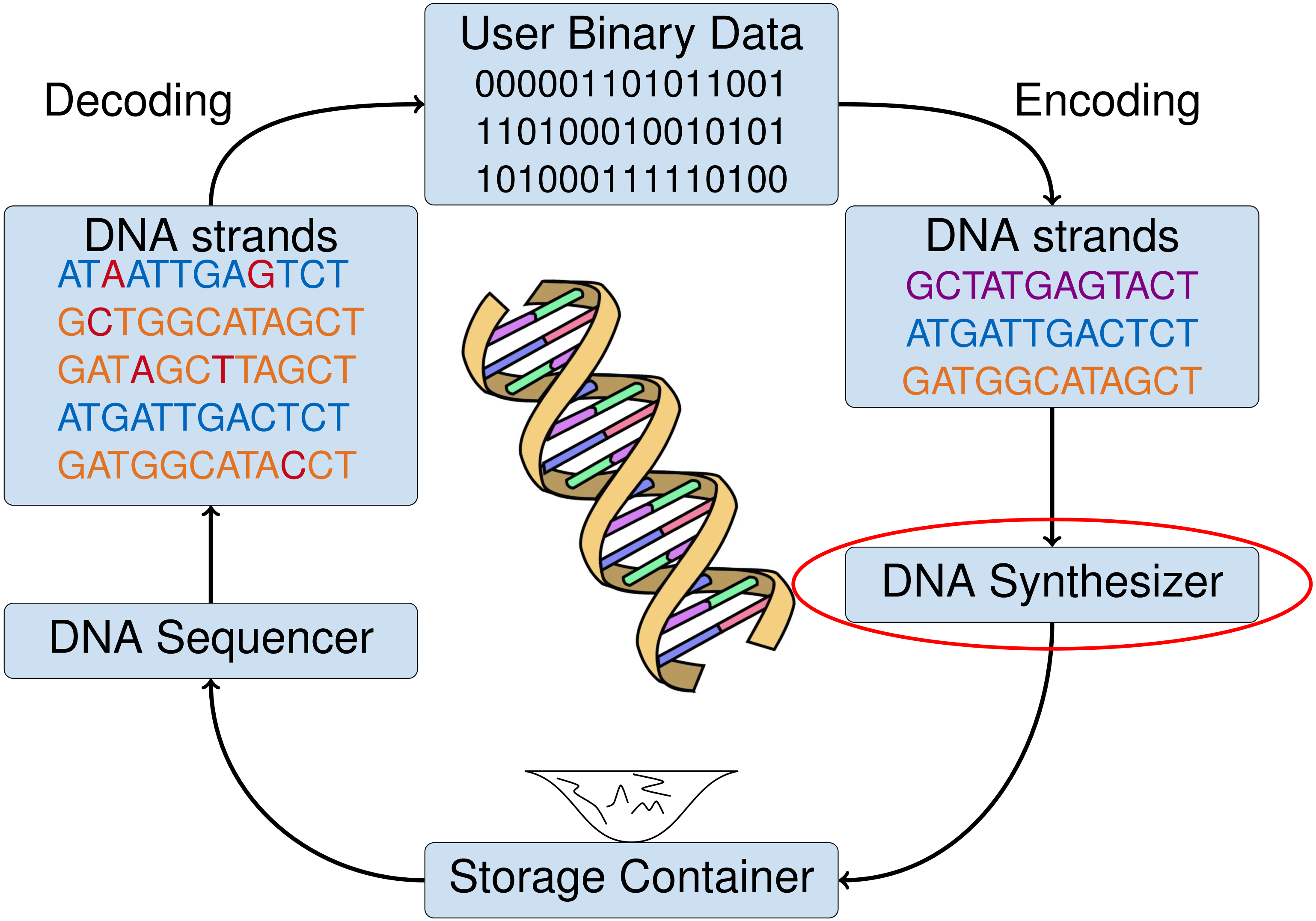
Andreas Lenz[1], **Yi Liu**[2], **Cyrus Rashtchian**[3],
Paul H. Siegel[2], **Andrew Tan**[2] **Antonia Wachter-Zeh**[1], **Eitan Yaakobi**[4]
[1]Institute for Communications Engineering,
Technische Universität München, Germany
[2]Department of Electrical and Computer Engineering,
University of California, San Diego, USA
[3]Computer Science and Engineering Department,
University of California, San Diego, USA
[4]Computer Science Department,
Israel Institute of Technology, Haifa, Israel



