Gapped String Indexing in Subquadratic Space and Sublinear Query Time

Philip Bille
Inge Li Gørtz
Moshe Lewenstein
Solon P. Pissis
Eva Rotenberg
Teresa Anna Steiner

String Indexing

- Preprocess string S of length n.
- Query(P): return all occurrences of P within S.

1	2	3	4	5	6	7	8	9	10	11	12	13	14
n	a	n	a	n	a	n	a	b	a	t	m	a	n

P = NA

Query(P): 1, 3, 5, 7

Gapped String Indexing

- Preprocess string S of length n.
- Query(P_1, P_2, α, β): return all occurrences of P_1 and P_2 in S whose distance is in [α, β].

1	2	3	4	5	6	7	8	9	10	11	12	13	14
n	a	n	a	n	a	n	a	b	a	t	m	a	n

P_1 = NA

P_2 = AN

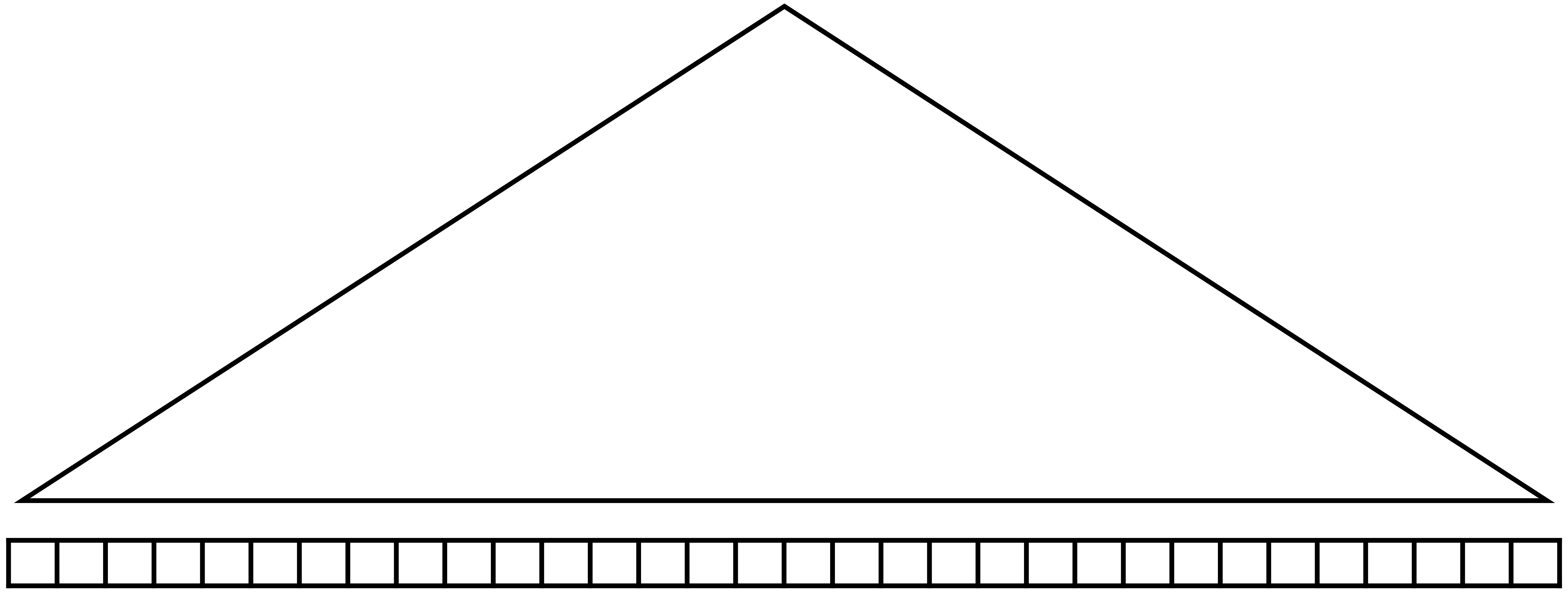
α, β = 3,6

Query(P_1, P_2, α, β): (1,4), (1,6), (3,6), (7,13)

