Sparse RNA folding revisited: space-efficient minimum free energy prediction

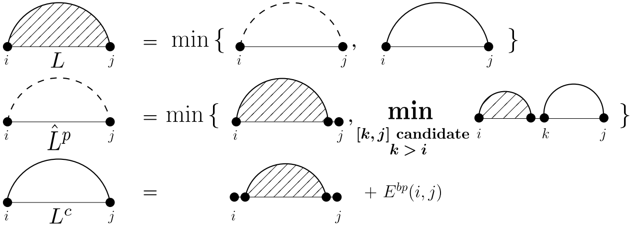
Sebastian Will

Benasque 2015

Bioinformatics, University Leipzig

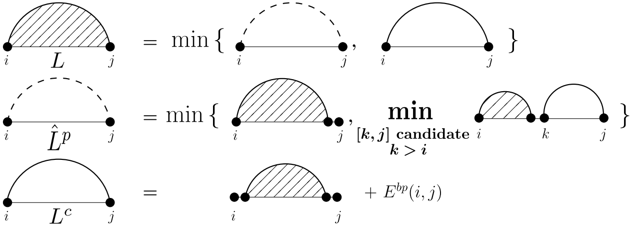
with Hosna Jabbari. To appear at WABI 2015.

Sparsified base pair-based prediction

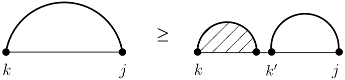

L
$= \min\{$, $\}$
$\hat{L}^p$
$= \min\{$, $\min_{\substack{[k,j] \text{ candidate} \\ k>i}}$ $\}$
L^c
$=$ $+ E^{bp}(i,j)$
i
j
k

Backofen et al. JDA 2011

Sparsified base pair-based prediction

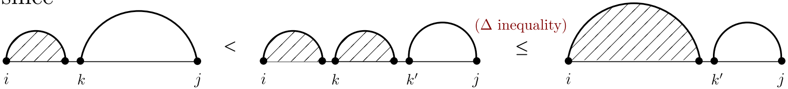


in min, no need to consider k if



otherwise *candidate*

since



Complexity $O(n^2 + n \cdot Z_L)$ time; $\Theta(n + Z_L)$ space

(Z_L = total # of candidates)

Backofen et al. JDA 2011

Minimum free energy prediction

Original recursions

[Zuker & Sankoff, 1984; i.e. with ML penalties; notation adapted]

$$\begin{aligned}
W(i,j) &= \min\{\, V(i,j), \min_{i<k<j} W(i,k) + W(k+1,j)\,\} \\
V(i,j) &= \min\{\, \mathcal{H}(i,j), \min_{\substack{i<p<q<j \\ p-i+j-q-2\leq M}} \mathcal{I}(i,j,p,q) + V(p,q), \\
&\qquad \min_{i<k<j} WM(i+1,k) + WM(k+1,j-1) + a\} \\
WM(i,j) &= \min\{\, V(i,j) + b, WM(i+1,j) + c, WM(i,j-1) + c, \\
&\qquad \min_{i<k<j} WM(i,k) + WM(k+1,j)\,\}
\end{aligned}$$

Note: previous work [Wexler et al., Backofen et al.] sparsified only [Zuker&Stiegler, 1981]; no space-efficient trace back

Rewrite to prepare sparsification . . .

$$W(i,j) = \min\{\, W^p(i,j), V(i,j) \,\}$$

$$W^p(i,j) = \min\{\, W(i,j-1), \min_{i<k<j} W(i,k-1) + W(k,j) \,\}$$

$$V(i,j) = \min\{\mathcal{H}(i,j), \min_{\substack{i<p<q<j \\ p-i+j-q-2\leq M}} \mathcal{I}(i,j,p,q) + V(p,q), WM^2(i+1,j-1) + a\}$$

