Universität Basel

Cost Partitioning Techniques for Multiple Sequence Alignment

Mirko Riesterer, 10.09.18

Agenda.

Introduction

Multiple Sequence Alignment

- Biological sequences mutate during evolution
- Insertion, deletion, substitution
- Some mutations are more likely (A↔G / C↔T)
- Observe phylogenetic relationships

Introduction

Multiple Sequence Alignment

– Insert gaps within sequences
– Maximize correspondence between letters in columns

Sequences		Alignment
ACGTG		ACGT-G
ACTAG	→	AC-TAG
CGTAG		-CGTAG

Introduction

Judging the alignment quality

- Count matches/mismatches
- Score matrix
  - Point accepted mutation (PAM_n) matrix (Dayhoff et al., 1978)
  - Blocks substitution matrix (BLOSUM) (Henikoff and Henikoff, 1992)

```
Score matrix:

  A C T G –
A 0 4 2 2 3
C   1 4 3 3
T     0 6 3
G       1 3
-         0
```

Agenda.

1	Introduction
2	Formal Definition
3	Solving MSA
4	Combining Multiple Pattern Databases
5	Cost Partitioning
6	Experiments
7	Conclusion

Formal Definition

Family of sequences $S = \{s_1, \ldots, s_n\}$ over alphabet Σ and $\Sigma' = \Sigma \cup \{-\}$

Alignment

Matrix $A^{n \times m} = (\mathrm{a_{ij}})$, where

- $a_{ij} \in \Sigma'$
- a_i without $\{-\}$ is exactly s_i
- No column contains only $\{-\}$

Score matrix:

	A	C	T	G	-
A	0	4	2	2	3
C		1	4	3	3
T			0	6	3
G				1	3
-					0

Sequences:

```
ACT
CTG
```

Alignment A:

```
A C T _
_ C T G
```

$C^A = 3 + 1 + 0 + 3 = 7$

Formal Definition

Score matrix can be viewed as function $sub : \Sigma' \times \Sigma' \rightarrow \mathbb{N}$

Given alignment A and score matrix sub.

Pair score

$$\boldsymbol{C_{ij}^{A} = \sum_{k=1}^{m} sub(a_{ik}, a_{jk})}$$

Sum of pairs score

$$\boldsymbol{C^{A} = \sum_{1 \leq i < j \leq n} C_{ij}^{A}}$$

Formal Definition

Shortest Path Problem

Directed acyclic graph $G = (V, E)$

$$V = \{(x_1, \dots, x_n) |\ x_i = 0, \dots, l_i\}$$
$$E = \cup_{e \in \{0,1\}^n} \{(v, v+e) |\ v, v+e \in V, e \neq 0\}.$$