Simple Solutions

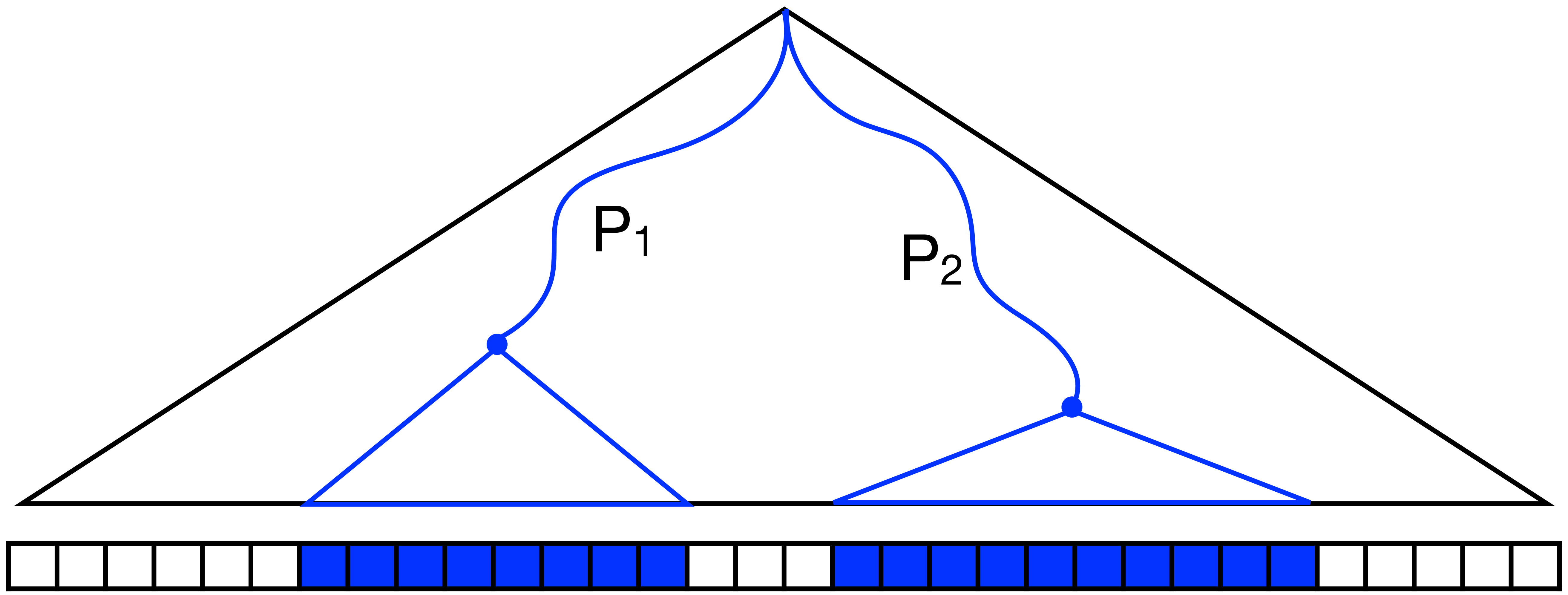
- Ignore reporting occurrences. Just support existence.
- Assume point range [α, α].

Set Intersection



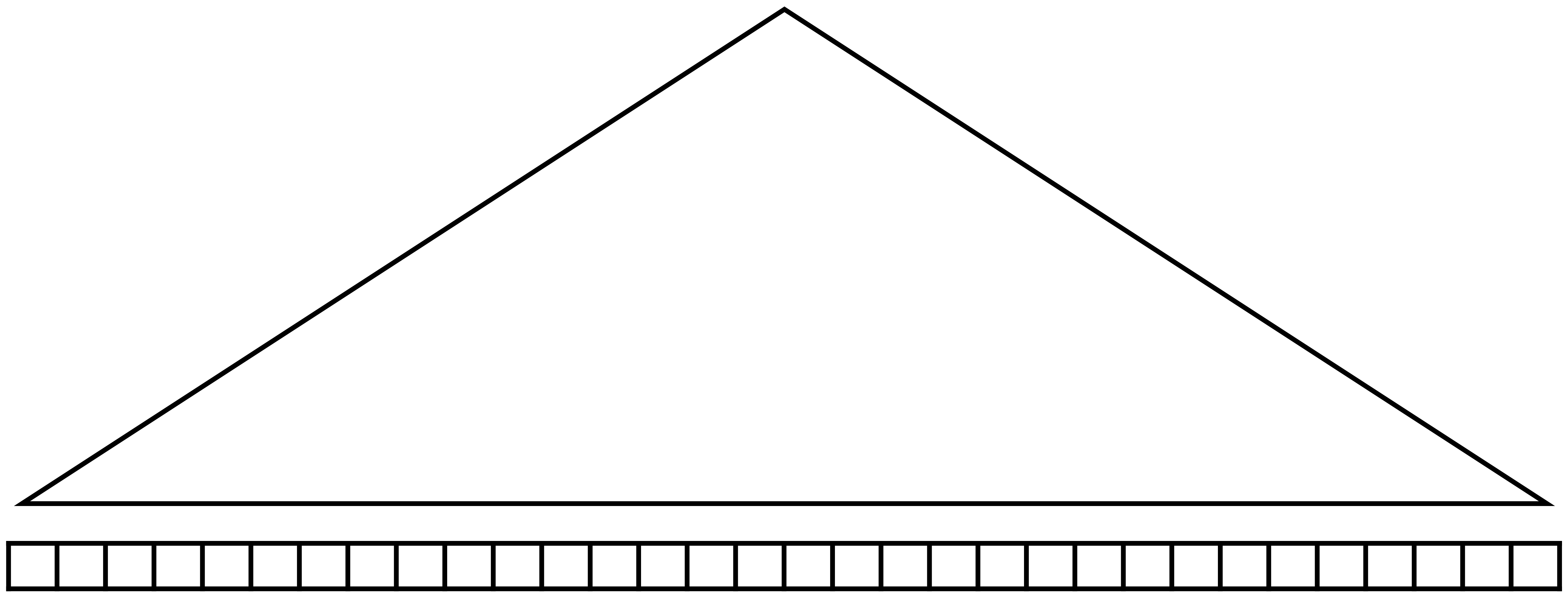
- Data structure.
  - Suffix tree.
  - Suffix array.
- ⇒ O(n) space.

