Exploiting Redundancy in the Genetic Code

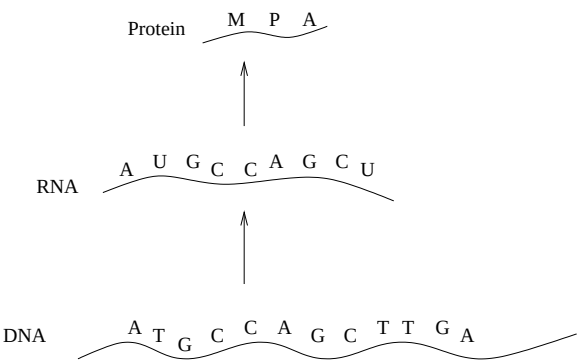
Steven Skiena

Department of Computer Science
State University of New York
Stony Brook, NY 11794–4400

http://www.cs.sunysb.edu/∼skiena

Protein Transcription

DNA sequences act as templates for building proteins according to the *triplet code*.
There are $4^3 = 64$ possible sequences or *codons* defined by three consecutive nucleotide bases.



The triplet code maps each of these 64 sequences to one of the 20 different amino acids or the stop symbol.

The Triplet Code

	U			C			A			G			
U	UUU	Phe	[F]	UCU	Ser	[S]	UAU	Tyr	[Y]	UGU	Cys	[C]	U
	UUC	Phe	[F]	UCC	Ser	[S]	UAC	Tyr	[Y]	UGC	Cys	[C]	C
	UUA	Leu	[L]	UCA	Ser	[S]	UAA	*STOP*		UGA	*STOP*		A
	UUG	Leu	[L]	UCG	Ser	[S]	UAG	*STOP*		UGG	Trp	[W]	G
C	CUU	Leu	[L]	CCU	Pro	[P]	CAU	His	[H]	CGU	Arg	[R]	U
	CUC	Leu	[L]	CCC	Pro	[P]	CAC	His	[H]	CGC	Arg	[R]	C
	CUA	Leu	[L]	CCA	Pro	[P]	CAA	Gln	[Q]	CGA	Arg	[R]	A
	CUG	Leu	[L]	CCG	Pro	[P]	CAG	Gln	[Q]	CGG	Arg	[R]	G
A	AUU	Ile	[I]	ACU	Thr	[T]	AAU	Asn	[N]	AGU	Ser	[S]	U
	AUC	Ile	[I]	ACC	Thr	[T]	AAC	Asn	[N]	AGC	Ser	[S]	C
	AUA	Ile	[I]	ACA	Thr	[T]	AAA	Lys	[K]	AGA	Arg	[R]	A
	AUG	Met	[M]	ACG	Thr	[T]	AAG	Lys	[K]	AGG	Arg	[R]	G
G	GUU	Val	[V]	GCU	Ala	[A]	GAU	Asp	[D]	GGU	Gly	[G]	U
	GUC	Val	[V]	GCC	Ala	[A]	GAC	Asp	[D]	GGC	Gly	[G]	C
	GUA	Val	[V]	GCA	Ala	[A]	GAA	Glu	[E]	GGA	Gly	[G]	A
	GUG	Val	[V]	GCG	Ala	[A]	GAG	Glu	[E]	GGG	Gly	[G]	G

Implications of the Triplet Code

Each of the 20 amino acids has as few as 1 (e.g. tryptophan TGG) or as many as 6 (e.g. arginine AGA, AGG, CGA, CGG, CGT, CGC) corresponding codons.

Thus the number of possible RNA sequences coding for a given protein grows exponentially in the length of the protein, e.g.. 10^{75} legal sequences for a 147-residue hemoglobin protein.

Why is one particular sequence chosen from this huge space of possibilities?

Alternately, *can biotechnology exploit this redundancy if we can design the 'best' coding sequence?*

Why Does Nature Use a Given Sequence?

Different theories exist as to why a particular coding sequence is selected by nature:

- Sequences evolve to use organism-preferred codons – reflecting its set of tRNAs.
- Coding helps regulate gene expression, by beginning with common/scarce codons.
- Coding evolves to protect from potential mutations by maximizing the mutation distance to broken proteins.

But other notions of best are possible. . .

