segemehl: a mapping tool for HTS reads

Steve Hoffmann
steve@bioinf.uni-leipzig.de

February 18, 2009

1. HTS: High Throughput for Hypothesis Trimmed Science?
   - High Throughput Research
   - Other Technology
   - Applications & Problems
2. Mapping
   - Goals and approaches
   - A new model
3. Results
   - Implementation
   - Simulations
   - solexa & 454 data
4. Summary
   - Summary

Example: High throughput

A random experimental setup

- evacuated lab

Example: High throughput

A random experimental setup

- evacuated lab
- sufficiently large receptacle

Example: High throughput

A random experimental setup

- evacuated lab
- sufficiently large receptacle
- 1 liter C_2 solution (11.5%)

Example: High throughput

A random experimental setup

- evacuated lab
- sufficiently large receptacle
- 1 liter C_2 solution (11.5%)
- 50g fatty acids

Example: High throughput

A random experimental setup

- evacuated lab
- sufficiently large receptacle
- 1 liter C_2 solution (11.5%)
- 50g fatty acids
- 2 liters of a solution labled "Aceto Balsamico"

Example: High throughput

A random experimental setup

- evacuated lab
- sufficiently large receptacle
- 1 liter C_2 solution (11.5%)
- 50g fatty acids
- 2 liters of a solution labled "Aceto Balsamico"
- 1 liter of beef stock solution
- sugar, herbs

Heat and mix constantaneously for 4 hours!

Example: High throughput (cont'd)

Results

1. the solution called "Aceto Balsamico" contains vinegar (majority voting)

Example: High throughput (cont'd)

Results

1. the solution called "Aceto Balsamico" contains vinegar (majority voting)
2. one underaged test person started puking (Ellias) [salt-bias!]

Overview

system	by	placed	price	max. len. (bp)	reads/run
454	Roche	2005	$500000	400	1 million
Solexa	Illumina	2006	$400000	50	50 million
SOLiD	ABI	2007	$600000	50	50 million

solexa/illumina

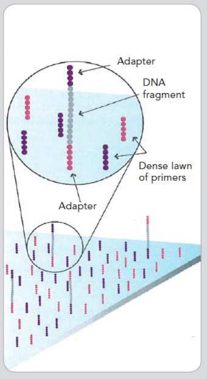


Figure: Illumina: reads immobilized and bridge-amplified.

454

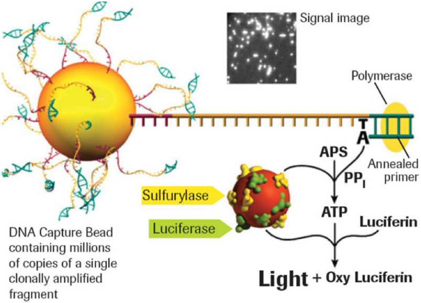


Figure: 454 pyrosequencing on beads: light reaction is induced by sulfurylases and luciferases.

SOLiD

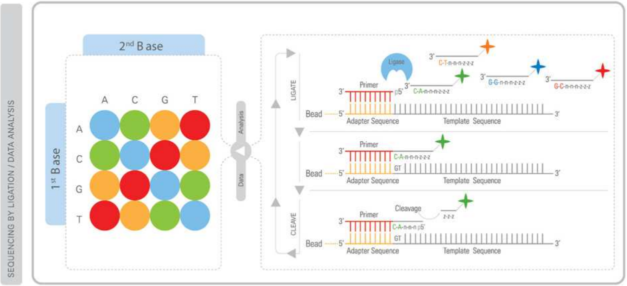


Figure: SOLiD sequencing by ligation. After bead amplifcation templates are interrogated by probes

one might want to buy a machine for ...

- *De Novo* Sequencing
- **targeted resequencing**
- **whole genome resequencing**
- **gene expression profiles**
- **small RNA analysis**
- **whole transcriptome analysis**

When the sales representative has left ...

you may experience:

- sequences are just too short for de novo assembly
- **significantly higher error rates for solexa**
- **read length dependent error rates for 454**
- considerable GC-bias for solexa sequences
- weak correlation among 454 and Solexa results
- **indels predominant error type in 454 sequences**

Huse et al. (2007) Accuracy and quality of massively parallel DNA pyrosequencing. Genome Biology 8:R143.
Dohm et al. (2008) Substantial biases in ultra-short read data sets from high-throughput DNA sequencing. Nucl Acids Res 36:e105.

Goals in short sequence mapping

1. error tolerant mapping (mismatches **and** indels)
2. tolerating trailing contamination (eg. poly-A, primers)
3. sensitive mapping (report multiple hits)
4. size independent mapping
5. fast
6. small memory footprint

Current methods

Popular tools for short sequence mapping

1. assume a fixed number of allowed errors
2. consider only mismatches
3. are mostly limited to a maximum read length (illumina)

and often use fast hash-lookup tables (*e.g.* `MAQ`, `SOAP`) or burrows-wheeler transformation (*e.g.* `BWA`, `Bowtie`)

Changing the perspective

Instead of enumerating mismatches (and differences in general) one might look at those parts of a read that **do not** contain errors.
First, lets look at some properties of ”error-free” substrings ...