Set Intersection



- Query(P_1, P_2, α).
  - Identify suffix array ranges of P_1 and P_2.
  - Scan suffix array ranges in sorted order.
  - Merge with respect to α.
- $\Rightarrow O(|P_1| + |P_2| + n) = O(n)$ time.

Tabulation

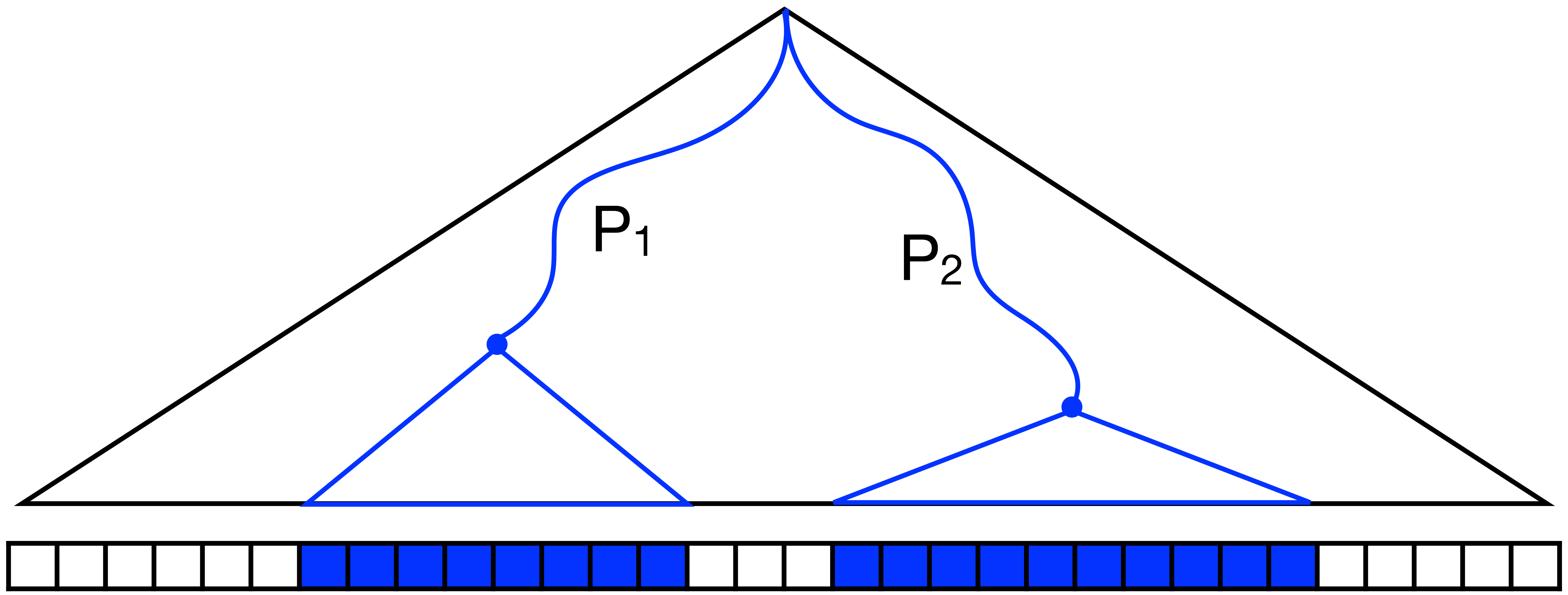


v_1	v_2	0	no
v_1	v_2	1	yes
v_1	v_3	2	no

⋮

- Data structure.
  - Suffix tree.
  - Suffix array
  - Table with answers to queries for all pairs of nodes and gaps.
- ⇒ $O(n^3)$ space.