Design Criteria for Artificial Genes

Optimizing certain technical criteria can have surprising biological motivations:

- Matching a given codon/pair distribution $\rightarrow$ maximizing protein expression.
- Folding into a desired 2- or 3-dimensional structure $\rightarrow$ RNA secondary structure optimization, code design for DNA computing.
- Minimizing the occurrence of certain patterns $\rightarrow$ increasing the robustness of bacteriophages.

RNA Folding

The folding problem seeks the structure or shape of a given sequence.

The shape of certain RNAs plays a major role in determining its interaction with other molecules, for example tRNAs.

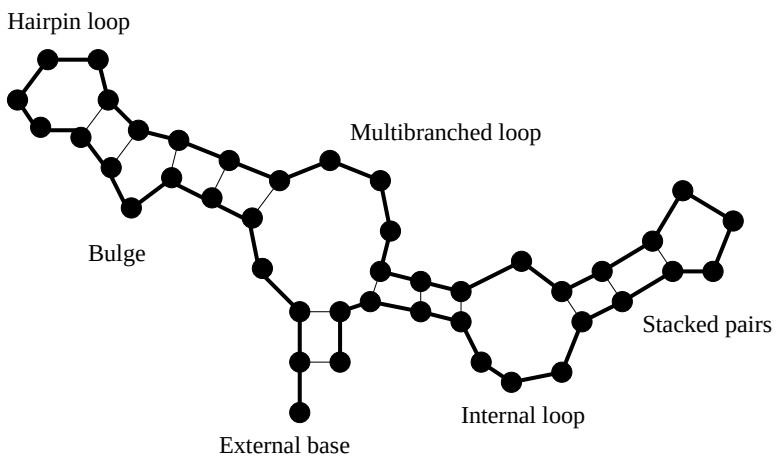
Folding occurs in both proteins and RNA, although the issues are different.

Since RNA is single-stranded, its component bases tend to bond with other bases analogously to the bonds formed in double-stranded DNA (A-U, C-G).

The set of all these pairs constitutes the *secondary structure* of an RNA molecule.

The Zuker-Turner RNA Folding Model

In this widely used model, binding pairs partition the RNA strand into nested loops.



A complicated energy function (derived from laboratory experiments) is used to measure the binding strength of short patterns.

Dynamic programming optimizes the secondary structure under the observed energy functions.

Zuker, et.al [LZP-99,ZS-84] has developed successful RNA secondary structure prediction algorithms – on average, 73% of known base pairs on domains of fewer than 700 nucleotides [MSZT-99].

RNA Folding Recurrences

Let $V[i,j]$ denote the minimum energy of the structure closed by $i \cdot j$.
$WM[i,j]$ denote the contribution of subsequence i to j, to a multiloop which contains i to j.
The energy of a hairpin loop closed by $i \cdot j$ is given by $eH(i,j) = \{stack(i,j) + hPenalty(j-i-1)\}$
The energy of the structure closed by pair $i \cdot j$ in a helix is given by $eS(i,j) = stack(i,j) + V(i,j)$
The energy of an internal loop with external closing pair $i \cdot j$ and internal closing pair $k \cdot l$ is

$$eI(i,j) = \min_{i<k<l<j} \{stack(i,j) + stack(k,l) + V(k,l)\}$$

The energy contribution of sequence from i to j to a multiloop is given by $WM(i,j) =$

$$\min \left(V(i,j) + b, WM(i,j-1) + c, WM(i+1,j) + c, \min_{i<k\leq j} \{WM(i,k-1) + WM(k,j)\} \right)$$

The energy of a multiloop closed by $i \cdot j$ is given by

$$VM(i,j) = \min_{i+1<k\leq j-1} \{WM(i+1,k-1) + WM(k,j-1) + a\}$$

Combining these recurrences, we arrive at the composite recurrence for the minimum energy:

$$V(i,j) = \min\{eH(i,j), eS(i,j), eI(i,j), VM(i,j)\}$$

The Optimal Energy RNA Sequence Problem

Among all RNA sequences coding for a given protein P, which has the minimum energy (maximum structure)?
Which has the maximum energy (minimum structure)?
Can we design a sequence to have a specified secondary structure, e.g. Vienna RNA package's inverse fold server (Hofacker,et.al-94)?
We envision that the optimal sequences have applications in biotechnology (e.g. high/low temperature environments).

