On Stabbing Queries for Generalized Longest Repeats

Bojian Xu
Eastern Washington University

IEEE International Conference on Bioinformatics and Biomedicine (BIBM)
Washington D.C., Nov 9-12, 2015
Longest Repeat (LR) Query

• Point query *(prior work)*

```
  a b a b b       a b a b b
  b a b a b a
```

• Interval query *(this work)*

```
  a b a b       b a b a b
  b a b a b a
```

(Point query becomes a special case of interval query)
Why Find Repeats?

• Important role in the biological functions of genomes and proteins.
  • About 1/3 of the whole human genome consists of repeated substrings
  • About 10–25% of all known proteins have some form of repetitive structures.

• A number of significant problems in molecular string analysis can be reduced to repeat finding.

• LR query provides a “stabbing” tool for finding most of the repeats that cover any particular string position interval.
New Challenges in Interval Queries

• Point query
  • Only $n$ string positions $\rightarrow$ only $n$ point queries.
  • Spend $O(n)$ space and time to compute for all $n$ LR point queries.

• Interval query
  • $\Theta(n^2)$ different intervals
  • Given $O(n)$ space and time budget, old strategy for point queries fails
Our Contributions

• Extended the point LR query to interval LR query

• Proposed an indexing structure, such that
  • It can be constructed in $O(n)$ time and space, where $n$ is the string size.
  • Any interval LR query can be answered in $O(1)$ time by using the index.
Left-bounded Longest Repeats (LLR)

- **Useless LLR**: those blue ones, who are substrings of the red ones.
- **Useful LLR**: those red ones.

- LLRs are easy to compute using $O(n)$ space and time (find details in the paper).

Any LR must be a *useful* LLR

Find LR $\rightarrow$ Find the longest LLR that covers the given interval
A Geometric Perspective of Useful LLRs

- Start positions (x) of useful LLRs strictly increase, so are their ending (y) positions.
- View each LLR as a dot in the x-y 2d space and its length as its weight.

Inspired by the insight from Hu et al., SPIRE 2014
LR Query ➔ Dominance Max Query (DMQ)
LR Query → Range Max Query (RMQ)

- \( L_y \) = the index of the lowest dot that is above \( y \) (inclusive)
- \( R_x \) = the index of the rightmost dot that is on the left of \( x \) (inclusive)
- Given the array of useful LLRs, Array L and R can be computed in \( O(n) \) time.

Given interval \([x, y]\), do RMQ over useful LLR array over range \([L_y, R_x]\).
The Complete Picture of Two Proposals

String

Construct Suffix array and longest common prefix array.

$O(n)$ space and time (existing technique)

Compute all useful LLRs

$O(n)$ space and time (find details in paper)

Construct DMQ index

$O(n)$ space and $O(n \log n)$ time

Construct RMQ index

$O(n)$ space and time

Each future query takes $O(occ \log n)$ time

Each future query takes $O(occ)$ time

$occ = \#\text{LRs of interest found in the string.}$
Experiments – Peak memory usage

![Graph showing peak memory usage](image-url)

- DNA
- [12]
- [14]
- RMQ-based
- DMQ-based

Y-axis: Peak Memory Usage in MBs
X-axis: Sequence Size in MBs
Experiments – Index construction time cost

![Graph showing the time cost of index construction for different sequence sizes. The graph includes lines for SA, Rank, LCP, RMQ, and DMQ, with markers indicating the time cost at various sequence sizes.](graph.png)
Experiments – Query time cost (1)

DNA: Find one choice for each LR

![Graph showing query time cost](image)

- RMQ, interval size = 1
- RMQ, interval size = 5
- RMQ, interval size = 10
- RMQ, interval size = 15
- RMQ, interval size = 20

Time Cost in Seconds vs. Sequence Size in MBs
Experiments – Query time cost (2)

DNA; Find all choices for each LR

- RMQ, interval size = 1
- RMQ, interval size = 5
- RMQ, interval size = 10
- RMQ, interval size = 15
- RMQ, interval size = 20

Time Cost in Seconds

Sequence Size in MBs
Similar observations for protein data. (Find details in the paper)
Question?

Thanks!