Tabulation

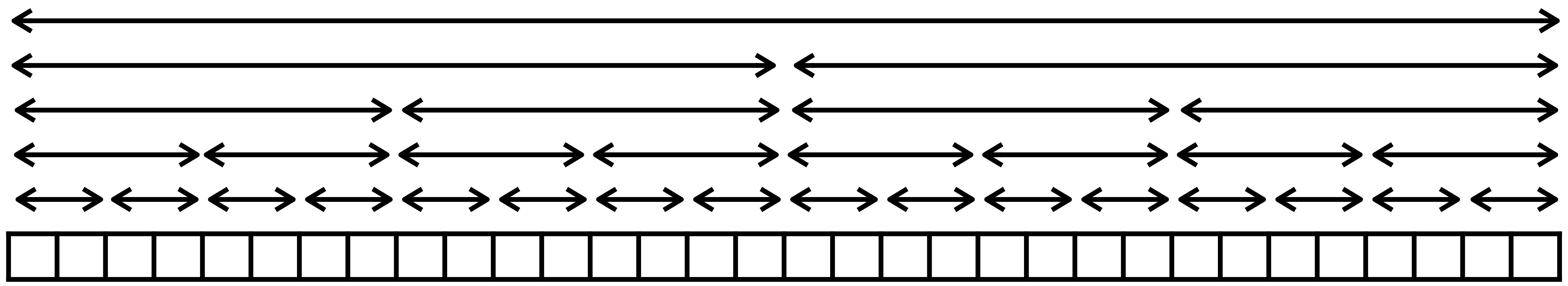


v_1	v_2	0	no
v_1	v_2	1	yes
v_1	v_3	2	no

⋮

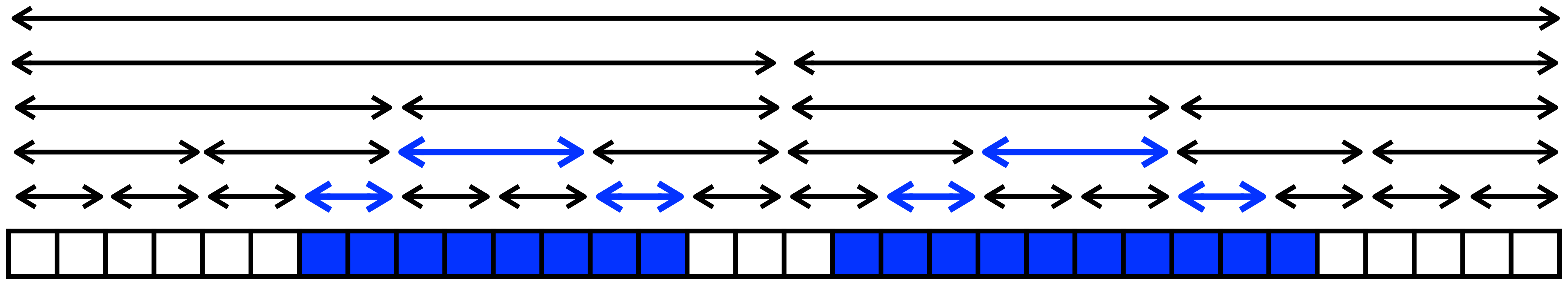
- Query(P_1, P_2, α).
  - Identify nodes for P_1 and P_2.
  - Lookup in table.
- $\Rightarrow O(|P_1| + |P_2|)$ time.

Improved Tabulation



- Data structure.
  - Build sets for the dyadic intervals of the suffix array.
  - For each pair of dyadic intervals store all pairwise distances.
- Total size of set for the dyadic intervals is $O(n \log n) \Rightarrow O((n \log n)^2) = \tilde{O}(n^2)$ space.

Improved Tabulation



- Query(P_1, P_2, α).
  - Cover suffix array ranges with O(log n) dyadic intervals.
  - Query all O($\log^2$ n) pairs.
- $\implies$ O($|P_1| + |P_2| + \log^2 n$) time.

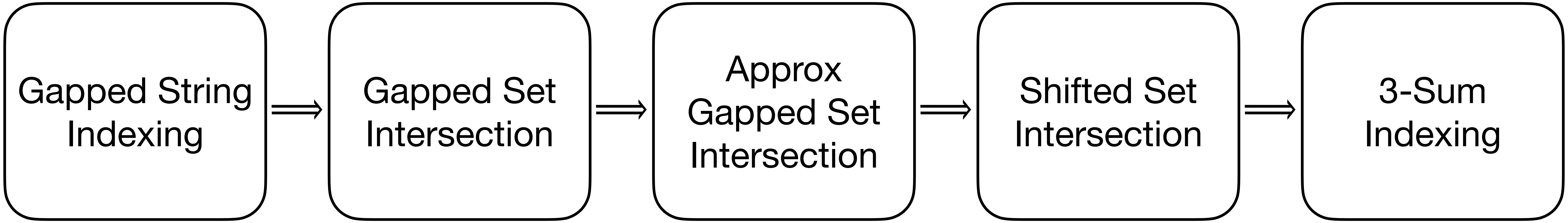
	Space	Time
Set Intersection	$O(n)$	$O(n)$
Tabulation	$\tilde{O}(n^2)$	$\tilde{O}(\lvert P_1\rvert + \lvert P_2\rvert)$
New	$\tilde{O}(n^{2-\delta/3})$	$\tilde{O}(\lvert P_1\rvert + \lvert P_2\rvert + n^{\delta})$

Question: Can we get subquadratic space **and** sublinear query time?