Non-Volatile Memory Workshop
7-10 March 2021

Outline

Data Storage in DNA



DNA Synthesis

Array-based synthesis

- Nucleotide-by-nucleotide parallel synthesis of DNA strands
- Flushing of nucleotides and selection of strands

Cycle	1	2	3	4	5	6	7	8	
$\mathbf{S}$	A	C	G	T	A	C	G	T	
$\mathbf{x}_1$		C		T	A	C	G		CTACG
$\mathbf{x}_2$	A		G	T	A				AGTA
$\mathbf{x}_3$		C		T				T	CTT

Figure: Synthesis of $\mathbf{x}_1 = (\text{CTACG})$, $\mathbf{x}_2 = (\text{AGTA})$, $\mathbf{x}_3 = (\text{CTT})$ using Synthesis sequence $\mathbf{S} = (\text{ACGTACGT})$.

DNA Synthesis

Array-based synthesis

- Parallel synthesis of $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_k \in \Sigma_q^*$ ($q = 4$ for DNA)
- Common synthesis sequence $\mathbf{S}$

Synthesis time

$$t(\mathbf{S}, \mathbf{x}_1, \ldots, \mathbf{x}_k) = \min_t \text{ s.t. } \mathbf{x}_i \text{ is subsequence of } \mathbf{S}_{1:t}$$

- E.g. $\mathbf{S} = (\text{ACGTACGTA})$, $\mathbf{x}_1 = (\text{CTACG})$, $\mathbf{x}_2 = (\text{AGTA})$, $\mathbf{x}_3 = (\text{CTT})$.
  $\Rightarrow t(\mathbf{S}, \mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3) = 8$.

Problem formulation

- Synthesis code $\mathcal{C} \subseteq \Sigma_q^*$
- Goal: maximize information synthesized in time T

$$N(\mathbf{S}, T) = \max |\mathcal{C}| \text{ s.t. } t(\mathbf{S}, \mathbf{x}) \leq T \; \forall \, \mathbf{x} \in \mathcal{C}$$

DNA Synthesis

$$N(\mathbf{S}, T) = \max |\mathcal{C}| \text{ s.t. } t(\mathbf{S}, \mathbf{x}) \leq T \; \forall \; \mathbf{x} \in \mathcal{C}$$

- Maximum information rate per cost time

$$R(\mathbf{S}) = \lim_{T \to \infty} \frac{\log N(\mathbf{S}, T)}{T}$$

- $R(\mathbf{S})$ measured in bits/synthesis cycle

Contribution

- Characterize $R(\mathbf{S})$ for **arbitrary** periodic sequences **S** using cost-constrained systems
- Highlight parallels to the subsequence spectrum

Subsequence Graph

- Vertices: Symbols of **S**
- Edges: each vertex has q outgoing edges, one to the next appearance of each $\sigma \in \Sigma_q$

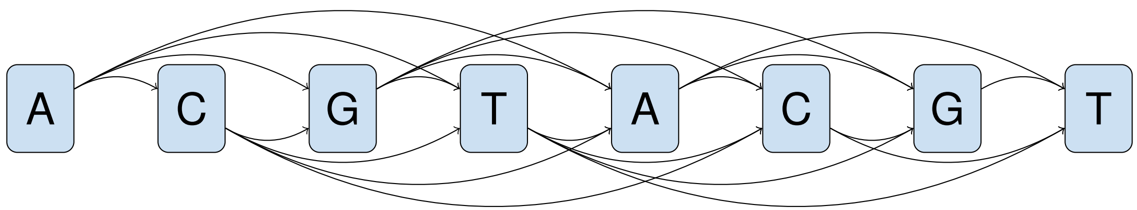


Figure: Subsequence graph $G(\mathbf{S})$ for Synthesis sequence $\mathbf{S} = (\text{ACGTACGT})$.