Maximizing Structure

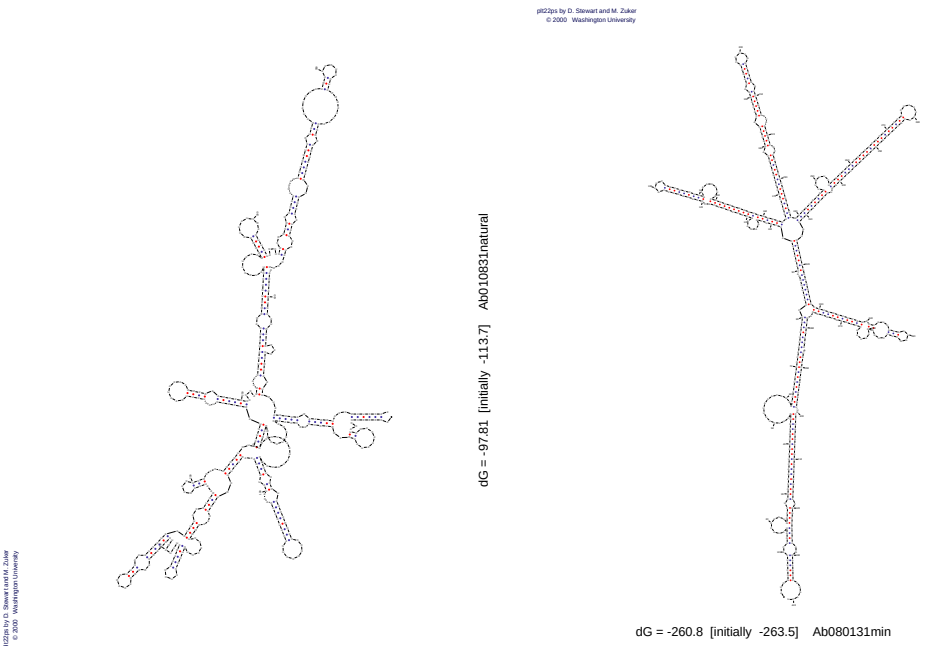
We have developed a dynamic programming algorithm for the *minimum energy RNA sequence problem*, which takes a target protein P, and seeks the most stable (lowest energy) RNA sequence which codes for P.

Asymptotically, our algorithm runs in $O(n^3)$ time, the same time as state-of-the-art programs [LZP-99] which only fold a given RNA sequence. But it has a big constant factor...

We compute the optimal RNA sequence among the 10^{75} legal sequences for a 147-residue hemoglobin protein in about an hour on a desktop computer.

We exploit the fact that each residue can be coded in only a constant number of ways.

Before and After



The new encoding requires twice as much energy to unfold.

Finding String Edit Distance

Our algorithm is best illustrated by looking at a simpler and more familiar dynamic programming problem.

The *edit distance* between strings S and T is the minimum number of character insertions, deletions and substitutions sufficient to transform S to T.

Let $D[i,j]$ = the minimum number of differences between $S_1, S_2, \ldots, S_i$ and the segment of T ending at j.

$D[i,j]$ is the *minimum* of the three ways to extend:

If $S_i = T_j$ then $D[i-1, j-1]$ else $D[i-1, j-1] + 1$ (match or do not match)

$D[i, j-1] + 1$ (extra character in S)

$D[i-1, j] + 1$ (extra character in T)

What T is Closest To S?

Dynamic programming can find the minimum distance m-string from S in $O(nm)$ time for *constant-sized alphabets*.

Let $D[i, j, c]$ = the minimum edit cost between $S_1, S_2, \ldots, S_i$ and the (unknown) segment of T *ending in character* c.

$D[i, j, c]$ is the *minimum* over all $\sigma \in \Sigma$ of the *minimum* of the three possible ways to extend smaller strings:

$D[i-1, j-1, \sigma] + d[c, S_j]$ (match?)

$D[i-1, j, c] + 1$ (extra character in S)

$D[i, j-1, \sigma] + 1$ (extra character in T)

Although we use similar techniques, our RNA design algorithm is substantially more involved, requiring several free characters and incorporating codon constraints.

Minimizing Structure

One can theorize that nature prefers to minimize RNA secondary structure since energy is required to break these bonds during transcription.

However, the algorithmics of finding the minimum energy coding is considerably different than that of finding a maximum energy coding.

To minimize the complications of specific energy functions, we consider a problem which captures the essence of minimum energy coding, specifically the case where the RNA has no secondary structure at all.