3-Sum Indexing

- Preprocess sets A and B of size n.
- Query(z): decide if there is $a \in A$ and $b \in B$ such that $a + b = z$.

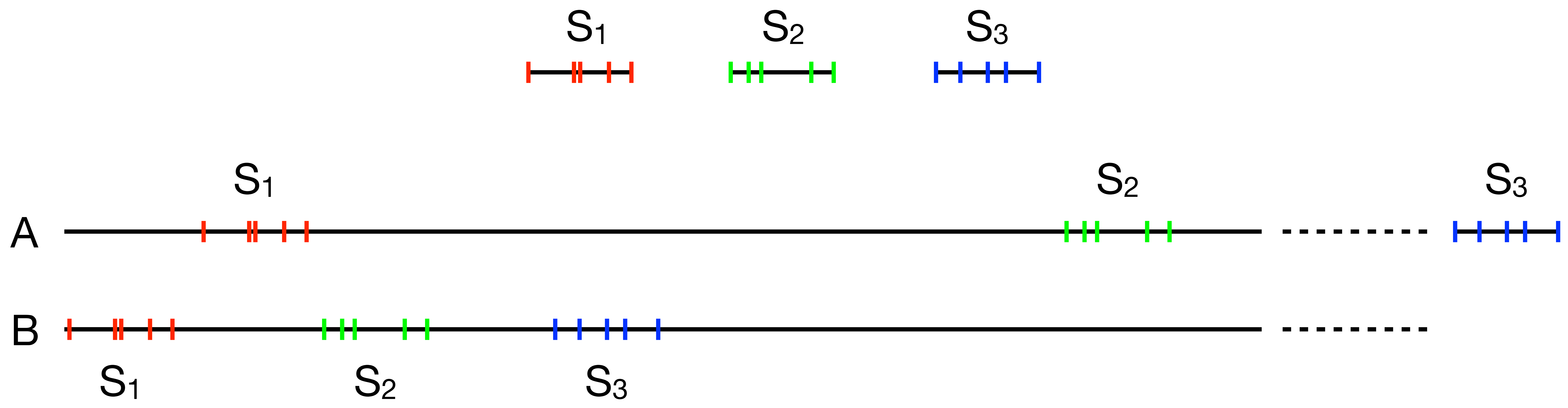
- Theorem [Golovnev et al., STOC 2020]
  - 3-sum indexing with $\tilde{O}(n^{2-\delta/3})$ space and $\tilde{O}(n^{\delta})$ query time.


Gapped String Indexing
Gapped Set Intersection
Approx Gapped Set Intersection
Shifted Set Intersection
3-Sum Indexing

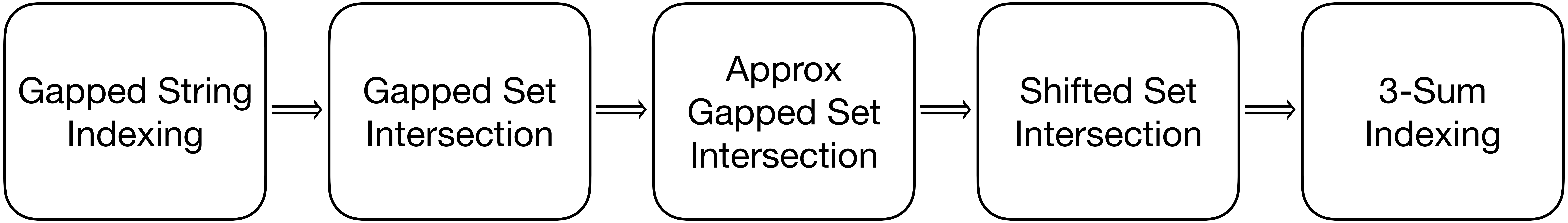
Shifted Set Intersection

- Preprocess sets $S_1, S_2, \ldots, S_k$ of total size n.
- Query(i, j, d): decide if there is $x \in S_i$ and $y \in S_j$ such that $y - x = d$.