Observation

$\mathbf{x}$ can be synthesized using $\mathbf{S}$ iff it is obtained by some path through $G(\mathbf{S})$

Lemma (informal)

$N(\mathbf{S}, T) = \#\text{paths through } G(\mathbf{S}_{1:T})$

Subsequence Spectrum

- Subsequences of **S**: All words obtained by deleting symbols from **S**.
- E.g.

$$D((\mathrm{ACG})) = \{(\mathrm{ACG}), (\mathrm{AC}), (\mathrm{AG}), (\mathrm{CG}), (\mathrm{A}), (\mathrm{C}), (\mathrm{G}), ()\}$$

Lemma

$N(\mathbf{S}, T) = |D(\mathbf{S}_{1:T})|$

- Proof follows from left-alignment of subsequences in subsequence graph

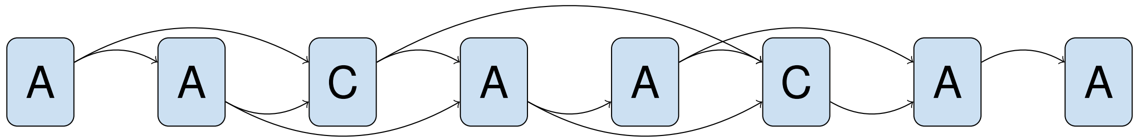
Equivalences

$$N(\mathbf{S}, T) \triangleq \text{largest synthesis code} \longleftrightarrow \#\text{paths through } G(\mathbf{S}) \longleftrightarrow \#\text{subsequences}$$

- Alternating sequence $\mathbf{A}_q = (0, 1, 2, \dots, q-1, 0, 1, 2, \dots)$
  $\Longrightarrow \arg\max_{\mathbf{S}} N(\mathbf{S}, T) = \mathbf{A}_q$

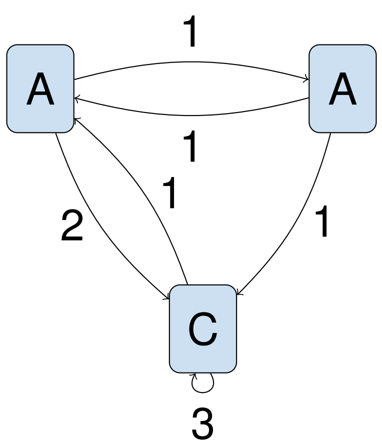
Periodic Synthesis Sequences

- Periodic sequence: $\mathbf{S} = (\mathbf{sss}\ldots)$, where $\mathbf{s} \in \Sigma_q^L$ is the period
- E.g.: Synthesis sequence $\mathbf{S} = (\text{AACAAC}\ldots)$



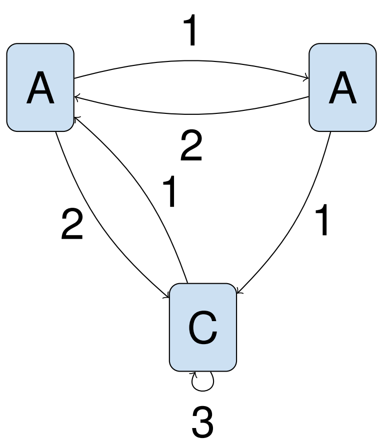
Periodic subsequence graph

- Vertices: Symbols of one period
- Edges: as in original graph, with step size as weight