String with No k-Repeat Problem

Input: An alphabet Σ, an integer k, and an ordered collection of subsets $S = \{s_1, \ldots, s_n\}$, where $s_i \subset \Sigma$, for all $1 \leq i \leq n$.
Question: Does there exist a string T such that $|T| = n$, $T[i] \in s_i$ for all $1 \leq i \leq n$, and there does not exist a $1 \leq u < v \leq n$ such that $T[u + j] = T[v + j]$ for all $0 \leq j < k$?

This is a reasonable model for minimum energy coding because (1) both problems seek to build a string where each character position is restricted to certain subset of possible characters, and (2) both problems seek a string which avoids a pair of complementary patterns of sufficient length to bind.

Hardness Proof

We use a reduction from the directed Hamiltonian path problem.
Our reduction constructs a string on an alphabet of vertices and spacers, with two parts:

- The header of T is completely specified, and includes patterns corresponding to all *non-edges* and all *non-vertices* in G
- The characters in the second portion of T will be left fairly unrestricted – this provides the "tape" to write the Hamiltonian path of the graph specified in the header.

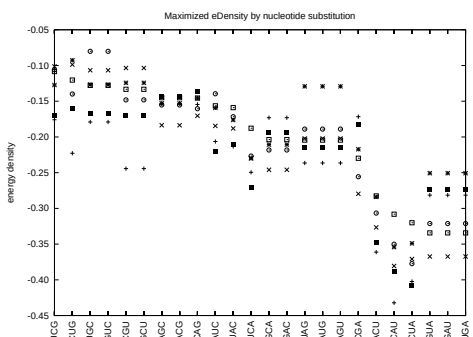
Finding a character assignment which avoids all length-3 patterns in the header means that:

- Each vertex pair between commas must represent an edge in G (i.e. avoid a non-edge in G).
- Each adjacent edges in the tour must end and start on the same vertex (i.e. avoid any non-vertex pattern).
- Visits each vertex exactly once (since none of the $V - 1$ pairs flanking ':' spacers are allowed to repeat).

Thus there is a string satisfying the constraints iff G has a Hamiltonian path.

Base Order To Minimize Structure

CG bonds are stronger than AU or GU bonds, so sequences with the lowest CG content have the highest energy (least stability). Thus using As wherever possible should construct high energy sequences.



Every nucleotide preference ranking implies a ranking among synonymous codons. No single ordering is always best but we use $AUCG$ to minimize structure.

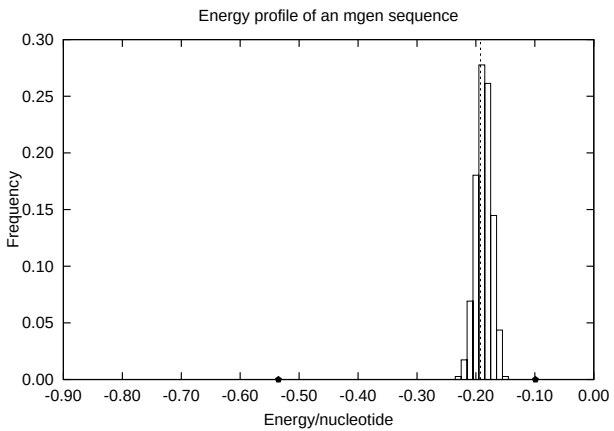
Local Structure Avoidance (LSA) Method

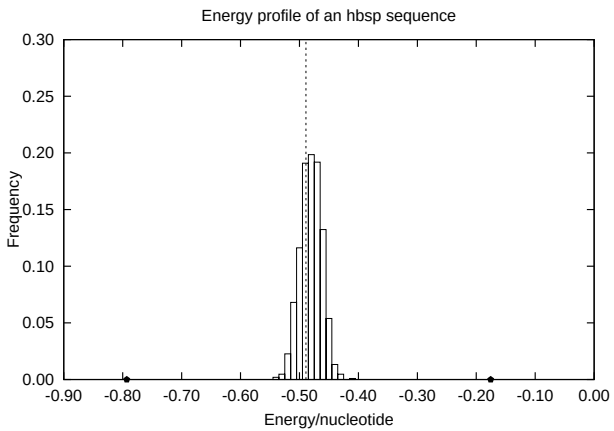
An alternate approach uses exhaustive search to identify an incremental extension that minimizes local secondary structure within a fixed-length suffix of S.