Shifted Set Intersection ⟹ 3-Sum Indexing

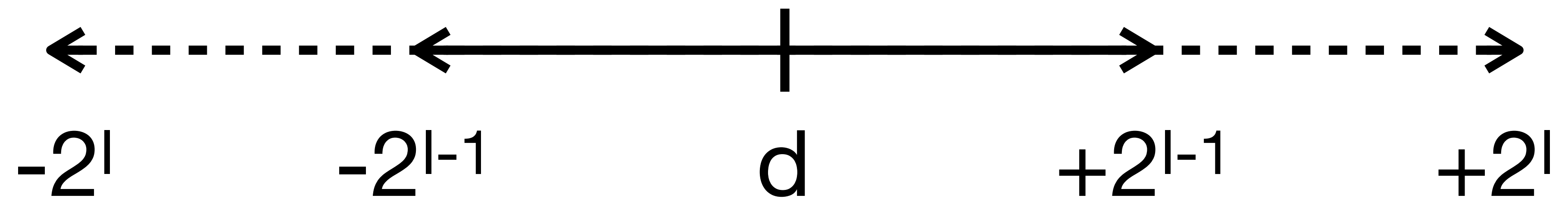


- Reduction.
  - Layout sets $S_1, S_2, \ldots, S_k$ in A and B to avoid the same differences.
  - Scale shifted set intersection query according to i and j ⟹ 3-sum indexing query.


Gapped String Indexing
Gapped Set Intersection
Approx Gapped Set Intersection
Shifted Set Intersection
3-Sum Indexing

Approximate Gapped Set Intersection

- Preprocess sets $S_1, S_2, \ldots, S_k$ of total size n.
- Query(i, j, d, l):
  - Yes: if there is $x \in S_i$ and $y \in S_j$ such that $y - x = d \pm 2^{l-1}$.
  - No: if there is no $x \in S_i$ and $y \in S_j$ such that $y - x = d \pm 2^{l}$



Approximate Gapped Set Intersection $\Longrightarrow$ Shifted Set Intersection

$$S_i = \{1, 2, 4, 5, 8, 12, 13, 17, \ldots\}$$

$$\widetilde{S}_i^1 = \{0, 1, 2, 4, 6, 8, \ldots\}$$

$$\widetilde{S}_i^2 = \{0, 1, 2, 3, 4, \ldots\}$$

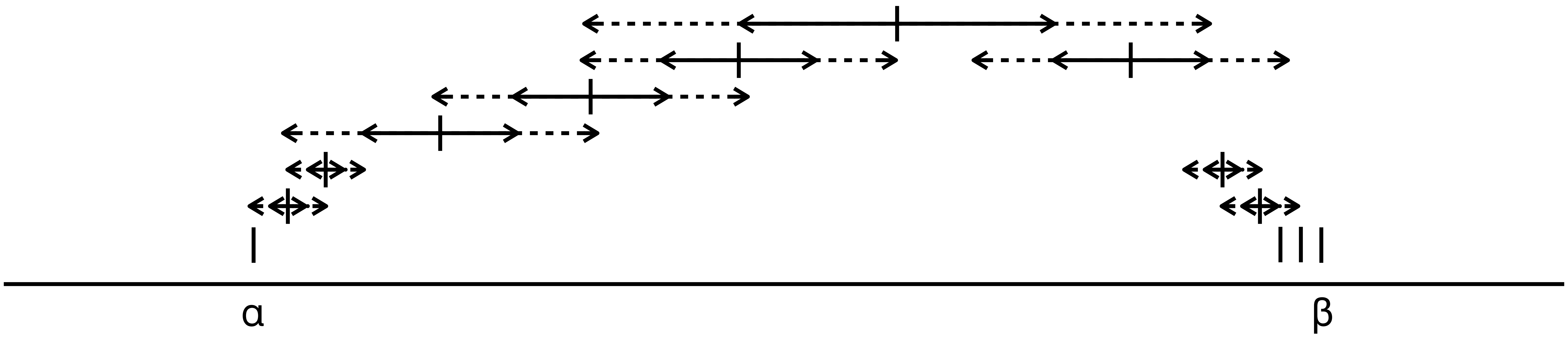
$$\vdots$$

- Reduction.
  - For each set S_i, construct "approximate" sets by dividing by powers of two and rounding down.
  - Approximate gapped set intersection query $\Longrightarrow$ O(1) shifted set intersection queries.

Gapped Set Intersection

- Preprocess sets $S_1, S_2, \ldots, S_k$ of total size n.
- Query(i, j, [α, β]): decide if there is $x \in S_i$ and $y \in S_j$ such that $y - x \in [\alpha, \beta]$.