Periodic Synthesis Sequences

- Periodic subsequence graph $\tilde{G}(\mathbf{S})$



Lemma (informal)

$N(\mathbf{S}, T) = \#$paths through $\tilde{G}(\mathbf{S})$ with sum weight at most T

- Edge weights: $\tau_{i,j}$ define costs from vertex i to j
- $\tilde{G}(\mathbf{S})$ is a cost-constrained system

Periodic Synthesis Sequences

- Define $C(\mathbf{S})$: Combinatorial capacity of cost-constrained system $\tilde{G}(\mathbf{S})$

Theorem

Maximal synthesis information rate per cost time

$$\lim_{T\to\infty} \frac{\log N(\mathbf{S}, T)}{T} =: R(\mathbf{S}) = C(\mathbf{S})$$

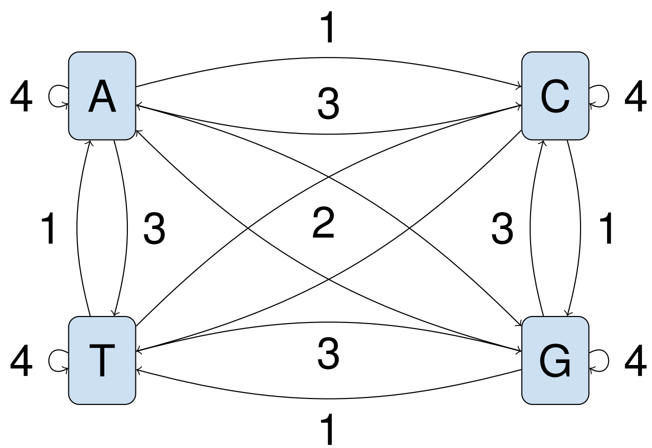
- Use results from constrained coding theory to find $R(\mathbf{S})$:

Combinatorial capacity of cost-constrained systems

- Given $\tilde{G}(\mathbf{S})$ with edge weights $\tau_{i,j}$
- Define $L \times L$ matrix $[P(z)]_{i,j} = \begin{cases} 0, & \text{if there is no edge from vertex } i \text{ to } j \\ 2^{-z\tau_{i,j}}, & \text{otherwise} \end{cases}$.
- z_0 : largest real solution to $\det(1 - P(z)) = 0$
  $\implies C(\mathbf{S}) = z_0$