$$WM(i,j) = \min\{\, WM^p(i,j), V(i,j) + b \,\}$$

$$WM^p(i,j) = \min\{\, WM(i+1,j) + c, WM(i,j-1) + c, WM^2(i,j) \,\}$$

$$WM^2(i,j) = \min_{i<k<j} WM(i,k-1) + WM(k,j)$$

. . . and sparsify: minimize only over candidates

$$\widehat{W}^{\mathrm{p}}(i,j) = \min\{\, W(i,j-1), \min_{\substack{[k,j]\ \text{W-candidate,} \\ k>i}} W(i,k-1) + V(k,j) \,\}$$

$$\widehat{WM}^2(i,j) = \min\{\, WM^2(i,j-1) + c, \min_{\substack{[k,j]\ \text{WM-candidate,} \\ k>i}} WM(i,k-1) + V(k,j) + b \,\}$$

candidate criteria:

- $[k,j]$ is a *W-candidate* iff $V(k,j) < \widehat{W}^{\mathrm{p}}(k,j)$ and
- $[k,j]$ is a *WM-candidate* iff $V(k,j) + b < WM^p(k,j)$.

Rewrite to prepare sparsification . . .

$$
\begin{aligned}
W(i,j) &= \min\{\, W^p(i,j), V(i,j) \,\} \\
W^p(i,j) &= \min\{\, W(i,j-1), \min_{i<k<j} W(i,k-1) + W(k,j) \,\} \\
V(i,j) &= \min\{\mathcal{H}(i,j), \min_{\substack{i<p<q<j \\ p-i+j-q-2\leq M}} \mathcal{I}(i,j,p,q) + V(p,q), WM^2(i+1,j-1) + a\} \\
WM(i,j) &= \min\{\, WM^p(i,j), V(i,j) + b \,\} \\
WM^p(i,j) &= \min\{\, WM(i+1,j) + c, WM(i,j-1) + c, WM^2(i,j) \,\} \\
WM^2(i,j) &= \min_{i<k<j} WM(i,k-1) + WM(k,j)
\end{aligned}
$$

. . . and sparsify: minimize only over candidates

$$
\begin{aligned}
\widehat{W}^{\mathrm{p}}(i,j) &= \min\{\, W(i,j-1), \min_{\substack{[k,j] \text{ W-candidate}, \\ k>i}} W(i,k-1) + V(k,j) \,\} \\
\widehat{WM}^2(i,j) &= \min\{\, WM^2(i,j-1) + c, \min_{\substack{[k,j] \text{ WM-candidate}, \\ k>i}} WM(i,k-1) + V(k,j) + b \,\}
\end{aligned}
$$

candidate criteria:

- $[k,j]$ is a *W-candidate* iff $V(k,j) < \widehat{W}^{\mathrm{p}}(k,j)$ and
- $[k,j]$ is a *WM-candidate* iff $V(k,j) + b < WM^p(k,j)$.

Rewrite to prepare sparsification . . .

$$W(i,j) = \min\{\, W^p(i,j), V(i,j) \,\}$$

$$W^p(i,j) = \min\{\, W(i,j-1), \min_{i<k<j} W(i,k-1) + W(k,j) \,\}$$

$$V(i,j) = \min\{\mathcal{H}(i,j), \min_{\substack{i<p<q<j \\ p-i+j-q-2\leq M}} \mathcal{I}(i,j,p,q) + V(p,q), WM^2(i+1,j-1) + a\}$$

$$WM(i,j) = \min\{\, WM^p(i,j), V(i,j) + b \,\}$$

$$WM^p(i,j) = \min\{\, WM(i+1,j) + c, WM(i,j-1) + c, WM^2(i,j) \,\}$$

$$WM^2(i,j) = \min_{i<k<j} WM(i,k-1) + WM(k,j)$$

. . . and sparsify: minimize only over candidates

$$\widehat{W}^{\mathrm{p}}(i,j) = \min\{\, W(i,j-1), \min_{\substack{[k,j]\ \text{W-candidate,} \\ k>i}} W(i,k-1) + V(k,j) \,\}$$

$$\widehat{WM}^2(i,j) = \min\{\, WM^2(i,j-1) + c, \min_{\substack{[k,j]\ \text{WM-candidate,} \\ k>i}} WM(i,k-1) + V(k,j) + b \,\}$$

candidate criteria:

- $[k,j]$ is a *W-candidate* iff $V(k,j) < \widehat{W}^{\mathrm{p}}(k,j)$ and
- $[k,j]$ is a *WM-candidate* iff $V(k,j) + b < WM^p(k,j)$.

Space-efficient bp-based prediction: Trace back

Sparse TB in base pair-based model:

Problem: forward evaluation stores only candidates

Solution (Backofen et al., JDA11):
recompute row-by-row for $i = 1$ to n
recomputation never needs non-candidates in rows $i' > i$, **since** closed substructures are candidates!

Not transferable to (full) MFE prediction!

- trace back (recomputation) of interior loops needs access to non-candidates in rows $i' > i$
- inner base pairs are not necessarily candidates

Example:

```
GCCAAAAGGGC
(((.....)))
```

Space-efficient bp-based prediction: Trace back

Sparse TB in base pair-based model:

Problem: forward evaluation stores only candidates

Solution (Backofen et al., JDA11):
recompute row-by-row for $i = 1$ to n
recomputation never needs non-candidates in rows $i' > i$, **since** closed substructures are candidates!

Not transferable to (full) MFE prediction!

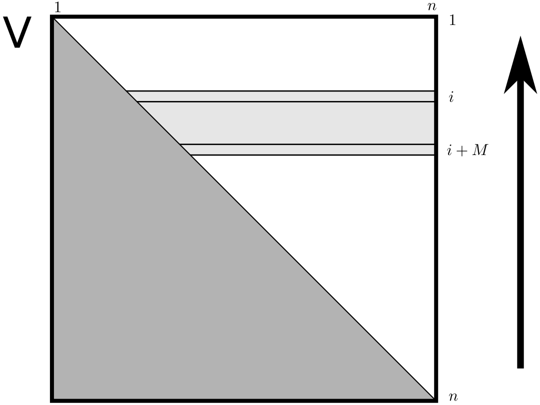
- trace back (recomputation) of interior loops needs access to non-candidates in rows $i' > i$
- inner base pairs are not necessarily candidates