Gapped Set Intersection ⟹ Approximate Gapped Set Intersection

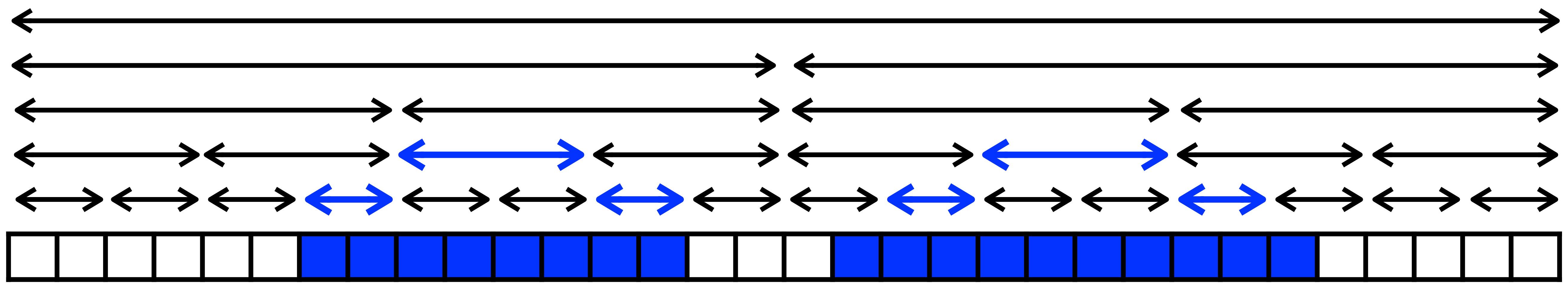


- Reduction.
  - Store approximate gapped set intersection structure.
  - Gapped set intersection query by covering [α, β] interval with O(log n) approximate gapped set intersection queries.

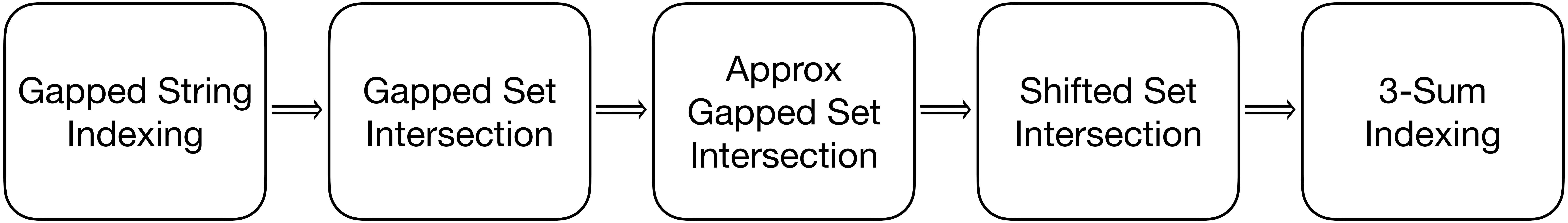
Gapped String Indexing

- Preprocess string S of length n.
- Query(P_1, P_2, α, β): decide if there is occurrence of P_1 and P_2 in S whose distance is in [α, β].

Gapped String Indexing ⟹ Gapped Set Intersection



- Reduction.
  - Store gapped set intersection structure for dyadic intervals of suffix array.
  - Gapped string indexing query ⟹ gapped set intersection on covering intervals.



Gapped String Indexing
Gapped Set Intersection
Approx Gapped Set Intersection
Shifted Set Intersection
3-Sum Indexing