A magic substring

Definition (characteristic substring)

Let S be a target sequence, P a read and f a substring of P. $occ_S(P)$ holds all occurences of P in S. f is a characteristic substring with respect to S if there is some $0 \leq d < m$ statisfying

$$\{i + d \mid i \in occ_S(f)\} = occ_S(P). \tag{1}$$

Greedy search in erroneous patterns

Lets turn to an erroneous version of $\hat{P}$. We might succeed in finding a characteristic substring ...

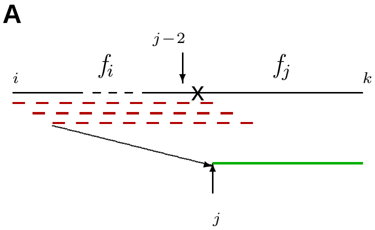


Figure: The success of a greedy method depends on the length of "error-free" substrings ... . (A) f_i is a rather short substring. (B) f_i is a sufficiently long substring.

Estimation of the length

Theorem (length of characteristic substring)

Assuming **uniform** *distribution of chars along the subject sequence, the minimum length of a characteristic substring can be estimated by*

$$\arg\min_{l}\{\mathbb{E}(l \mid S, \Sigma) \leq 1\} \approx \frac{\lg(|S|)}{\lg(\sigma)} \tag{2}$$

Estimation of the length (folklore)

length of characteristic substring.

The probability of some substring f of length l in S is given by $P(f \mid S, \Sigma) = (\sigma^{-1})^l$ and the expectation value to find such a substring in a subject sequence boils down to

$$\mathbb{E}(l \mid S, \Sigma) = (\sigma^{-1})^l \cdot |S| \tag{3}$$

since the expectation value of f only depends on its length l. Setting

$$\mathbb{E}(l \mid S, \Sigma) = (\sigma^{-1})^l \cdot |S| = 1 \tag{4}$$

we derive $\sigma^l = |S|$ and $\lg_\sigma(|S|)$ yields the solution. □

Error-free substrings

Definition (error-free substrings)

Let $\mathcal{A}$ be an optimal sequence alignment of $\hat{P}$ and P with a sequence of eops $(\alpha, \beta) \in (\Sigma^1 \cup \{\epsilon\}) \times (\Sigma^1 \cup \{\epsilon\}) \setminus \{(\epsilon, \epsilon)\}$ such that $P = \alpha_0 \dots \alpha_h$ and $\hat{P} = \beta_0 \dots \beta_h$. Then a set of differences is given by

$$\mathcal{D} = \{i \mid (\alpha_i, \beta_i) \in \mathcal{A}, \alpha_i \neq \beta_i\}. \tag{5}$$

Hence, the set of error-free is given by

$$\mathcal{F} = \{(i,j) \mid i \leq k \leq j : k \notin \mathcal{D} \wedge i-1, j+1 \in \mathcal{D}\} \tag{6}$$

An old concept revisited: greedy matching statistics

Given a read P of length m, the matching statistics reports the longest common prefix (lcp) with S for **each suffix of** P and returns exactly one hit position.

The implementation of this concept can easily be modified to report all hits.

Task: detect characteristic substrings

From recent analysis we know:

1. sequencing error rates increase towards the end of the read
2. contaminations can occur at 3-prime and 5-prime ends

If those error types were the only one, we would easily find characteristic error-free substrings using a greedy method:

Example: terminal errors

35 bp read, 10 mismatches $\Rightarrow$ error free substring of length 25.

But what about errors in the middle of a read?

The matching stem (informal)

Assume we are mapping a substring of P, namely P_i, character by character to S. Each additional character match reduces (not always!) the number of positions in S, the substring can be mapped to. This sequence of shrinking sets is called matching stem.

In other words: the matching stem is the greedy matching path along the S.

The matching stem (formal)

Definition (matching stem)

A matching stem $\mathcal{M}_i$ for a suffix P_i with some target S is a family of at most $m-i$ non-empty sets (segments) $\mathcal{M}_i^j = occ_S(p_i \dots p_{i+j-1})$, partially ordered by $(\mathcal{M}, \supseteq)$

$$\mathcal{M}_i = (\mathcal{M}_i^i, \mathcal{M}_i^{i+1}, \dots, \mathcal{M}_i^l) \tag{7}$$

such that $l \geq i$, $\mathcal{M}_i^j \neq \emptyset$ for all $j, i \leq j \leq l$, and $l = m$ or $\mathcal{M}_i^l = \emptyset$ with height $h(\mathcal{M}_i) = |\mathcal{M}_i|$.

The matching branch

To correct possible errors we have to **branch off** from that matching stem. Consider the optimal alignment

$$\mathcal{A}_{i,j} = (\beta_0 \rightarrow \gamma_0 \cdots \beta_h \rightarrow \gamma_h) \tag{8}$$

of P_i and S_j .

To allow the introduction of a first error at position $i+k$, the matching branch holds all elements of $\mathcal{M}_i^{k-1}$ that can be extended by $\gamma_k \neq \beta_k$. We denote:

$$_{\beta\rightarrow\gamma}\mathcal{B}_i^j \tag{9}$$

similarily: branches of branches ...