Periodic Synthesis Sequences

- Example: Alternating sequence $\mathbf{S} = (\text{ACGTACGT}\dots)$



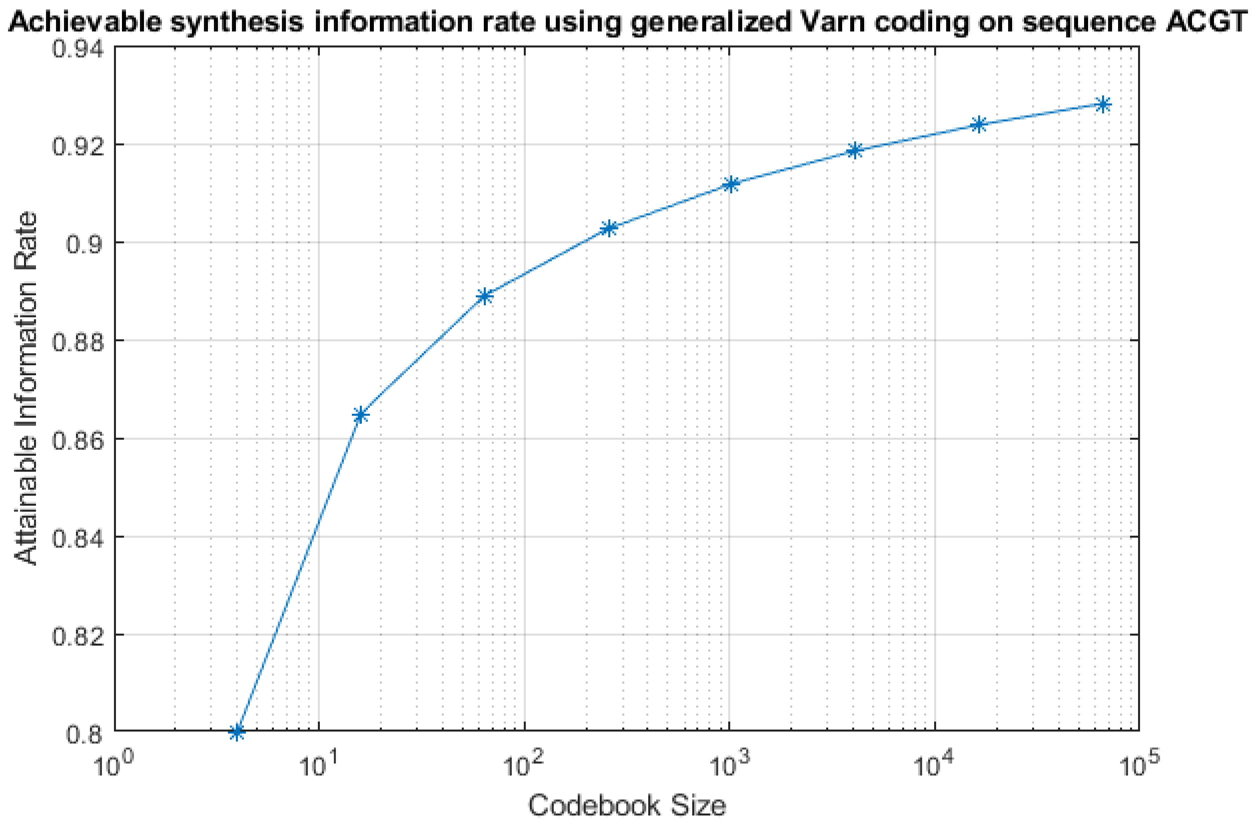
$$P(z) = \begin{pmatrix} 2^{-4z} & 2^{-z} & 2^{-2z} & 2^{-3z} \\ 2^{-3z} & 2^{-4z} & 2^{-z} & 2^{-2z} \\ 2^{-2z} & 2^{-3z} & 2^{-4z} & 2^{-z} \\ 2^{-z} & 2^{-2z} & 2^{-3z} & 2^{-4z} \end{pmatrix}$$

$\Longrightarrow R(\mathbf{S}) = z_0 \approx 0.947$ bit/cycle

- Uncoded: 0.5 bit/cycle
- Arbitrary q: $\sum_{i=1}^{q} 2^{-zi} = 1$

Explicit Codes: Varn Codes

- Varn codes: Explicit prefix-free codes for cost-constrained systems
- Construction via tree expansion



- With codebook size 4^8 we attain an information rate of $R \approx 0.928$ bit/cycle vs. capacity $R(\mathbf{S}) \approx 0.947$ bit/cycle

Fixed Length Sequences

- Restrict $\mathbf{x} \in \Sigma_q^n$ to have fixed length n.
- Largest synthesis code with fixed length: $N(\mathbf{S}, T, n)$
- Information rate per cycle:

$$R(\mathbf{S}, \alpha) = \lim_{T \to \infty} \frac{N(\mathbf{S}, T, \alpha T)}{T}$$