With an exponential-time dependence on variable-window length, $|V| > 20$ becomes impractical impractical.

Both heuristics prove effective in producing low-structure sequences. LSA produces better results in 73% of the sequences.

What Freedom Does Nature Have?



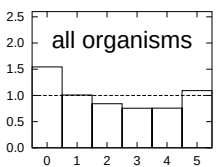


The bell-shaped energy distribution for 1,000 randomly encodings for 438-bp from *M. genitalium* and 435-bp *Halobacterium sp. NRC-1*.

The vertical dashed line gives the wildtype energy while solid circles show our minimum/maximum energies.

What Does Nature Do?

To test for evolutionary pressure on encoding energy, we compare the wild-type energy against five randomly selected sequences coding for the same protein.



Thus the wildtype sequence occupies one of six possible “energy ranks” – the number of synthetic sequences which are more stable than it.

Sequences with a preference to stability are in the left bin. A uniform distribution would show the absence of selective pressure.

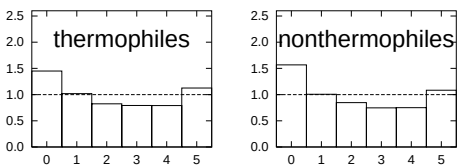
Bacterial Stability Signatures

Observed Biases

- *Clear Bias for Stability* – In 34 of 36 cases, the number of stable structures exceeds the expected value. The structure most strongly skewed toward stability belongs to *M. genitalium*.
- *Secondary Bias for Instability* – Highly unstable structures are over-represented in 19 of 36 cases. In 9 they are the primary mode; in 10 they are the secondary mode. The structures most strongly skewed toward unstable sequences belongs to *X. fastidiosa*.

Thermophiles

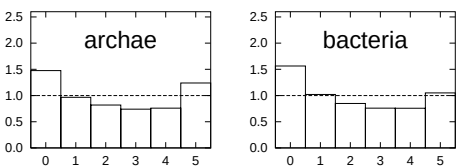
We initially conjectured that selection for stable structure might occur more intensely under extreme temperatures and pressures.



The seven thermophiles in our data set (*A. pernix*, *A. fulgidus*, *A. aeolicus*, *M. jannaschii*, *P. abyssi*, *P. horikoshii*, *T. maritima*) do not display different patterns of stability than other bacteria.

Archae vs. Bacteria

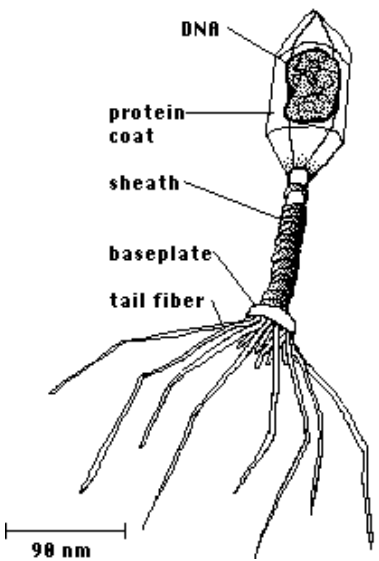
We initially conjectured that selection for stable structure might differ between archae and bacteria due to their long divergence.

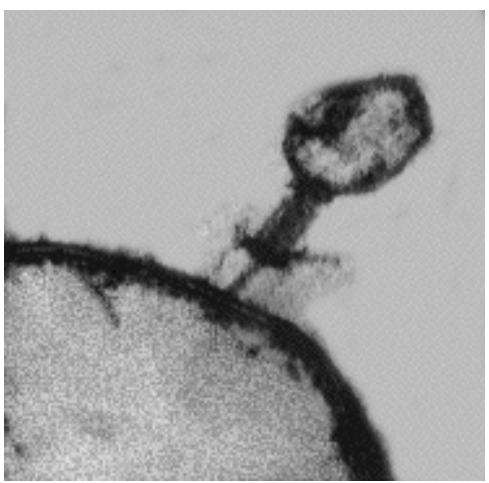


No discernible bias between archae or bacteria could be identified.

Bacteriophages

Bacteriophages are viruses which kill bacteria by injecting DNA.