Example:

```
GCCAAAAGGGC
(((.....)))
```

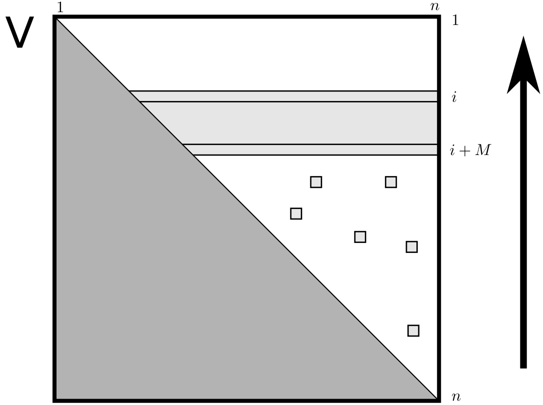
Sparse space-efficient MFE trace back

Problem motivation



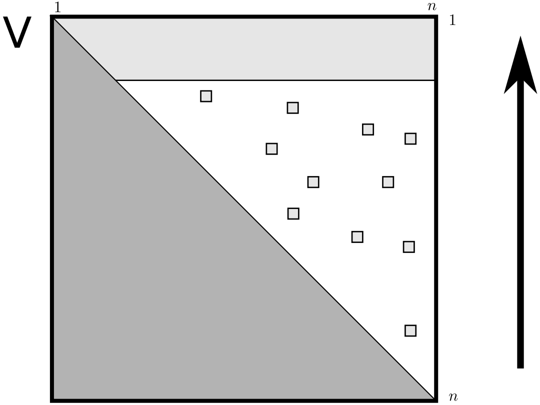
Sparse space-efficient MFE trace back

Problem motivation



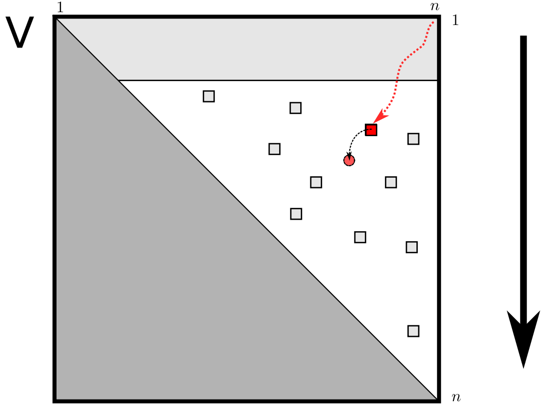
Sparse space-efficient MFE trace back

Problem motivation

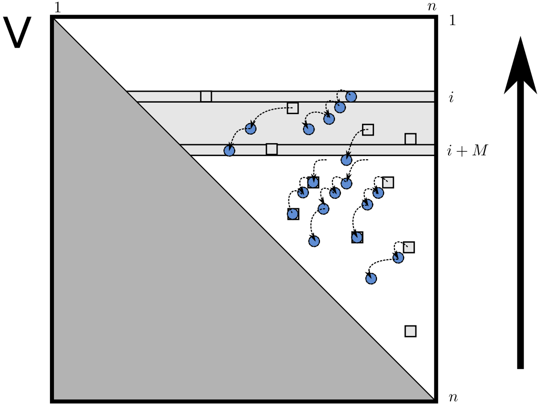


Sparse space-efficient MFE trace back

Problem motivation



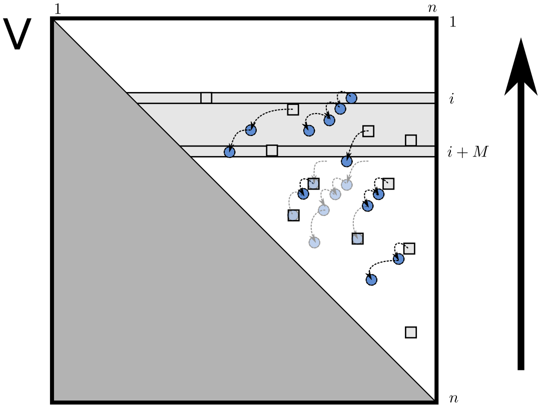
Sparse space-efficient MFE trace back



Naïve solution: store all trace arrows . . .

. . . but too many TAs; compromises “space-efficient”

Sparse space-efficient MFE trace back



Idea: avoid storing many TAs & garbage collect

- avoid TAs in case $WM(i+1,j)+c$ of WM^p (rewrite recursions)
- avoid TAs to candidates (since we can recompute)
- garbage collect: keep only accessible TAs

Results

Theory: $O(n^2 + nZ)$ time; $\Theta(n + Z + T)$ space
Z = total # of *candidates*; T = maximum # of accessible TAs.

Note: $T + Z < n^2$ (idea "<<")

Practice: Free C++ implementation SparseMFEFold

- interface to Vienna RNA lib 2.x [Lorenz et al., 2011]
- predictions identical to `RNAfold -d0`

SparseMFEFold is available at
`www.bioinf.uni-leipzig.de/~will/Software/SparseMFEFold`

Results

Theory: $O(n^2 + nZ)$ time; $\Theta(n + Z + T)$ space
Z = total # of *candidates*; T = maximum # of accessible TAs.