- $0 \leq \alpha \leq 1$

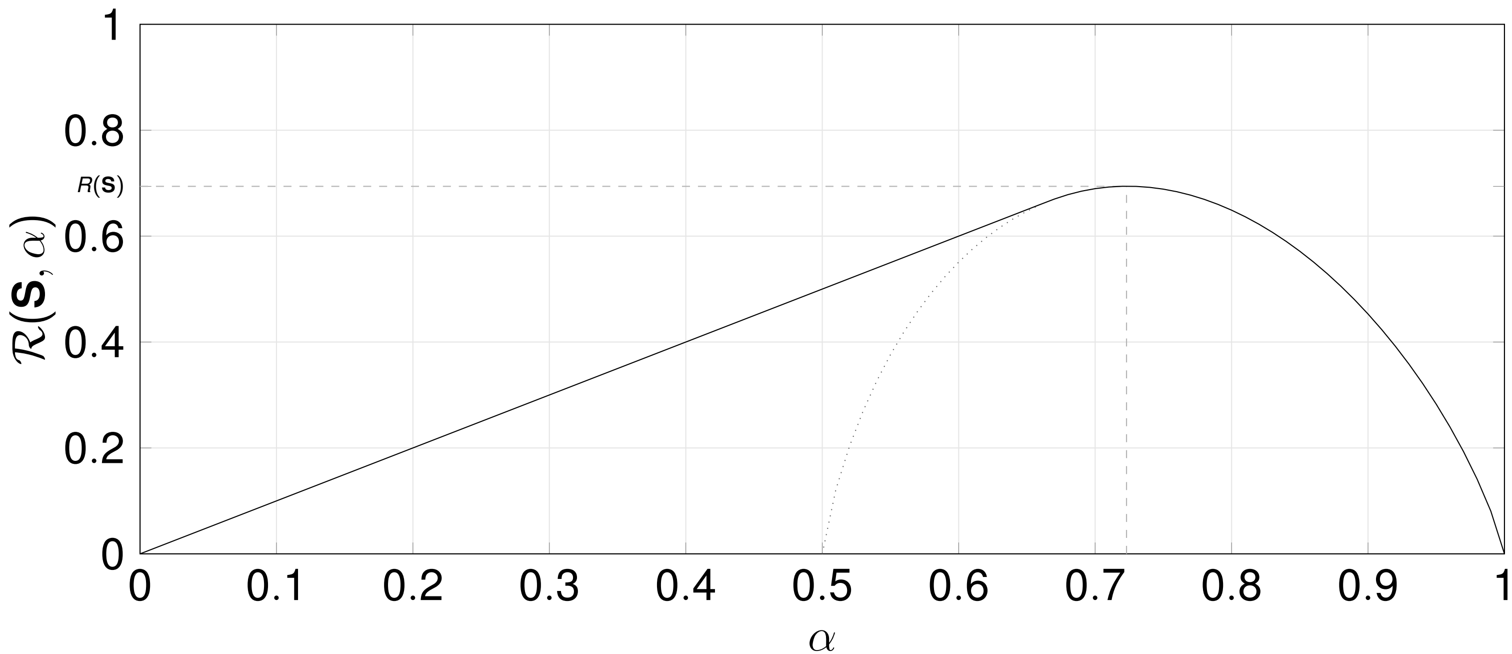
Equivalences

$$\underset{\text{largest synthesis code}}{N(\mathbf{S}, T, n)} \longleftrightarrow \#\text{length-}n \text{ paths through } G(\mathbf{S}) \longleftrightarrow \#\text{length-}n \text{ subsequences}$$

Fixed Length Sequences

- Example: Binary alternating sequence $\mathbf{S} = (\text{ACACAC}\dots)$

$$R(\mathbf{S},\alpha) = \begin{cases} \alpha, & \alpha \leq \frac{2}{3} \\ \alpha h(\alpha^{-1}-1), & \alpha > \frac{2}{3} \end{cases} .$$



Conclusion

Summary

- Time-efficient synthesis
- Optimal synthesis codes $\leftrightarrow$ paths through graph $\leftrightarrow$ subsequences
- Solution for arbitrary periodic sequences using constrained coding techniques
- Implies results for asymptotic growth rate of number of subsequences

Outlook

- Incorporate error-correction
- Variable synthesis sequence

Thank you!