Medical Applications

There is increasing interest in medical applications of phages, motivated by the growing problem of antibiotic resistance.
Phages have been widely used in the former Soviet Union to fighting infections since the 1930's.
Several hurdles must still be overcome before phages can successfully compete with antibiotics…

Problems with Bacteriophages

- Phages tend to be narrowly targeted to specific bacteria.
- Mammalian circulatory systems are adept at removing phages from the blood stream.
- Crude phage lysates often contain toxins.

Recent work towards isolating long-circulating bacteriophages [Merril et.al. 1996] shows considerable promise in overcoming these hurdles.
We propose a complementary approach to building better phages, by engineering them to avoid a primary bacteriological defense mechanism.

Restriction Enzymes

Restriction enzymes cut unmethylated DNA at specific patterns, or *cutter sequences*.

E.g.. EcoRI cuts at *GAATTC*, and EcoAI cuts at *GAGNNNNNNGTCA*.

Cutter sequences range in length from 2 to 16 bases, and may include wildcard symbols.

Such enzymes, isolated from bacteria, have become a fundamental tool in biotechnology.

However, their primary biological role is as a defense mechanism against phages.

Bacteria deploy restriction enzymes to cut the injected phage DNA into fragments before it becomes biologically active.

Exploiting the Triplet Code

We propose increasing the effectiveness of a therapeutic phage by engineering its genome to minimize the number of restriction sites for a given set of cutter sequences.

One must be careful not to change the phenotype of the resulting phage, however.

We seek the coding sequence which minimizes the number of restriction sites while still coding for all the desired proteins.

The intergenic regions of bacteriophages may or may not be recoded, but are typically very short, a tribute to the evolutionary advantage of minimizing genome length in phages.

The Minimum Restriction Site Genome Problem

Input: The sequence S of a given phage P, with genes $G = \{g_1, \ldots, g_k\}$, a set of restriction enzymes cutters $E = \{e_1, \ldots, e_m\}$, and an integer x.

Output: A DNA sequence S' where $|S| = |S'|$, such that for each gene $g_i \in G$ in S and S' code for identical proteins, all non-coding regions of S are conserved in S', and there are at most x of restrictions sites in S' over E.

Bad news – NP-Complete!

The reduction is from *3-satisfiability*.

Note that every CNF clause has one pattern of literals (all false) which causes it to fail.

The *gene to synthesize / logical variables* will be represented by a string of length n, where each position can either be T or G.

The *enzyme to cut / logical clause* will have don't care bases except at the three symbols in the clause, cutting only when all relevant literals are negated.

For example, clause $(v_1, \bar{v_2}, v_3)$ yields enzyme GTG.

Good News – Dynamic Programming!

Let $M[i, W]$ be the minimum number of enzyme cuts possible to encode the first i bases of S, where the last l bases of S' are defined by the string $W = w_1 w_2 \ldots w_l$.

$$M[i, W] = \min_{\alpha \in ACGT} M[i-1, \alpha w_1 \ldots w_{l-1}] + cuts(W)$$

where $cuts(W)$ is the number of restriction sites created by the implicit enzyme set whose cutter matches a suffix of W.

The number of restriction sites in the optimal sequence, $M[n]$, is found by testing all possible suffix windows, i.e.:

$$M[n] = \min_{i=0}^{4^l-1} M[n, W_i]$$

Since the longest known cutter is only 16 bases, this is a tractable computation.

Complications

- *Overlapping genes* – It is fairly common for two or more genes to share a given region on a genome.
- *Genes on alternate strands* – Most phages exhibited genes on both the upper and lower strands of the genome.
- *Non-palindromic cutters* – Most but not all cutters for restriction enzymes are *palindromic*, meaning that the cutter is equal to its reverse complement.

Efficient implementation of the sparse dynamic program requires some care.

Our implementation, which is 2000 lines of C, can run each of our phage / enzyme set pairs in a few minutes on a modest desktop workstation.