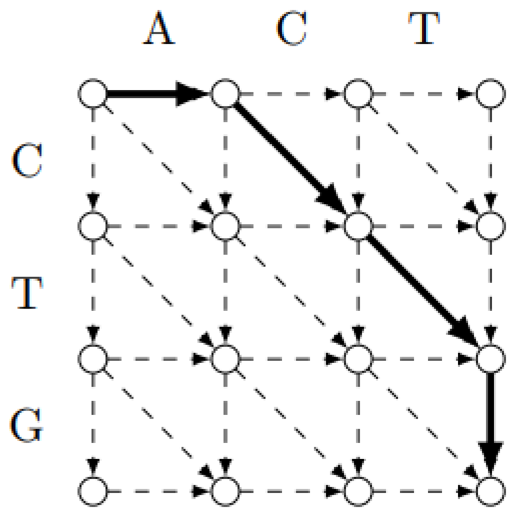


Figure: 2D graph alignment

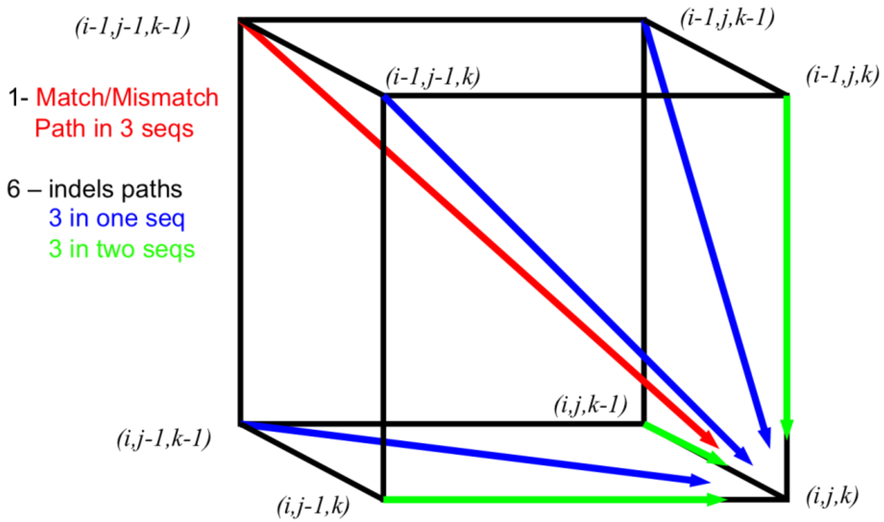


Figure: 3D edge structure

(http://www.csbio.unc.edu/mcmillan/Comp555S16/Lecture14.html)

Agenda.

Solving MSA

Needleman-Wunsch algorithm
Dynamic programming approach

Generates zero-based index table with optimal scores

Dim n, lengths l: Complexity $O(l^n)$

	A	C	T	
C	7	4	6	9
T	8	6	3	6
G	8	6	6	3
	9	6	3	0

Score matrix:

	A	C	T	G	_
A	0	4	2	2	3
C		1	4	3	3
T			0	6	3
G				1	3
_					0

Figure 2: Needleman-Wunsch score table using a score matrix

Solving MSA

Pattern databases

Family of sequences $S = \{s_1, \dots, s_n\}$.

A *pattern* is a subset $P \subseteq S, |P| \geq 2$.

A *pattern database* (PDB) is the perfect heuristic h^*
for the subproblem induced by pattern P.

Solving MSA

Heuristic search estimators

Family of sequences $S = \{s_1, \ldots, s_n\}$.

$\boldsymbol{h_{pair}}$ (Ikeda and Imai, 1994):

$$h_{pair}(v) = \sum_{1 \le i < j \le n} h^{ij}(v)$$

– Uses the information of every 2-dimensional PDB

Solving MSA

Heuristic search estimators

Family of sequences $S = \{s_1, \ldots, s_n\}$.

$\boldsymbol{h_{all,k}}$ (Kobayashi and Imai, 1998):

$$h_{all,k}(v) = \frac{1}{\binom{n-2}{k-2}} \sum_{1 \le x_1 < \cdots < x_k \le n} h^{x_1,\ldots,x_k}(v)$$

- Uses the information of every 3-dimensional PDB
- Every pair of sequences appears $\binom{n-2}{k-2}$ times → normalize
- If $k = 3$, lenghts ~ 500, each PDB contains 10^8 vertices!
- Branching factor $2^n - 1$

Solving MSA

Heuristic search estimators

Family of sequences $S = \{s_1, \dots, s_n\}$.

$\boldsymbol{h_{one,k}}$ (Kobayashi and Imai, 1998):

$$h_{one,k}(v) = h^{x_1,\dots,x_k}(v) + h^{x_{k+1},\dots,x_n}(v) + \sum_{i=1}^{k} \sum_{j=k+1}^{n} h^{x_i,x_j}(v)$$

- 1 or 2 higher-dimensional PDBs + *remaining* 2-dimensional PDBs
- Avoids normalization by choosing PDBs carefully

$$\boldsymbol{h_{pair} \le h_{one,k} \le h_{all,k}}$$

Agenda.