Neighboring matching stems are related

The matching stems for P_i and P_j might be related:

Related matching stems and lcp

Assume the query $P :=$`MISSISSIPPI`. If the suffix $P_1 :=$`ISSISSIPPI` has a longest common prefix of 5 with the target sequence, than P_2 has an *lcp* of **at least** 4. In our terminology:

$$\mathcal{M}_1^t \subseteq \mathcal{M}_2^{t-1} \ominus t \qquad 1 \leq t \leq 5 \tag{10}$$

In suffix trees and ESAs we can use suffix links to go directly from $\mathcal{M}_1^5$ to $\mathcal{M}_2^4$!

A heuristic for speed up

Related matching stems and lcp (cont'd)

After jumping directly from $\mathcal{M}_1^5$ to $\mathcal{M}_2^4$, we only have to evaluate the remaining characters `SIPPI` to complete the sequence $\mathcal{M}_2$.

We restrict branching to this rest, namely the tip $\mathcal{T}_2$, of the matching stem.

Do we have to consider branches for all characters to the end of the suffix? No! Average height of matching stem is

$$\frac{\lg(|S|)}{\lg(\sigma)}!!!$$

The model in a suffix tree

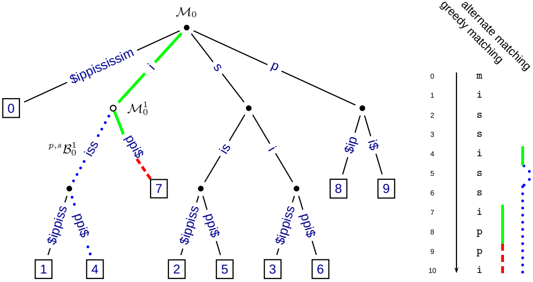
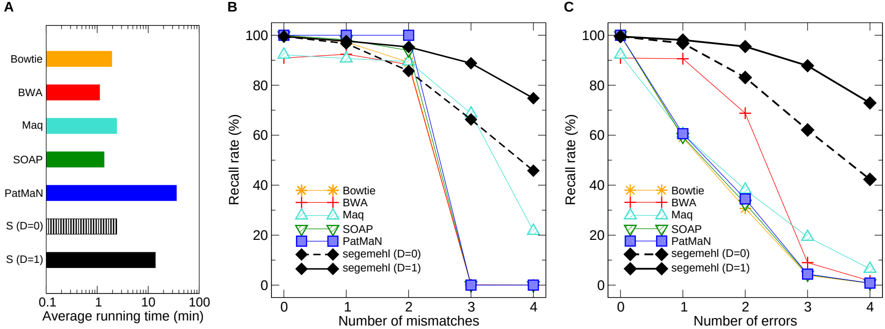


Figure: Evaluation of alternatives for the erroneous read `ipsissippi`. The branch ${}^{p\rightarrow s}\mathcal{B}_0^1$ denotes the alternative that accepts the mismatch $p \longrightarrow s$ at position 1 of the pattern

Implementation

1. based on enhanced suffix arrays (ESA)
2. for each substring of a pattern, the best scoring hits are reported to an alignment procedure
3. hits are omitted if the number of hits exceeds a given threshold (`maxocc`)
4. hits are omitted if they undercut a given score based E-value
5. final alignment: myers bit vector algorithm
6. alignments are reported if user defined accuracy criterion (default: 85%) is met.

Simulations



Simulations (cont'd)

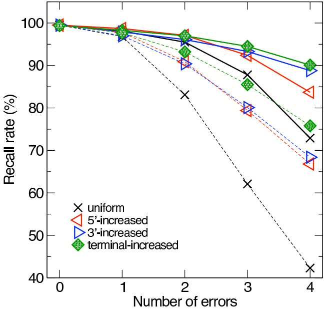


Figure: Different error distributions. `segemehl` works best for terminal errors.

Real-life data

		number of allowed errors			
		0	1	2	≥ 3
a) Human genomic data set ERR000475 (Illumina)					
`Bowtie`	16'011'867 (81%)	12'006'627	2'824'359	1'180'881	-
`MAQ`	16'762'361 (85%)	12'006'627	2'829'601	1'199'110	727'023
`segemehl`	18'191'858 (92%)	12'002'123	2'872'615	1'221'313	2'095'807
b) arabidobsis short RNA data set (454)					
`Bowtie`	26'969 (71%)	18'739	5'390	2'840	-
`MAQ`	29'987 (79%)	18'738	5'389	3'093	2'767
`segemehl`	35'942 (95%)	18'737	10'525	3'744	2'936

Summary

1. outcompetes other methods' recall rates if indels or more than 2 mismatches (contaminations) are involved.
2. heuristics to look for characteristic substrings → no fixed numer of errors
3. shows signifcantly better results not only for 454.
4. complexity for greedy matching (all lcps): $O(m)$.
5. complexity for matching with a single branch: $O(\sigma \cdot m(m+1))$.
6. increases exponentially (D=2 still suitable).
7. large memory footprint.
8. uncovered aspects: paired reads, quality values