Phages Studied

Phage	Length	Coding	Non-coding	Genes	Layers
186	30624	28380	2244	46	4
A118	40834	38461	2373	72	2
Alteromonas	10079	9318	761	23	2
C2	22172	20532	1640	39	2
Chp1	4877	2927	1950	12	6
Cp-1	19343	16715	2628	28	2
D29	49136	44191	4945	77	2
G4	5577	5295	282	11	4
HK022	40751	24781	15970	35	2
HK97	39732	34641	5091	60	2
HP1	32355	29645	2710	41	2
L2	11965	9140	2825	14	2
MAV1	15644	14115	1529	15	2
MS2	3569	3246	323	4	2
MX1	4215	4017	198	4	2
Mu	36717	33276	3441	53	3
N15	46375	41847	4528	60	2
NL95	4248	4053	195	4	2
P2	33593	30748	2845	42	2
P4	11624	9457	2167	13	2
PZA	19366	18181	1185	27	2
SPP1	44007	38883	5124	102	4
Streptococcus	40739	32311	8428	38	2
bIL170	31754	28806	2948	63	2
bIL67	22195	19846	2349	37	2
fs-2	8651	7334	1317	9	2
lambda	48502	40842	7660	67	2
psiM2	26111	22588	3523	31	2

Enzyme Sets

REBASE contains about 3409 enzymes with 255 distinct cutter sequences.

- *All cutter sequences* – These sets consist of all known sequences of particular length, all simple cutters (ALL-SIMPLE), all cutters including ambiguous bases (NON-SIMPLE), and all simple cutters of length 4, 6, and 8 (ALL-SIMPLE-4, -6, and -8).
- *Dense organism-specific sequences* – These sets constitute the host-specific sets of enzymes for the six genus with the largest known enzyme sets, from 49 to 26 distinct cutters each. *Pseudomonas*, *Escherichia*, *Streptomyces*, *Helicobacter*, *Neisseria*, and *Thermus*.

Why Large Enzyme Sets?

Large enzyme sets occur because restriction enzymes move laterally across species.

The T7 phage can infect *E. coli* unless the bacterium has previously been infected by the *P1* phage, which supplies it with the restriction enzyme *Eco P1*.

Bacteriophage genomes are substantially depleted of palindromic sequences, typically associated with restriction sites [Karlin 1992].

Bacterial genomes avoid cut sites associated with their own enzymes because of the occasional failure of their own methylation systems [Gelfand and Koonin 1997]

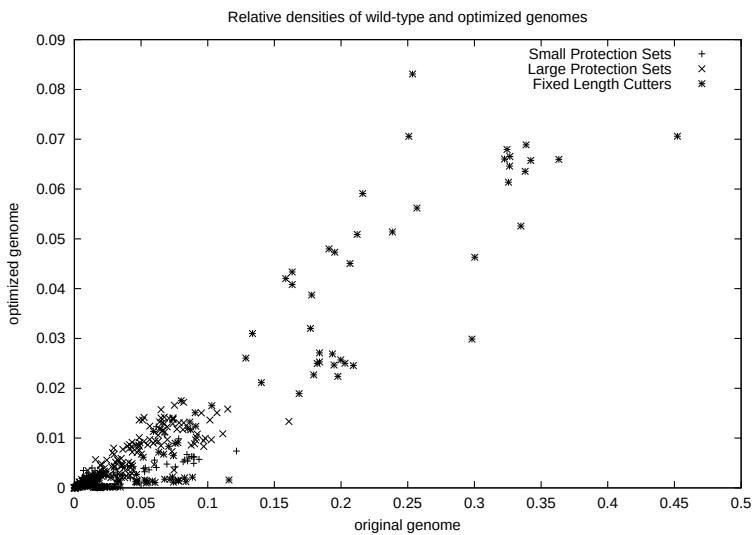
Results By Cutter Length

	4-cutters		6-cutters		8-cutters		All simple	
Phage	wild	opt	wild	opt	wild	opt	wild	opt
186	2514	61	695	7	9	0	10311	1872
A118	2114	45	716	8	12	0	8156	1957
Alter.	658	11	151	2	4	0	2489	567
C2	963	27	323	2	4	0	3354	890
Chp1	151	18	80	6	0	0	633	173
Cp-1	780	45	221	15	1	0	3191	802
D29	5126	70	1523	8	9	0	19985	3120
G4	314	60	115	14	0	0	1343	440
HK022	1840	28	661	5	3	0	8377	1575
HK97	2482	60	869	8	8	0	11176	2287
HP1	1597	29	533	4	6	0	7070	1523
L2	492	13	178	0	3	0	1627	354
MAV1	854	15	267	1	7	0	2499	452
MS2	241	21	108	4	1	0	1060	216
MX1	272	7	121	1	0	0	1206	186
Mu	2585	51	736	9	4	0	11273	2292
N15	3267	59	1211	5	9	0	14324	2751
NL95	339	5	126	1	0	0	1357	213
P2	2396	43	690	4	4	0	10010	1887
P4	782	19	215	2	4	0	3086	611
PZA	959	29	307	2	7	0	3758	819
SPP1	2603	263	736	61	5	0	9752	2745
Strep.	1484	40	560	5	2	0	6310	1529
bIL170	1355	40	446	5	5	0	4704	1176
bIL67	931	20	287	2	3	0	3146	834
fs-2	429	10	119	2	0	0	1884	412
lambda	3005	89	987	10	9	0	13249	2775