Combining Multiple Pattern Databases

Additivity

- A pattern collection of $S = \{s_1, \dots, s_n\}$ is a collection $P = \{P_1, \dots, P_m\}, \mathrm{P_i} \subseteq S$.
- P is non-conflicting, if no pair of elements of P conflict.
- Then the sum of PDBs is additive

Pattern collection heuristic

$$h^P(v) = \sum_{i=1}^{m} h^{P_i}(v)$$

- Admissible, if P is non-conflicting

Combining Multiple Pattern Databases

- Conflicting pattern collections may violate admissibility
- Parts may still be useful?

Canonical PDB heuristic (Haslum et al., 2007)

$$h^{\mathrm{CAN}}(\mathrm{v}) = \max_{s \in MNS} \sum_{P \in S} h^{P}(v)$$

Combining Multiple Pattern Databases

Post-hoc optimization (Pommerening et al., 2013)

- Use linear programming to solve constrained problem
- Pattern collection is *strictly conflicting* if $|\cap_{i=0}^{m} P_i| > 1$

Let $\langle w_1, \dots, w_m \rangle$ be the solution to the linear program that maximizes

$$h^{PHO}(v) = \sum_{i}^{m} w_i \, h^{P_i}(v)$$

$$s.t. \sum_{i:P_i \in S'} w_i \leq 1 \text{ for all strictly conflicting pattern collections } S' \subseteq P$$

$$s.t. \, 0 \leq w_i \leq 1 \text{ for all } \mathrm{P_i}$$

Combining Multiple Pattern Databases

Post-hoc optimization (Pommerening et al., 2013)

$h_{all,k}$ equals h^{PHO} if we choose the *same patterns*

Proof sketch:

Four sequences $S = \{s_1, s_2, s_3, s_4\}$ of length 1

Score matrix:

	A	C	T	G	–
A	1	1	1	0	1
C		1	1	0	1
T			1	0	1
G				1	1
-					1

$$s_1 = A, s_2 = C, s_3 = T, s_4 = G$$

$$P = \{P_1 = \{s_1, s_2, s_3\}, P_2 = \{s_1, s_2, s_4,\}, P_3 = \{s_1, s_3, s_4,\}, P_4 = \{s_2, s_3, s_4\}\}$$

$$h^{P_1}(s) = 3$$

$$h^{P_2}(s) = h^{P_3}(s) = h^{P_4}(s) = 1$$

→ $h^{PHO}(s) = 1 * 3 + 0 * 1 + 0 * 1 + 0 * 1 = \mathbf{3} = \frac{3+1+1+1}{2} = h_{all,3}$

Combining Multiple Pattern Databases

A factored representation of MSA with operators

$$O = \left\{ o^{i,j}_{\langle x,y \rangle \to \langle x',y' \rangle} \middle| \, 1 \leq i < j \leq n, 0 \leq x \leq l_i, 0 \leq y \leq l_j \right\}$$

An operator $o^{i,j}_{\langle x,y \rangle \to \langle x',y' \rangle}$ *affects* heuristic h^P if $s_i, s_j \in P$

Example:

e.g. edge $\langle 3,3,5 \rangle \to \langle 4,3,6 \rangle$ is factored into 3 operators:

$$\{ o^{1,2}_{\langle 3,3 \rangle \to \langle 4,3 \rangle}, o^{1,3}_{\langle 3,5 \rangle \to \langle 4,6 \rangle}, o^{2,3}_{\langle 3,5 \rangle \to \langle 3,6 \rangle} \}$$

- Basic factors for opreators in higher dimensions
- Less operators than defining all operators

Agenda.

Cost Partitioning

Better estimates than using max. among available?