Note: $T + Z < n^2$ (idea "<<")

Practice: Free C++ implementation SparseMFEFold

- interface to Vienna RNA lib 2.x [Lorenz et al., 2011]
- predictions identical to `RNAfold -d0`

SparseMFEFold is available at

`www.bioinf.uni-leipzig.de/~will/Software/SparseMFEFold`

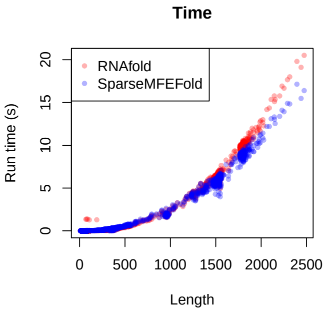
Empirical results

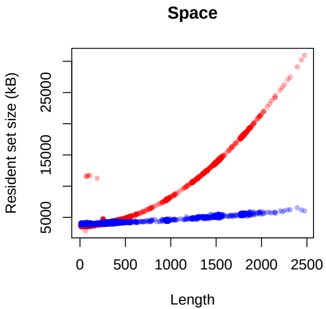
Benchmark: RNA STRAND 2.0

Performance of SparseMFEFold vs. RNAfold (length $\geq$ 2500)

	Run time (s)		**Memory: resident set size (kB)**	
	RNAfold	SparseMFEFold	RNAfold	SparseMFEFold
Minimum	16.9	15.4	31800	5932 (19%)
Median	29.7	22.9	42828	7262 (17%)
Maximum	89.9	57.4	88548	9048 (10%)

length $\leq$ 2500:

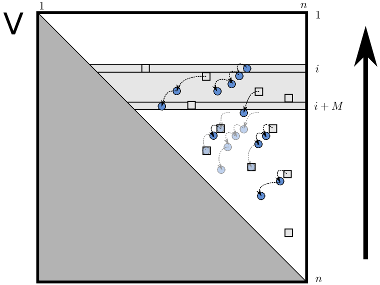




Empirical results: TA savings

Benchmark: RNA STRAND 2.0 (length $\geq$ 2500)

	Number of candidates	**Number of trace arrows**		
		Maximum	Avoided	GC-Removed
Minimum	17,032	52,293	137,892	467,230
Median	41,215	94,443	237,717	706,365
Maximum	71,508	148,947	419,825	1,748,491



Perspectives

Techniques are generalizable

Promising applications:

Traceback of *highly complex* structure prediction

- MFE Pseudoknot prediction [Rivas, Eddy]
  - $O(n^4)$ space
  - [Moehl et al., 2011]: sparse evaluation, not space-efficient
- MFE PK-prediction "CCJ" [Chen, Condon, Jabbari]
  - $O(n^4)$ space
  - work in progress with Hosna Jabbari
  - motivation of this work
- MFE RNA-RNA-interaction prediction [Alkan et al.]
  - $O(n^4)$ space
  - [Salari et al., 2010]: space-efficient evaluation, but no space-efficient TB
- Simultaneous Folding and Alignment
  - $O(n^4)$ space [Sankoff, 1985]
  - $O(n^2)$ space [LocARNA, 2007], [SPARSE, 2015]

Conclusions

- Sparsification can strongly reduce memory demands (constant # of rows + candidates)
- Traceback of MFE prediction needs additional information (TAs)
- The novel approach keeps additional memory requirements low
- Techniques (rewriting, partial recomputation, and GC) generalize
- Promising: Apply to highly complex prediction algorithms

Thanks to . . .

- Hosna Jabbari & Anne Condon
- the organizers of the Benasque RNA meeting
- you, for your attention

Conclusions

- Sparsification can strongly reduce memory demands (constant # of rows + candidates)
- Traceback of MFE prediction needs additional information (TAs)
- The novel approach keeps additional memory requirements low
- Techniques (rewriting, partial recomputation, and GC) generalize
- Promising: Apply to highly complex prediction algorithms

Thanks to . . .

- Hosna Jabbari & Anne Condon
- the organizers of the Benasque RNA meeting
- you, for your attention