Results By Organism Specific Enzymes

	Pseudomonas		Escherichia		Streptomyces		Helicobacter	
Phage	wild	opt	wild	opt	wild	opt	wild	opt
186	946	42	652	84	335	48	2915	274
A118	370	20	308	46	167	25	1786	330
Alter.	133	3	60	3	18	1	550	63
C2	123	6	102	14	40	6	886	167
Chp1	54	1	41	4	30	3	184	31
Cp-1	52	2	60	12	9	2	723	153
D29	3301	159	3237	322	1928	194	7107	589
G4	103	15	63	12	29	9	434	91
HK022	937	50	894	112	538	67	2404	206
HK97	1120	65	1037	145	618	86	3149	334
HP1	316	18	287	47	171	27	1433	226
L2	148	24	145	52	92	32	449	83
MAV1	111	5	89	17	43	7	454	80
MS2	143	12	110	10	77	9	348	49
MX1	154	14	120	27	79	18	394	40
Mu	1174	70	1001	145	544	84	3044	339
N15	1630	101	1500	207	896	125	4055	396
NL95	174	12	142	24	91	16	451	44
P2	956	39	636	71	294	42	2774	272
P4	316	14	252	26	129	16	828	81
PZA	176	4	141	13	71	4	1076	163
SPP1	870	125	792	188	461	101	2920	645
Strept.	261	27	276	70	115	38	1572	270
bIL170	166	6	159	19	54	9	1143	242
bIL67	88	4	76	10	22	3	821	158
fs-2	92	3	54	2	19	1	476	62
lambda	1334	79	1137	173	636	106	3765	441
psiM2	945	56	922	116	534	62	2295	308

Density of sites in optimized vs. wild-type genomes



Results

We have the freedom to remove *every* instance of every 8-cutter from the coding regions of every phage we considered.
We remove 90% of all 4-cutter and 6-cutter sites from every phage.
We typically remove 80% or more of *all* sites for *all* simple cutters.
There seems to be no significant difference in our ability to remove simple or ambiguous cutter sequences.
We typically remove about 90% of all wild-type sites even in the densest organism-specific enzyme sets.
We believe that these ideas may provide a method to increase the host range of specific phages.

Why Hasn’t Evolution Solved the Problem?

If phages benefit from genomes with the minimum possible number of restriction sites, why hasn’t evolution removed all of them?

Extensive evidence shows that palindromic subsequences are indeed selected against in bacteria and phage genomes.

We propose a simple evolutionary model which may explain why so many still remain. . .

An Evolutionary Model

Suppose we model a genome as an n-bit binary string, where 0 denotes a restriction site and 1 no site.
Random mutations/flips will leave half the sites active, while a very strong bias is needed to remove all of them.
Equilibrium is achieved when the probability of forward mutations equals that of backwards mutations, at

$$i_{eqi} \approx \frac{n}{(1 - p_{cut})^2 + 1} \tag{1}$$

where p_{cut} is the probability that any given site is fatally cut.
If $p_{cut} \to 0$, there is no bias and $i_{eqi} = n/2$ sites are inactive.
If $p_{cut} \to 1$, there is such strong bias that all $i_{eqi} = n$ sites are inactive.

How Often are Sites Cut?

Since typical phage genomes can have dozens of restriction sites, p_{cut} must be fairly low for any significant phage population to survive.

Indeed, studies show that that between $1/10$th to $1/100,000$th of bacteria survive a phage infection.

Assuming a survival value of $1/1000$ for a phage with ten restriction sites yields $p_{cut} \approx 0.5$, which places equilibrium at removing only 80% of all restriction sites.

Similar calculations for a phage with 100 restriction sites yields $p_{cut} \approx 0.06$ and equilibrium at only 53% of all restriction sites.

Open Problems

- Conforming to a given codon distribution.
- Other optimization criteria
- Laboratory implementation / experimentation.

For Further Reading