- Patterns only consider parts of the problem
- Combine multiple heuristic values
- Distribute operator costs among them

Formal (Seipp et al., 2017)

Given a pattern collection of size m
Cost partitioning is a tuple $C = \langle c_1, \ldots, c_m \rangle$ $s.t. \sum_{i=1}^{m} c_i(o) \leq c(o)$
CP heuristic is $h^C(v) \coloneqq \sum_{i=1}^{m} h^{P_i, c_i}(v)$

Cost Partitioning

Greedy zero-one cost partitioning (Haslum et al. 2005; Edelkamp, 2006)

- Assign full costs to at most one PDB
- Multiple PDBs affected? Greedily chose from ordering

- Assign full costs to $\mathrm{c_i}(o)$ if $o \in aff(h^{P_i})$ and $o \notin \cup_{j=1}^{i-1} aff(h^{P_j})$

Cost Partitioning

Saturated cost partitioning (Seipp and Helmert, 2014)

- Assign exploitable parts of the costs to components
- Remainder can contribute to other components

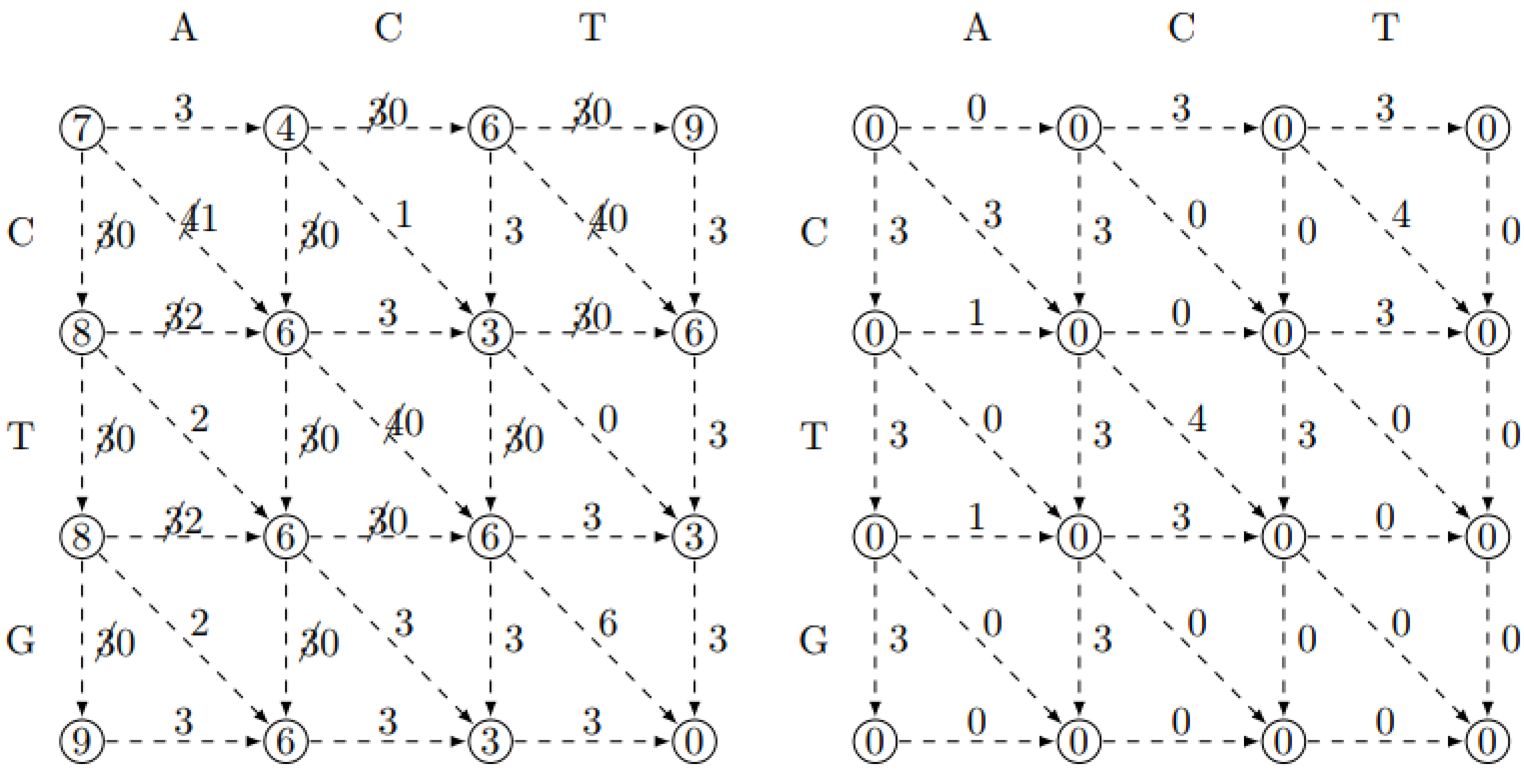
Saturated cost function
Assigns the least possible cost without changing outcome