Gapped String Indexing

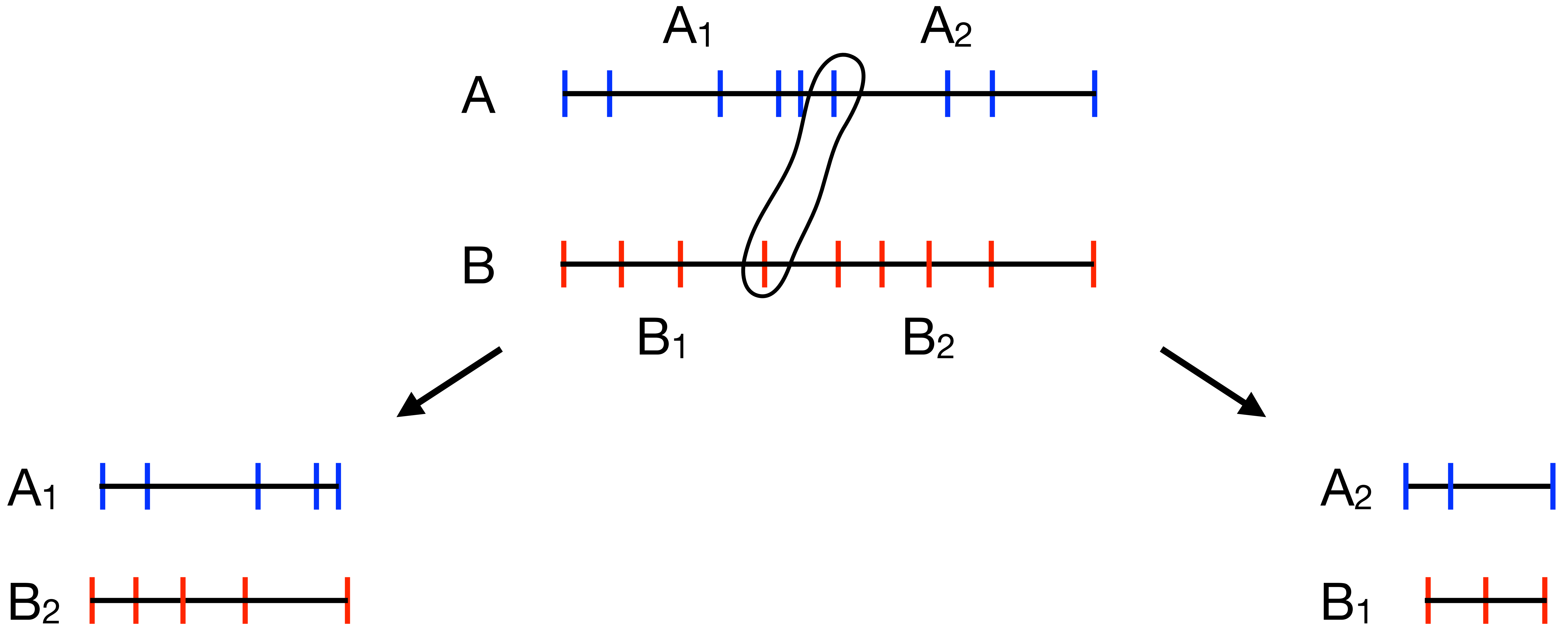
- Theorem.
  - Gapped string indexing with $\tilde{O}(n^{2-\delta/3})$ space and $\tilde{O}(n^{\delta})$ query time.

- What about reporting?

3-Sum Indexing with Reporting

- Preprocess sets A and B of size n.
- Query(z): report all pairs $a \in A$ and $b \in B$ such that $a + b = z$.

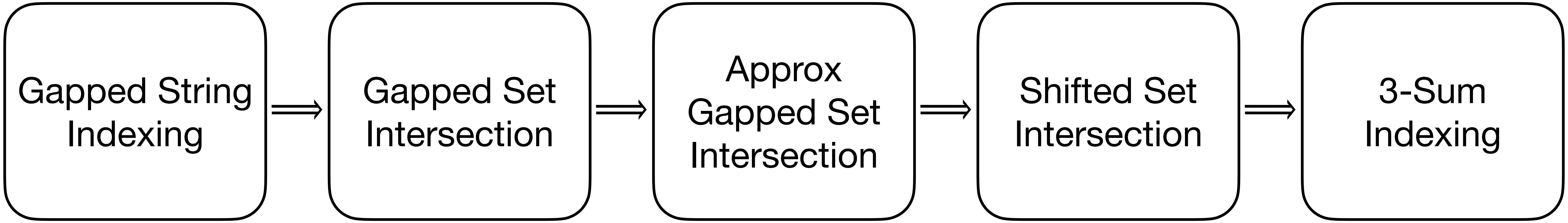
- Algorithm idea.
  - 3-sum existence query returns a certificate.
  - Output certificate and recurse on subproblems.



A1
A2
A
B
B1
B2
A1
B2
A2
B1

3-Sum Indexing with Reporting

- Theorem.
  - 3-sum indexing with reporting with $\tilde{O}(n^{2-\delta/3})$ space and $\tilde{O}(n^{\delta}(occ+1))$ query time.


Gapped String Indexing
Gapped Set Intersection
Approx Gapped Set Intersection
Shifted Set Intersection
3-Sum Indexing

3-Sum Indexing with Reporting

- Theorem.
  - 3-sum indexing with reporting with $\tilde{O}(n^{2-\delta/3})$ space and $\tilde{O}(n^{\delta}(occ+1))$ query time.

- Theorem.
  - Gapped string indexing with reporting with $\tilde{O}(n^{2-\delta/3})$ space and $\tilde{O}(|P_1| + |P_2| + n^{\delta}(occ+1))$ query time.

Gapped String Indexing

- Conclusion.
  - Gapped string indexing with reporting with $\tilde{O}(n^{2-\delta/3})$ space and $\tilde{O}(|P_1| + |P_2| + n^{\delta}(occ+1))$ query time.
- Other results.
  - Alternative trade-off for gapped string indexing.
  - New trade-off for jumbled indexing.
  - Better trade-offs for one-sided intervals.
- Open problems
  - Can we take advantage of structure in gapped string indexing?