Formal:
$saturate(h^P, c)$ is the minimal cost function $c' \leq c$ with $h^{P,c'}(v) = h^{P,c}(v)$

Saturated cost partitioning $C = \langle c_1, \ldots, c_m \rangle$
remaining cost functions $\langle \bar{c}_0, \ldots, \bar{c}_m \rangle$

$$\bar{c}_0 = c$$
$$c_i = saturate(h^{P_i}, \bar{c}_{i-1})$$
$$\bar{c}_i = \bar{c}_{i-1} - c_i$$

Cost Partitioning



(a) Saturated operator costs (b) Remaining operator costs

Figure 4.1: Saturated and remaining operator costs for a two-dimensional PDB.

Agenda.

Experiments

- Using MSASolver Java program by Matthew Hatem
- BAliBASE Benchmark Reference Set 1 (Thompson et al., 1999)

1aab_ref1.seq (4 sequences)					
Pattern collection in order ω	$h_{pair}(s)$	$h_{one,3}(s)$	$h_{all,3}(s)$	$h^{PHO}(s)$	$h_{\omega}^{GZOCP}(s)$
$\{0,1\},\{0,2\},\{0,3\},\{1,2\},\{1,3\},\{2,3\}$	14179	-	-	14179	14179
$\{0,1,2\},\{0,3\},\{1,3\},\{2,3\}$	-	14199	-	14199	14199
$\{0,1,2\},\{0,1,3\},\{0,2,3\},\{1,2,3\}$	-	-	14302	14302	14199

Table 5.1: Initial heuristic estimate comparison for the instance 1aab_ref1.seq

Experiments

1aab_ref1.seq (4 sequences)		
#3-fold	Pattern collection	$h^{PHO}(s)$
1	$\{0,1,2\}, \{0,3\}$	9466
	$\{0,1,2\}, \sout{\{0,1\}}, \sout{\{0,2\}}, \sout{\{1,2\}}, \{0,3\}, \{1,3\}, \{2,3\}$	14199
	$\{0,1,2\}, \{0,3\}, \{1,3\}, \{2,3\}$	14199
2	$\{0,1,2\}, \sout{\{0,1,3\}}, \sout{\{0,1\}}$	7186
	$\{0,1,2\}, \sout{\{0,1,3\}}, \{2,3\}$	9466
	$\sout{\{0,1,2\}}, \{0,1,3\}, \{1,2\}, \sout{\{1,3\}}, \{2,3\}$	11849
	$\{0,1,2\}, \sout{\{0,1,3\}}, \{0,3\}, \{1,3\}, \{2,3\}$	14199
3	$\{0,1,2\}, \{0,1,3\}, \{0,2,3\}, \{1,2\}$	11951
	$\{0,1,2\}, \{0,1,3\}, \{0,2,3\}, \{1,2\}, \{1,3\}$	13119
	$\{0,1,2\}, \{0,1,3\}, \{0,2,3\}, \sout{\{0,1\}}, \sout{\{0,2\}}, \sout{\{0,3\}}, \{1,2\}, \{1,3\}, \{2,3\}$	14259
	$\{0,1,2\}, \{0,1,3\}, \{0,2,3\}, \{1,2\}, \{1,3\}, \{2,3\}$	14259
4	$\{0,1,2\}, \{0,1,3\}, \{0,2,3\}, \{1,2,3\}$	14302

Table 5.2: Initial heuristic estimates of h^{PHO} for the instance 1aab_ref1.seq. Crossed out patterns are assigned weights of 0 by the linear program

Agenda.

Conclusion

GZOCP

- No benefit over existing heuristics

PHO

- LP Solver in every search step
- Expensively computed PDB may be left unused

Future Work

- Implement other cost partitioning techniques
- Generate PDBs automatically e.g. like Haslum et al. (2007)
- M&S heuristics (Dräger et al., 2009; Helmert et al., 2014)

Universität
Basel

Thank you
for your attention.

Bibliography

MO Dayhoff, RM Schwartz, and BC Orcutt. A model of evolutionary change in proteins. In Atlas of protein sequence and structure, volume 5, pages 345–352. National Biomedical Research Foundation Silver Spring, MD, 1978.

Steven Henikoff and Jorja G Henikoff. Amino acid substitution matrices from protein blocks. Proceedings of the National Academy of Sciences, 89:10915–10919, 1992.

Saul B Needleman and Christian D Wunsch. A general method applicable to the search for similarities in the amino acid sequence of two proteins. Journal of molecular biology, 48 (3):443–453, 1970.

Takahiro Ikeda and Hiroshi Imai. Fast a algorithms for multiple sequence alignment. Genome Informatics, 5:90–99, 1994.

Hirotada Kobayashi and Hiroshi Imai. Improvement of the A* algorithm for multiple sequence alignment. Genome Informatics, 9:120–130, 1998.

Patrik Haslum, Adi Botea, Malte Helmert, Blai Bonet, Sven Koenig, et al. Domainindependent construction of pattern database heuristics for cost-optimal planning. In AAAI, volume 7, pages 1007–1012, 2007.

Florian Pommerening, Gabriele Röger, and Malte Helmert. Getting the most out of pattern databases for classical planning. In Francesca Rossi, editor, Proceedings of the 23rd International Joint Conference on Artificial Intelligence (IJCAI 2013), pages 2357–2364, 2013.

Jendrik Seipp, Thomas Keller, and Malte Helmert. A comparison of cost partitioning algorithms for optimal classical planning. In Proceedings of the Twenty-Seventh International Conference on Automated Planning and Scheduling (ICAPS 2017). AAAI Press, 2017.

Patrik Haslum, Blai Bonet, Héctor Geffner, et al. New admissible heuristics for domainindependent planning. In AAAI, volume 5, pages 9–13, 2005.

Bibliography

Stefan Edelkamp. Automated creation of pattern database search heuristics. In International Workshop on Model Checking and Artificial Intelligence, pages 35–50. Springer, 2006.

Jendrik Seipp and Malte Helmert. Diverse and additive cartesian abstraction heuristics. In Proceedings of the Twenty-Fourth International Conference on Automated Planning and Scheduling (ICAPS 2014). AAAI Press, pages 289–297, 2014.

Julie D. Thompson, Frédéric Plewniak, and Olivier Poch. Balibase: a benchmark alignment database for the evaluation of multiple alignment programs. Bioinformatics (Oxford, England), 15:87–88, 1999.

Klaus Dräger, Bernd Finkbeiner, and Andreas Podelski. Directed model checking with distance-preserving abstractions. International Journal on Software Tools for Technology Transfer, 11(1):27–37, 2009.

Malte Helmert, Patrik Haslum, Jörg Hoffmann, and Raz Nissim. Merge-and-shrink abstraction: A method for generating lower bounds in factored state spaces. Journal of the ACM (JACM), 61(3):16, 2014.